# On Orbits and Border of Constant Read Circuits and Lower Bounds for Constant Depth Circuits

A THESIS

SUBMITTED FOR THE DEGREE OF

**Doctor of Philosophy**

IN THE

**Faculty of Engineering**

BY

**Thankey Bhargav Deepakkumar**


Computer Science and Automation
Indian Institute of Science
Bangalore – 560 012 (INDIA)

February, 2025

# Declaration of Originality

I, **Thankey Bhargav Deepakkumar**, with SR No. **04-04-00-10-12-20-1-18577** hereby declare that the material presented in the thesis titled

**On Orbits and Border of Constant Read Circuits and Lower Bounds for Constant Depth Circuits**

represents original work carried out by me in the **Department of Computer Science and Automation** at **Indian Institute of Science** during the years **2020-2024**.
With my signature, I certify that:

- I have not manipulated any of the data or results.
- I have not committed any plagiarism of intellectual property. I have clearly indicated and referenced the contributions of others.
- I have explicitly acknowledged all collaborative research and discussions.
- I have understood that any false claim will result in severe disciplinary action.
- I have understood that the work may be screened for any form of academic misconduct.

Date: Student Signature

In my capacity as supervisor of the above-mentioned work, I certify that the above statements are true to the best of my knowledge, and I have carried out due diligence to ensure the originality of the report.

Advisor Name: Advisor Signature

DEDICATED TO

*My Parents*

*For their constant love and support*

# Acknowledgments

First and foremost I would like to thank my advisor Chandan Saha. He has been an amazing mentor to me for the last six years; the many invaluable lessons I have learned from him during this time have helped me become a better researcher. I will forever treasure all that I have learned from him about thinking and expressing my ideas clearly. The discussions that I have had with him about, research, life, and career have helped me immensely in making an informed decision about my future endeavors. I can not thank him enough for all the support and motivation he has provided me during my M.Tech. and Ph.D. journey.

I would also like to thank the faculty members in the Department of Computer Science and Automation for the many interesting courses that they have taught during my M.Tech. and Ph.D. years. I would especially like to thank Prof. Siddharth Barman, Prof. Anand Louis, and Prof. Arindam Khan for the host of algorithms courses that I have been fortunate enough to take in the last six and a half years. I would also like to thank Prof. Chaya Ganesh for her course on proof systems in cryptography.

I thank my collaborators Prashnath Amireddy, Pranjal Dutta, Ankit Garg, Nikhil Gupta, Neeraj Kayal, Vineet Nair, Chandan Saha, and Ashish Sharma. I had a wonderful time working with them and have learned a lot from all the brainstorming meetings we had during the course of our collaborations. I am especially thankful to Nikhil Gupta who has been a frequent collaborator during my M.Tech. and Ph.D. I admire his tenacity and his ability to meticulously work out all the little details of long and complex arguments. I would also like to thank Vineet Nair for introducing me to the area of practical zero knowledge proofs.

I thank the organizers of the Workshop in Algebraic Complexity (WACT) 2019, the workshop on Sensitivity, Query Complexity, Communication Complexity and Fourier Analysis of Boolean Function 2020, the Academic Research and Careers for Students (ARCS) symposium 2020, and Bangalore Crypto Day 2024 for giving me the opportunity to attend these

informative and engaging events.

I have made many wonderful friends during my stay in IISc; I thank all of them for making life at IISc fun and enjoyable. I would especially like to thank Neha Jawalkar for the countless long, usually late night, walks and conversations we have had on various topics ranging from cryptography to the meaning of life and everything in between. I also thank Sruthi Gorantala, a helpful senior turned fun batch mate, frequent collaborator on homework assignments, and project partner. I also thank Nikita Yadav who has been a good friend since my M.Tech. days. I thank Nikhil Gupta for being a great collaborator and friend and for all the many discussions we had about research and just life in general. Finally I would like to thank B Prateek, Alok Kumar, Stanly Samuel, Arka Ray, Swetha Pandey, Aditya Subramanian, Akash Panda, and Swati Allabadi for making my stay in IISc memorable.

Last but not the least, I thank my family, my parents, sister, grandmother, and (late) grandfather, without whose constant love and support none of these would have been possible. I can not express in words how grateful I am for all that they have done for me.

# Abstract

Two of the most common ways in which arithmetic circuits can be restricted is by requiring that they be constant read or constant depth. Over the last decade, constant depth circuits have received a lot of attention in the arithmetic circuit literature as proving strong enough lower bounds for constant depth arithmetic circuits imply lower bounds for general arithmetic circuits. The orbits of constant read models are extremely interesting because the orbit closures (i.e. the border of the orbit) of some constant read models capture arithmetic formulas and algebraic branching programs. In this thesis we study some problems about the orbits and borders of certain constant read models as well as the lower bound problem for constant depth arithmetic circuits.

The orbit of $f \in \mathbb{F}[x_1, \ldots, x_n]$ is defined as $\text{orb}(f) := \{f(A\mathbf{x} + \mathbf{b}) : A \in \text{GL}(n, \mathbb{F}), \mathbf{b} \in \mathbb{F}^n\}$, where $\mathbf{x} = (x_1 \ldots x_n)^T$. The orbit of a circuit class $\mathcal{C}$ is defined as the union of the orbits of all polynomials computable by circuits in $\mathcal{C}$. In our first work, we study the (black-box) PIT problem for the orbits of low individual degree commutative ROABPs, multilinear constant width ROABPs, constant depth constant occur formulas, and occur once formulas. We give quasi-polynomial time PIT algorithms for all four models. The PIT algorithm for the first model implies a quasi-polynomial time PIT algorithm for orbits of elementary symmetric polynomials and multilinear sparse polynomials. A quasi-polynomial time PIT algorithm for the orbits of $\{\text{IMM}_{3,d}\}_{d\in\mathbb{N}}$ is obtained as a corollary to the PIT algorithm for the orbits of multilinear constant width ROABPs. Also, the PIT algorithm for the third model yields a quasi-polynomial time PIT algorithm for orbits of multilinear depth 4 circuits with constant top fan-in as well as polynomial time PIT algorithms for the orbits of the sum-product polynomials and power symmetric polynomials. These results are obtained by building upon and strengthening the rank concentration by translation technique of [ASS13].

In our second work, we study the equivalence test problem for read-once arithmetic formulas (ROFs) as well as the polynomial equivalence problem for the orbits of ROFs. Two polynomials $f, g$ are said to be equivalent if $g \in \text{orb}(f)$. The equivalence test problem for ROFs is as follows: given black-box access to an $n$-variate polynomial $f$ check if it is in

the orbit of an ROF. If yes, output an ROF $\mathtt{C}$ as well as $A \in \mathrm{GL}(n, \mathbb{F}), \mathbf{b} \in \mathbb{F}^n$ such that $f = \mathtt{C}(A\mathbf{x} + \mathbf{b})$. In other words, it is the circuit reconstruction problem for orbits of ROFs. The polynomial equivalence problem for the orbits of ROFs is the following: given black-box access to $n$-variate polynomials $f, g$ in the orbits of two *unknown* ROFs, determine if $f$ and $g$ are equivalent. If yes, then output $A \in \mathrm{GL}(n, \mathbb{F}), \mathbf{b} \in \mathbb{F}^n$ such that $g = f(A\mathbf{x} + \mathbf{b})$. We give the first randomized polynomial time equivalence testing algorithm for ROFs. We also give the first randomized polynomial time algorithm for the polynomial equivalence problem for orbits of additive constant free ROFs which are a slightly restricted class of ROFs. These results are obtained by analyzing the Hessian determinant and the set of essential variables of an ROF.

In our third work, we study the lower bounds problem for constant depth arithmetic circuits. In a breakthrough work, Limaye, Srinivasan, and Tavenas [LST21] resolved the long-standing open problem of proving super polynomial lower bounds for constant depth arithmetic circuits. Their proof involves a hardness escalation step wherein the problem of proving lower bounds for constant depth homogeneous circuits is reduced to the problem of proving lower bounds for set-multilinear circuits. We give a more direct proof of their result by avoiding this hardness escalation step. As this step introduces an exponential blow-up in the circuit size, our proof opens up the possibility of proving exponential lower bounds for constant depth homogeneous circuits. Our lower bounds hold for the IMM and Nisan-Wigderson design polynomials. They are obtained by directly upper bounding the shifted partials and affine projection of partials (APP) measures of constant depth homogeneous circuits. We hope that this would help in obtaining average case reconstruction algorithms for constant depth homogeneous circuits via the lower bounds to learning framework introduced in [KS19a, GKS20, BGKS22].

Finally, in our fourth work, we study the border of sums of 2 ROFs. The border of a circuit class $\mathcal{C}$ consists of all polynomials that can be "approximated" by a sequence of polynomials computable by circuits in $\mathcal{C}$. The border of ROFs is contained in ROFs. We show that the class of sum of $k$ many $n$-variate ROFs is not closed under the border for $2 \leq k \leq \frac{n}{5}$. Nevertheless, we prove that the border of sums of 2 $n$-variate additive constant free ROFs is contained in the sum of $O(n)$ many ROFs. We also give a quasi-polynomial time PIT algorithm for the border of sum of 2 homogeneous depth-5 ROFs.

# Publications based on this Thesis

1. Low-Depth Arithmetic Circuit Lower Bounds: Bypassing Set-Multilinearization,
   Joint work with Prashanth Amireddy, Ankit Garg, Neeraj Kayal and Chandan Saha,
   Appeared in 50-th International Colloquium on Automata, Languages, and Programming (ICALP), 2023.

2. Equivalence Test for Read-Once Arithmetic Formulas,
   Joint work with Nikhil Gupta and Chandan Saha,
   Appeared in 34-th ACM-SIAM Symposium on Discrete Algorithms (SODA), 2023.

3. Hitting Sets for Orbits of Circuit Classes and Polynomial Families,
   Joint work with Chandan Saha,
   ACM Transactions on Computing Theory (ToCT), Volume 16, 2024.
   Conference version appeared in 25-th International Conference on Randomization and Computation (RANDOM), 2021.

4. On the border of sums of constantly many Read-Once Arithmetic Formulas,
   Joint work with Pranjal Dutta,
   Manuscript under preparation.

The content of the following pre-print and publication are not included in this thesis.

1. BrakingBase - a linear prover, poly-logarithmic verifier, field agnostic polynomial commitment scheme,
   Joint work with Vineet Nair and Ashish Sharma,
   Cryptology ePrint Archive, 2024.

2. A Super-Quadratic Lower Bound for Depth Four Arithmetic Circuits,
   Joint work with Nikhil Gupta and Chandan Saha,
   Appeared in 35-th Computational Complexity Conference (CCC), 2020.

# Contents

**Acknowledgments** i

**Abstract** iii

**Publications based on this Thesis** v

**Contents** vi

**List of Tables** x

**1 Introduction** 1

1.1 Algebraic complexity theory . . . 3

1.1.1 Lower bounds . . . 3

1.1.2 Polynomial identity testing . . . 6

1.1.3 Circuit reconstruction . . . 11

1.1.4 Geometric complexity theory, orbits and borders of polynomials . . . 13

1.2 Our results . . . 16

1.2.1 Hitting sets for orbits of circuit classes . . . 16

1.2.2 Equivalence test for read-once arithmetic formulas . . . 21

1.2.3 Lower bounds for constant depth arithmetic circuits . . . 24

1.2.4 Border of the sums of two ROFs . . . 26

1.3 Organisation of the thesis . . . 26

**2 Preliminaries** 28

2.1 Arithmetic models of computation . . . 28

2.2 Algebraic complexity classes . . . 31

2.3 Polynomial families . . . 32

2.4 Complexity measures . . . 33

2.5 Hitting sets . . . 35
2.5.1 The Shpilka-Volkovich generator . . . 35
2.5.2 Low support rank concentration . . . 37
2.5.3 Algebraic rank and faithful homomorphisms . . . 38
2.6 Read-once arithmetic formulas . . . 39
2.7 The Hessian of a polynomial . . . 41
2.8 Essential and redundant variables . . . 42
2.9 The orbit of a polynomial . . . 46
2.10 The border of a circuit class . . . 47

**3 Hitting sets for orbits of ROABPs** **49**
3.1 Introduction . . . 49
3.2 Hitting sets for the orbits of commutative ROABPs . . . 50
3.2.1 The goal: low support rank concentration . . . 52
3.2.2 Achieving rank concentration . . . 53
3.2.3 Proof of Theorem 3.2 . . . 59
3.2.4 Proofs of Theorem 1.1 . . . 60
3.2.5 Proofs of Theorem 1.2 . . . 60
3.2.6 Proof of Theorem 3.1 . . . 61
3.3 Hitting sets for the orbits of multilinear constant-width ROABPs . . . 61
3.3.1 Low support rank concentration: an inductive argument . . . 63
3.3.2 Details of the induction step . . . 65
3.3.3 Proof of Theorem 1.3 . . . 71

**4 Hitting sets for orbits of constant occur formulas** **72**
4.1 Introduction . . . 72
4.2 Hitting sets for the orbits of depth-4, constant-occur formulas . . . 73
4.2.1 Upper bounding the top fan-in of $f$ . . . 74
4.2.2 Constructing a faithful homomorphism for the orbits . . . 76
4.2.3 Proof of Theorem 1.4: the depth-4 case . . . 78
4.3 Hitting sets for the orbits of constant-depth, constant-occur formulas . . . 78
4.3.1 Upper bounding the top fan-in of $f$ . . . 79
4.3.2 Constructing a faithful homomorphism . . . 81
4.3.3 Proof of Theorem 1.4 . . . 82
4.4 Hitting sets for the orbits of occur-once formulas . . . 83

4.4.1 Structural results . . . 84
4.4.2 Proof of Theorem 1.5 . . . 86

**5 Equivalence test for read-once arithmetic formulas 88**
5.1 Introduction . . . 88
5.2 Proof techniques . . . 89
5.2.1 A basic approach . . . 90
5.2.2 Outline of the ROF equivalence test: Salvaging the basic approach . . . 91
5.3 Preliminaries . . . 98
5.3.1 Structural preliminaries . . . 99
5.3.2 Algorithmic preliminaries . . . 105
5.4 The Hessian of an ROF . . . 108
5.5 Equivalence test for ROFs . . . 118
5.5.1 An overview of the algorithm . . . 119
5.5.2 The algorithm . . . 126
5.5.3 Analysis of the algorithm . . . 128
5.6 ROF reconstruction . . . 159
5.6.1 The algorithm . . . 160
5.6.2 Analysis of the algorithm . . . 162
5.6.3 Canonization: Recovering scaling and translation . . . 165
5.7 A pictorial overview of Algorithm 3 . . . 167

**6 Polynomial equivalence for orbits of read-once arithmetic formulas 171**
6.1 Introduction . . . 171
6.2 Polynomial Equivalence for orbits of additive constant free ROFs . . . 172
6.2.1 Rooted tree isomorphism . . . 172
6.2.2 The algorithm . . . 172
6.2.3 Analysis of the algorithm . . . 173
6.3 Polynomial equivalence for orbits of product depth-2 ROFs . . . 174
6.3.1 Overview of the algorithm . . . 174
6.3.2 The Algorithm . . . 177
6.3.3 Analysis of the algorithm . . . 179

**7 Lower bounds for constant depth arithmetic circuits 181**
7.1 Introduction . . . 181
7.2 Preliminaries . . . 182

7.3 Proof techniques . . . 184
7.4 Structure of the space of partial derivatives of a product . . . 187
7.5 Lower bound for low-depth homogeneous formulas . . . 195
7.5.1 Decomposition of low-depth formulas . . . 196
7.5.2 Low-depth formulas have high residue . . . 199
7.5.3 High residue implies lower bounds . . . 202
7.5.4 The hard polynomials . . . 205
7.5.5 Putting everything together: the low-depth lower bound . . . 212
7.5.6 A non-set-multilinear hard polynomial . . . 215

**8 Border of sums of constantly many read-once arithmetic formulas 217**
8.1 Introduction . . . 217
8.2 Lower bounds for sums of ROFs against their border . . . 218
8.3 De-bordering the border of sum of 2 ROFs . . . 223
8.4 PIT for the border of sum of 2 depth-4 ROFs . . . 229
8.5 PIT for the border of sum of 2 depth-5 ROFs . . . 233

**9 Directions for future work 238**

**Bibliography 242**

# List of Tables

5.1 Notations . . . . . 99

# Chapter 1

# Introduction

The central open question in the area of computational complexity is the celebrated $\mathsf{P} \stackrel{?}{=} \mathsf{NP}$ problem which asks whether there is a computational problem that can be solved efficiently by a non-deterministic Turing machine but not by a deterministic Turing machine. The most widely used method for proving that Turing machines can not perform some computation in a given amount of time is the celebrated diagonalization technique introduced by Georg Cantor in 1894 to show that there are strictly more real numbers than rational numbers. When Alan Turing introduced the notion of Turing machines, he used diagonalization to prove the existence of computational problems that can not be solved by any Turing machine. In particular, he showed that the Halting problem, which asks whether a Turing machine $M$ will ever halt if run with input string $x$, can not be decided by any Turing machine. In doing so, he gave a negative answer the Entscheidungsproblem posed by David Hilbert and Wilhelm Ackermann. In subsequent decades, diagonalization was used to prove a wide variety of results showing various limitations of Turing machines. Some examples of these results are the "hierarchy" theorems [HS65, Coo72] for both deterministic and non-deterministic Turing machines as well as Ladner's result [Lad75] establishing the existence of problems that are neither in $\mathsf{P}$ nor in $\mathsf{NP}$ if $\mathsf{P} \neq \mathsf{NP}$. However, in 1975 Baker, Gill, and Solovay showed that the diagonalization technique has some serious limitations [BGS75]. They proved that diagonalization can not be used to resolve the $\mathsf{P} \stackrel{?}{=} \mathsf{NP}$ question.[1] This result is known as the relativization barrier.

After the discovery of the relativization barrier, computational complexity theorists turned their attention to studying models of computation like Boolean circuits. A Boolean circuit is a directed acyclic graph whose nodes are called gates and edges are called wires. Every

[1] Interestingly this limitation of diagonalization was itself proved using diagonalization!

gate with in degree 0 is labelled by a variable or 0/1 and these are called the input gates. All other gates are AND, OR, or NOT gates. Gates with out degree 0 are the output gates. A Boolean circuit with $m$ output gates and with $n$ variables naturally computes a function $f : \{0,1\}^n \to \{0,1\}^m$.[1] The proof of the well-known Cook-Levin theorem [Coo71, Lev73] shows that the computation performed by any Turing machine $M$ can be captured by a family of Boolean circuits $\{\mathtt{C}_n\}_{n\in\mathbb{N}}$, with $\mathtt{C}_n$ capturing the computation performed by $M$ on inputs of size $n$. Thus, one can certainly hope that studying the limitations of Boolean circuits will help us better understand the limitations of Turing machines. Researchers have defined many sub-classes of Boolean circuits and studied the relative computational powers of these sub-classes. Decades of efforts have resulted in a rich body of literature on the limitations of these sub-classes of circuits and about the relationships between them.

The computational model studied in this thesis is an algebraic analogue of Boolean circuits variously called arithmetic circuits, algebraic circuits, or straight line programs. Just like a Boolean circuit, an arithmetic circuit is also a directed acyclic graph with input and output gates. However, in an arithmetic circuit the input gates are labeled by either variables or elements from some field $\mathbb{F}$. Also, instead of AND, OR, and NOT gates, an arithmetic circuit has $+$ and $\times$ gates. An arithmetic circuit (with one output gate) naturally computes a polynomial. The polynomial computed by an arithmetic circuit can be defined recursively as follows: All input gates compute the variable or field element labeling them. A $+$ gate computes the sum of its inputs and a $\times$ gate computes a product of its inputs. The polynomial computed by the circuit is the polynomial computed by its output gate. The size of a circuit is the number of edges in it while the depth of a circuit is the length of the longest path in it. See Definition 2.1 for a formal definition of an arithmetic circuit.

Just as Boolean circuits can capture the computation performed by Turing machines, arithmetic circuits can be used to capture Boolean circuits: an AND gate with inputs $f_1, \ldots, f_k$ can be replaced by a $\times$ gate with the same inputs. A NOT gate with input $f$ can be replaced by a $+$ gate with inputs $f$ and $-1$, while every OR gate can first be replaced by AND and NOT gates using De-Morgan's laws and then these gates can be replaced by $+, \times$ gates as just described. Note that an arithmetic circuit constructed in this manner from a Boolean circuit agrees with the Boolean circuit on all Boolean inputs. This method of converting Boolean computation into algebraic computation is called arithmetization and has been widely used in complexity theory as well as cryptography to great effect. A couple of examples of this are the proof of $\mathsf{IP} = \mathsf{PSPACE}$ [Sha92] and the vast body of literature on succinct zero-knowledge proofs (see [Tha22] for a survey). Furthermore, arithmetic circuits

[1] Generally, the Boolean circuits studied in computational complexity theory have a single output gate.

are a natural model for studying many problems with an algebraic flavour frequently encountered in computer science. Some such problems are matrix multiplication, determinant computation, polynomial multiplication, interpolation, and factorization. Algorithms for these problems can be described as a sequence of additions and multiplications and therefore, as arithmetic circuits.

In his seminal 1979 paper [Val79], Leslie Valiant introduced the algebraic complexity classes VP and VNP and initiated the study of computational hardness for arithmetic circuits; VP and VNP can be thought of as the algebraic versions of P and NP.[1] To define VP and VNP we first fix a field $\mathbb{F}$. Then $\mathsf{VP}_{\mathbb{F}}$ is a class of all polynomial families $\{f_n\}_{n\in N}$ over $\mathbb{F}$ such that there exists a polynomial function $t : \mathbb{N} \to \mathbb{N}$, such that for all $n \geq 1$, $f_n$ is a polynomial in at most $t(n)$ variables, of degree at most $t(n)$ and computed by an arithmetic circuit of size at most $t(n)$. $\mathsf{VNP}_{\mathbb{F}}$ is a class of all polynomial families $\{f_n\}_{n\in N}$ over a field $\mathbb{F}$ such that there exist polynomial functions $k, t : \mathbb{N} \to \mathbb{N}$ and a family of polynomials $\{g_n\}_{n\in N} \in \mathsf{VP}_{\mathbb{F}}$ such that for all $n \geq 1$,

$$f_n(x_1, ..., x_{k(n)}) = \sum_{w\in\{0,1\}^{t(n)}} g_{t(n)}(x_1, ..., x_{k(n)}, w_1, ..., w_{t(n)}).$$

It is easy to see that $\mathsf{VP} \subseteq \mathsf{VNP}$; Valiant conjectured that this containment is strict. To date, the conjecture remains wide open and is the central open problem in algebraic complexity. In the following section, we discuss the considerable body of literature that has been developed about arithmetic circuits as well as the progress made towards this conjecture.

## 1.1 Algebraic complexity theory

We divide the discussion of the literature on arithmetic circuits into three fundamental categories: results on lower bounds for arithmetic circuits, polynomial identity testing algorithms, and circuit reconstruction algorithms. These are discussed in Sections 1.1.1, 1.1.2, and 1.1.3 along with the close relationships between these categories. In Section 1.1.4, we talk briefly about geometric complexity theory and introduce orbits and borders, two objects studied in this thesis.

### 1.1.1 Lower bounds

Do there exist polynomial families $\{f_n\}_{n\in\mathbb{N}}$, where $f_n$ is a polynomial in $\mathrm{poly}(n)$ variables and of $\mathrm{poly}(n)$ degree such that for any circuit family $\{\mathsf{C}_n\}_{n\in\mathbb{N}}$ where $\mathsf{C}_n$ computes $f_n$, $\mathsf{C}_n$ must have size which is super-polynomial in $n$? The answer to this question is a resounding

[1] Actually VP is much closer to the class NC and VNP to the class #P.

yes; it can be shown that *most* polynomial families can not be computed by families of circuits with polynomial size. The lower bounds problem asks this very question but requires that the polynomial family $\{f_n\}_{n\in\mathbb{N}}$ be "explicit". There are a few ways to formalize the notion of an explicit polynomial family; in this thesis, when we say that a polynomial family is explicit, we mean that it is in VNP. Note that since VP is the class of all polynomial families of polynomial degree computed by families of polynomial sized circuits, the lower bounds problem is just another way to phrase the question of whether $\mathsf{VP} \stackrel{?}{=} \mathsf{VNP}$.

The $\mathsf{VP} \stackrel{?}{=} \mathsf{VNP}$ problem has some direct implications on the lower bounds question for Boolean circuits. Burgisser has shown that if $\mathsf{VP} = \mathsf{VNP}$ over a finite field then the non-uniform versions of P and NP, viz. $\mathsf{P/poly}$ and $\mathsf{NP/poly}$ are also the same [Bür00]. Burgisser also proved a similar result for fields of characteristic 0 in the same work, but this result assumes the Generalised Riemann Hypothesis. It is known that if $\mathsf{P/poly} = \mathsf{NP/poly}$, then the polynomial hierarchy collapses to the second level. Thus, it is unlikely that $\mathsf{VP} = \mathsf{VNP}$ and we expect the lower bounds question to have an affirmative answer. The lower bounds question also has deep connections to the existence of efficient polynomial identity testing and circuit reconstruction algorithms, but we defer a discussion of these connections to Sections 1.1.2 and 1.1.3, respectively.

To make progress towards the $\mathsf{VP} \stackrel{?}{=} \mathsf{VNP}$ problem, researchers have defined many restricted classes of arithmetic circuits and proved various lower bounds for these circuit classes. They have also shown that proving strong enough lower bounds for some of these restricted classes of circuits would actually imply that $\mathsf{VP} \neq \mathsf{VNP}$. We now give a brief account of various known lower bound results.

**Prior work**[1]

**General circuits and formulas.** Not much is known about lower bounds for general arithmetic circuits and formulas computing explicit polynomials. Baur and Strassen [BS83, Str73] proved that any arithmetic circuit computing the power symmetric polynomial ($\mathsf{PSym}_{n,d} := x_1^d + \cdots + x_n^d$) or the elementary symmetric polynomial ($\mathsf{ESym}_{n,d} := \sum_{\substack{S\subseteq[n]:\\ |S|=d}} \prod_{i\in S} x_i$) must have size $\Omega(n \log d)$; both these lower bounds are tight. However, we do expect there to be polynomials in VNP which require exponential sized circuits. Yet, to date no lower bound stronger than the $\Omega(n \log d)$ is known for general arithmetic circuits.

The best-known lower bound for general arithmetic formulas is quadratic.[2] [Kal85]

[1] This account of the known lower bounds result appears in a joint work with Prashanth Amireddy, Ankit Garg, Neeraj Kayal, and Chandan Saha [AGK+23].

[2] Arithmetic formulas are arithmetic circuits whose underlying graph is a tree.

proved that any arithmetic formula computing the polynomial $\sum_{i,j\in[n]} x_i^j y_j$ must have size $\Omega(n^2)$. Recently, [CKSV22] proved an $\Omega(n^2)$ lower bound for arithmetic formulas computing $\mathrm{ESym}_{n,0.1n}$. They also showed that any 'layered' Algebraic Branching Program (ABP) computing $\mathrm{PSym}_{n,n}$ has size $\Omega(n^2)$. ABPs are algebraic analogues of (Boolean) branching programs and, as a model of computation, are known to be at least as powerful as formulas. Because of the apparent difficulty of proving lower bounds for general models of computation, restricted classes of circuits like multilinear, homogeneous, and low-depth circuits have received a lot of attention in the last few decades. We now discuss a few results for these models.

**Multilinear and set-multilinear circuits.** A circuit or formula is said to be multilinear if every gate in it computes a multilinear polynomial. [RSY08] showed a lower bound of $\Omega(n^{4/3}/\log^2 n)$ for syntactically multilinear circuits. This lower bound was improved to an $\Omega(n^2/\log^2 n)$ bound in [AKV18]. Unlike the case of general circuits and formulas, a super-polynomial separation is known between multilinear circuits and formulas. [Raz06, RY08] proved that there exists a polynomial computable by polynomial-size multilinear circuits but can only be computed by multilinear formulas of size $n^{\Omega(\log n)}$. [DMPY12] showed a similar lower bound but for a polynomial computable by a polynomial-size multilinear ABP.

Exponential lower bounds are known for low-depth multilinear circuits. A lower bound of $2^{n^{\Omega(1/\Delta)}}$ for multilinear circuits of product-depth $\Delta = o(\log n/\log\log n)$ computing the $n \times n$ permanent and determinant was shown in [RY09]. [CLS19] proved a lower bound of $2^{\Omega(\Delta d^{1/\Delta})}$ for multilinear circuits of product-depth at most $\Delta \leq \log d$ computing $IMM_{2,d}$. A quasi-polynomial separation between product-depth $\Delta$ multilinear circuits and product-depth $\Delta+1$ multilinear circuits was proved in [RY09] and improved to an exponential separation in [CELS18].

Notice that the lower bounds mentioned in the previous paragraph are of the form $n^{O(1)} \cdot f(d)$ where $f(d)$ is a superpolynomial but sub-exponential function of the degree. Borrowing terminology from parameterised complexity, [LST21] calls such lower bounds FPT lower bounds. As pointed out in [LST21], it is unclear if FPT bounds can be used to prove lower bounds for low-depth circuits. [LST21] and later [BDS22] prove a non-FPT lower bound of $n^{d^{\exp(-\Delta)}}$ for set-multilinear formulas of product-depth $\Delta$ computing $\mathsf{IMM}_{n,d}$ when $d = O(\log n)$. These lower bounds are then used to prove super-polynomial lower bounds for low-depth circuits. In [TLS22], a non-FPT lower bound of $(\log n)^{\Omega(\Delta d^{1/\Delta})}$ is proved for set-multilinear formulas of product-depth $\Delta = O(\log d)$ computing $\mathsf{IMM}_{n,d}$. They also prove a lower bound of $(\log n)^{\Omega(\log d)}$ for any set-multilinear formula computing

$\mathsf{IMM}_{n,d}$. [KS22] proved a lower bound of $n^{\Omega\left(d^{1/\Delta}/\Delta\right)}$ for set-multilinear formulas of product-depth $\Delta$ computing the Nisan-Wigderson design polynomial. Later [KS23b] improved this result by obtaining an $n^{\Omega\left(d^{1/\Delta}/\Delta\right)}$ lower bound for depth $\Delta$ set-multilinear formulas computing a polynomial $P_n$ which can be computed by a set-multilinear ABP. They also show that any set-multilinear formula of arbitrary depth computing $P_n$ must have size $n^{\Omega(\log d)}$.

**Homogeneous and low-depth circuits.** A long line of work [VSBR83, AV08, Koi12, GKKS16, Tav15] on depth-reduction results has shown if an $n$-variate, degree-$d$ polynomial can be computed by an arithmetic circuit of size $s$, then it can also be computed by a $\Sigma\Pi\Sigma$[1] circuit of size $s^{O(\sqrt{d})}$ as well as by a homogeneous $\Sigma\Pi\Sigma\Pi$[2] circuit of size $s^{O(\sqrt{d})}$. Thus, an $n^{\Omega(\sqrt{d})}$ lower bound for either $\Sigma\Pi\Sigma$ circuits or homogeneous $\Sigma\Pi\Sigma\Pi$ circuits would yield a separation between VP and VNP. Motivated by these depth reduction results, a significant body of work has developed for lower bounds against constant depth circuits.

[SS97, Raz10] proved a lower bound of $\Omega(\Delta n^{1+1/\Delta})$ for depth $\Delta$ circuits with multiple output gates. In a classic work [NW97], Nisan and Wigderson showed that any homogeneous depth 3 circuit computing $\mathsf{ESym}_{n,d}$ has size $\left(\frac{n}{d}\right)^{\Omega(d)}$. A series of papers [Kay12b, GKKS14, KSS14, FLMS15, KLSS17, KS17b] resulted in an $n^{\Omega(\sqrt{d})}$ lower bound for homogeneous depth 4 circuits computing the Nisan-Wigderson design polynomial and $\mathsf{IMM}_{n,d}$.

[SW01] proved a quadratic lower bound for depth 3 circuits computing elementary symmetric polynomials of degree $\Omega(n)$. This was improved to an almost cubic lower bound in [KST16a] for a polynomial in VNP. Subsequently, [BLS16, Yau16] proved similar lower bounds for polynomials in VP. [GST20] obtained a lower bound of $\widetilde{\Omega}(n^{2.5})$ for depth 4 circuits computing the Nisan-Wigderson design polynomial. In a recent breakthrough work [LST21], Limaye, Srinivasan, and Tavenas proved a lower bound of $n^{\Omega(d^{1/(2^{\Delta}-1)}/\Delta)}$ for product-depth $\Delta$ circuits computing $\mathsf{IMM}_{n,d}$, $d = O(\log n)$. [BDS22] improved this to a lower bound of $n^{\Omega(d^{1/\phi^{2\Delta}}/\Delta)}$ where $\phi = (\sqrt{5}+1)/2 \approx 1.618$.

We refer the reader to an excellent survey on lower bounds by Saptharishi (and other contributors) [Sap] for more details about the above mentioned results.

### 1.1.2 Polynomial identity testing

Given a polynomial $f$, is there an efficient algorithm that can determine whether $f$ is identically zero or not? We say that a polynomial is identically zero (denoted by $f \equiv 0$) if the coeffi-

[1] A depth 3 circuit with alternating layers of $+$ and $\times$ gates with a $+$ gate at the top.
[2] A depth 4 circuit with alternating layers of $+$ and $\times$ gates with a $+$ gate at the top.

cients of all the monomials in it are 0.[1] Algebraic geometers have known for at least a century and a half that if an $n$-variate polynomial $f$ is not identically zero and the field $\mathbb{F}$ it is defined over is sufficiently large, then it will be non-zero at most of the points $(a_1, \ldots, a_n) \in \mathbb{F}^n$. The celebrated Schwartz-Zippel lemma [Ore22, DL78, Sch80, Zip79] shows that if $f$ is a non-zero $n$-variate, degree $d$ polynomial then for any $S \subseteq \mathbb{F}$,

$$\Pr_{(a_1,\ldots,a_n)\in S^n}(f(a_1, \ldots, a_n) = 0) \leq \frac{d}{|S|}.$$

This gives a natural randomised polynomial time algorithm for checking if $f$ is identically zero or not: fix an $S \subseteq \mathbb{F}$, $|S| \geq 2d$, pick $(a_1, \ldots, a_n)$ uniformly at random from $S^n$ and check if $f(a_1, \ldots, a_n) \stackrel{?}{=} 0$. The Schwartz-Zippel lemma has played a crucial role in the proof of $\mathsf{IP} = \mathsf{PSPACE}$ [Sha92] and is also used in almost all succinct zero knowledge proofs.

The Polynomial Identity Testing (or PIT for short) problem asks if there is a deterministic polynomial time algorithm to determine whether a given polynomial is identically zero or not. Note that there are multiple ways to give a polynomial as input: it can be given as a list of coefficients, as an arithmetic circuit, or we can be given black-box/oracle access to the polynomial.[2] Since the PIT problem becomes trivial when the polynomial is given as a list of coefficients, we only consider the situations in which either a circuit computing the polynomial or black-box access is given. The PIT problem for these two situations is referred to as the white-box PIT problem and the black-box PIT problem, respectively. The latter is also known as the hitting-sets problem. A hitting set $\mathcal{H} \subseteq \mathbb{F}^n$ for a circuit class $\mathcal{C}$ is a set of points such that for any $f \not\equiv 0 \in \mathcal{C}$, there exists an $(a_1, \ldots, a_n) \in \mathcal{H}$ such that $f(a_1, \ldots, a_n) \neq 0$. It is not difficult to see that a deterministic polynomial time black-box PIT algorithm exists for a circuit class $\mathcal{C}$ if and only if there exists a polynomial time constructible hitting set for that class.

The PIT problem is one of the flagship de-randomization problems studied in theoretical computer science. It has connections to proving computational hardness results as well as to other important problems in algorithm design. The connection between PIT and computational hardness was established in [HS80, KI04, Agr05]. [KI04] showed that a sub-exponential time algorithm for the PIT problem in the white-box or the black-box setting implies that either $\mathsf{NEXP} \not\subseteq \mathsf{P/poly}$ or that the permanent polynomial can not be computed by

[1] Note that this is not the same as a polynomial evaluating to zero over the entire field; $x^2 - x$ over $\mathbb{F}_2$ is a good example of a polynomial that is zero over the entire field but is not identically zero.

[2] By black-box access to a polynomial $f \in \mathbb{F}[x_1, \ldots, x_n]$, we mean the ability to evaluate $f(a_1, \ldots, a_n)$ at any $(a_1, \ldots, a_n) \in \mathbb{F}^n$ in unit time.

polynomial sized arithmetic circuits. [HS80, Agr05] proved that a deterministic polynomial time algorithm for black-box PIT would imply that there exists a polynomial which requires exponential size circuits and whose coefficients can be computed in PSPACE. [KI04] showed that an exponential lower bound for arithmetic circuits would imply a quasi-polynomial time PIT algorithm. The celebrated deterministic primality testing algorithm of Agrawal, Kayal, and Saxena [AKS04] was obtained by de-randomizing an instance of a polynomial identity test from [AB03]. Further, a deterministic polynomial time algorithm for PIT can be used to de-randomize the isolation lemma of Mulmuley, Vazirani, Vazirani [MVV87]; this would yield a deterministic parallel algorithm for finding perfect matchings in graphs.

**Prior work**[1]

**Constant-depth models.** The polynomial-time hitting set construction for depth-2 circuits (i.e., sparse polynomials) in [KS01] is one of the widely used results in black-box PIT. Depth-3 circuit PIT has also received a lot of attention. [DS07] gave a quasi-polynomial time PIT algorithm for depth-3 circuits with constant top fan-in by showing a structural result on the rank[2] [KS07] improved the complexity to polynomial-time using a different method, which is based on a generalization of the Chinese Remaindering Theorem (CRT). The structural result of [DS07], along with the rank extractors of [GR08], played a central role in devising polynomial-time constructible hitting sets for depth-3 circuits with constant top fan-in over $\mathbb{Q}$ by [KS11], [KS09b], [SS13]. Ultimately, a combination of ideas from the CRT method and rank extractors led to a polynomial-time hitting set construction for the same model over any field [SS12, SS13]. Meanwhile, [Sax08], [Kay10] gave polynomial-time PIT for depth-3 powering circuits. Using ideas from [KS07], [Sax08], [SSS13] gave polynomial-time PIT for the sum of a depth-3 circuit with constant top fan-in and a *semi-diagonal* circuit (which is a special kind of a depth-4 circuit). [SSS09] showed that polynomial-time PIT (hitting sets) for the affine projections of $\mathsf{IMM}_{2,d}$ implies polynomial-time PIT (hitting sets) for depth-3 circuits.

A quasi-polynomial time hitting set for set-multilinear depth-3 circuits with known variable-partition was given by [FS12]. Independently and simultaneously, [ASS13] gave a quasi-polynomial time hitting set for set-multilinear depth-3 circuits with *unknown* variable-partition (and more generally, for constant-depth *pure* formulas of [NW97]) using a different technique, namely *rank concentration by translation*. Set-multilinear depth-3 circuits (in fact, pure formulas) form a subclass of ROABPs. [dOSlV16] gave subexponential-time hitting sets for

[1] This account of the known PIT result appears in a joint work with Chandan Saha [ST24].

[2] Rank of a depth-3 circuit is the number of linearly independent linear polynomials appearing in the "simple part" of a circuit.

multilinear depth-3 and depth-4 formulas (and more generally, for constant-depth multilinear regular formulas) by reducing the problem to constructing hitting sets for ROABPs. For multilinear depth-4 circuits with constant top fan-in, [KMSV13] gave a quasi-polynomial time hitting set. This was improved to a polynomial-time hitting set in [SV18]. Multilinear depth-4 circuits with constant top fan-in form a subclass of depth-4 constant-occur formulas. [ASSS16] gave a unifying method based on the algebraic independence technique introduced by [BMS13] to design polynomial-time hitting sets for both depth-3 circuits with constant top fan-in and constant-depth, constant-occur formulas over fields of large characteristic. Recently, [BSV23] gave a polynomial time hitting set for constant-occur, depth-4 formulas over any field. A generalization of depth-3 powering circuits to depth-4 is sums of powers of constant degree polynomials; Forbes [For15] gave a quasi-polynomial time hitting set for this model. Recently, a sequence of works by [PS21, PS20] and [Shp19] led to a polynomial-time hitting set for depth-4 circuits with top fan-in at most 3 and bottom fan-in at most 2 via a resolution of a conjecture of Gupta [Gup14], [BMS13] on the algebraic rank of the factors appearing in such circuits. [DDS21] gave a quasi-polynomial time PIT algorithm for depth 4 circuits with constant top and bottom fan-in. In a breakthrough paper, Limaye, Srinivasan, and Tavenas [LST21] proved super-polynomial lower bounds for low-depth circuits; this yields a sub-exponential hitting set for arithmetic circuits of depth $o(\log\log\log n)$.

**Constant-read models.** [SV15] initiated the study of PIT for read-once formulas. They gave a polynomial-time PIT algorithm and a quasi-polynomial time hitting set construction for sums of constantly many *preprocessed* read-once formulas (PROFs). The leaves of a PROF are labelled by univariate polynomials and every variable appears in at most one leaf; PROFs form a subclass of occur-once formulas. Later, a polynomial-time hitting set construction for the same model was given by [MV18]. A sum of $k$ ROFs is a special case of a multilinear read-$k$ formula. [AvMV15] gave a quasi-polynomial time hitting set construction for multilinear read-$k$ formulas. Their construction also works for multilinear *sparse-substituted* read-$k$ formulas, wherein the leaves are replaced by sparse polynomials and every variable appears in at most $k$ of the sparse polynomials. Observe that a sparse-substituted read-$k$ formula is an occur-$k$ formula (without the powering gates), however the arguments in [AvMV15] additionally require the multilinearity assumption.

A polynomial-time PIT for ROABPs follows from the PIT algorithm for non-commutative formulas [RS05]. [FS13] gave a quasi-polynomial time construction of hitting sets for ROABPs, when the order of the variables is known; prior to their work, a quasi-polynomial time hitting set for multilinear, constant-width, known-variable-order ROABPs was given by

[JQS10]. Building on the rank concentration by translation technique from [ASS13] and the merge-and-reduce idea from [FS13], [FSS14] gave a quasi-polynomial time hitting set construction for multilinear ROABPs (more generally, low individual degree ROABPs). Finally, [AGKS15] obtained a quasi-polynomial time constructible hitting set for ROABPs using a different and simpler method, namely *basis isolation*, which can be thought of as a generalization of the monomial isolation method in [KS01]. It was also shown later by [GKST17], [FGS18] that translation by a basis isolating weight assignment leads to rank concentration, and so, constructing a basis isolating weight assignment is a stronger objective than showing rank concentration by translation. This fact was used effectively by [GKST17] to design hitting sets for sums of constantly many ROABPs in quasi-polynomial time; they also gave a polynomial-time PIT algorithm for the same model. A conjunction of the basis isolation and the rank concentration techniques have been used to give more efficient constructions of hitting sets for ROABPs by [GG20], sometimes under additional restrictions on the model such as commutativity and constant-width [GKS17]. The latter work also gave a polynomial-time hitting set for constant-width ROABPs, when the order of the variables is known. For read-$k$ oblivious algebraic branching programs, [AFS$^+$18] obtained a subexponential-time PIT algorithm.

**Edmonds' model.** An important special case of PIT is the following problem first posed by Edmonds [Edm67]: given $f = \det(A_0 + \sum_{i\in[n]} x_i A_i)$, where $A_i \in \mathbb{F}^{n\times n}$ is a rank-1 matrix for every $i \in [n]$ and $A_0 \in \mathbb{F}^{n\times n}$ is an arbitrary matrix, check if $f = 0$. This case of PIT, which can be thought of as a generalization of PIT for determinants of read-once symbolic matrices, played an instrumental role in devising fast parallel algorithms for several problems such as perfect matching, linear matroid intersection and maximum rank matrix completion [Lov79, KUW86, MVV87, FGT16, ST17, NSV94, Mur93, GT20]. A polynomial-time PIT for this model is known [Edm79, Lov89, Mur93, Gee99, IKS10]. [GT20] gave a quasi-polynomial time hitting set via a certain derandomization of the Isolation Lemma of Mulmuley, Vazirani, Vazirani [MVV87]. It is interesting to note that hitting sets for the orbits of polynomials computable by this model imply hitting sets for the orbit of the determinant polynomial and also the orbit of the iterated matrix multiplication polynomial via a known reduction by Valiant [Val79] from ABPs to p-projections of the determinant polynomial family.

**Orbits and orbit closures.**[1] A polynomial-time hitting set for the *orbit* of the power symmetric polynomial $\mathsf{PSym}_{n,d} = x_1^d + \ldots + x_n^d$ was given by [KS21]. For the orbit closures of

[1] See Section 2.9 for a definition of the orbit of a polynomial.

polynomials that are computable by low-degree, polynomial-size circuits (i.e., VP circuits), [FS18], [GSS18] gave PSPACE constructions of hitting sets.

We refer the reader to the excellent surveys by Saxena [Sax09, Sax14], Shpilka and Yehudayoff [SY10] for more details on some of the results and the models mentioned above.

### 1.1.3 Circuit reconstruction

Arithmetic circuit reconstruction is the problem of learning an arithmetic circuit given black-box access to it. More precisely, given black-box access to an $n$-variate, degree-$d$ polynomial $f$, a circuit reconstruction algorithm has to output a circuit computing $f$, ideally in time $\text{poly}(n, d, s)$, where $s$ is the size of the smallest circuit computing $f$. When one considers the circuit reconstruction problem for a circuit class $\mathcal{C}$, two natural variants emerge: the proper reconstruction problem and the improper reconstruction problem. The proper circuit reconstruction problem requires that the circuit output by the algorithm also belong to the class $\mathcal{C}$, while the improper version of the problem imposes no such restriction. Both these versions of the circuit reconstruction problem have been studied for various special circuit classes. Our result on equivalence test for ROFs can be thought of as a circuit reconstruction algorithm for formulas in the average case. We talk about the equivalence test and polynomial equivalence problems in Section 1.1.4.1; here we shall discuss some of the reasons for studying the circuit reconstruction problem and document known circuit reconstruction algorithms.

The first reason that we shall discuss is the implication that a proper circuit reconstruction algorithm for a circuit class has for the Minimum Circuit Size Problem (MCSP, for short) for that circuit class. MCSP for arithmetic circuits is the following decision problem: Given the $\binom{n+d}{d}$ dimensional coefficient vector of an $n$-variate, degree-$d$ polynomial $f$ and an $s \in \mathbb{N}$, does $f$ have an arithmetic circuit of size $s$? It is known that the existence of cryptographically secure one-way functions implies that the Boolean version of the problem[1] is not in P [KC00]; in fact it is known that under this assumption, even an $n^{1-o(1)}$ approximation algorithm can not exist [AH19]. However, no such result is known in the algebraic setting. Observe that a $\text{poly}(n, d, s)$ time reconstruction algorithm for a circuit class $\mathcal{C}$ would imply an $s^{O(1)}$ approximation algorithm for MCSP for $\mathcal{C}$. This is so because the circuit output by a $\text{poly}(n, d, s)$ time algorithm must have size at most $\text{poly}(n, d, s)$.

Another compelling reason for studying the circuit reconstruction problem is its inter-

[1] In the Boolean MCSP, the input is the truth table of a Boolean function and one has to determine if there is a size $s$ Boolean circuit computing the given function.

play with the lower bounds problem. [FK09, Vol16] showed that a circuit reconstruction algorithm for a circuit class implies a super-polynomial lower bound for that class. Conversely, reconstruction algorithms for a lot of classes have relied heavily on the ideas used to prove lower bounds for those classes. However, a lot of such reconstruction algorithms are only known for the *average-case* version of the reconstruction problem. An average-case reconstruction algorithm for a circuit class $\mathcal{C}$ can learn random circuits picked from $\mathcal{C}$ using some "nice" probability distributions. An average-case reconstruction algorithm for set multilinear depth-3 circuits implies an average-case algorithm for tensor decomposition, while an average-case reconstruction algorithm for depth-3 powering circuits implies an average-case algorithm for the decomposition of symmetric tensors. It has also been observed recently that an average-case reconstruction algorithm for very special types of depth-4 circuits leads to an algorithm for learning random mixtures of Gaussians [GKS20] and subspace clustering [CGK+24, BESV24].

**Prior work**

**Constant depth circuits.** [KS01] gave a deterministic polynomial time circuit reconstruction algorithm for spare polynomials, i.e. for $\Sigma\Pi$ circuits. This result, in conjunction with the randomized polynomial time factorisation algorithm [KT90], immediately yields a randomized polynomial time learning algorithm for $\Pi\Sigma\Pi$ circuits. Circuit reconstruction algorithms are also known for $\Sigma\Pi\Sigma$ circuits with top fan-in 2. [Shp07] gave an algorithm over finite fields that runs in time which is quasi-polynomial in the number of variables and the degree of the polynomial as well as in $|\mathbb{F}|$, while [Sin16] gave an algorithm over fields of characteristic 0. In a follow up paper [Sin22] an algorithm that works over finite fields was given. Building upon [Shp07], [KS09a] gave a deterministic quasi-polynomial reconstruction algorithm for $\Sigma\Pi\Sigma$ circuits with constant top fan-in over finite fields. The case is open for characteristic 0 fields.

[KS19a] gave average-case reconstruction algorithms for homogeneous $\Sigma\Pi\Sigma$ circuits as well as for set-multilinear depth-3 circuits and depth-3 powering circuits. [GKS20, BHKX22, CGK+24] gave average-case reconstruction algorithms for sums of powers of low-degree polynomials. [BGKS22] gave an average-case polynomial time reconstruction algorithm for learning generalized depth three circuits.

**Multilinear circuits.** [Shp07] gave a randomized $\mathrm{poly}(n, |\mathbb{F}|)$ reconstruction algorithm for multilinear $\Sigma\Pi\Sigma$ circuits with top fan-in 2 over finite fields. [GKL12] extended this result to multilinear depth-4 circuits with top fan-in 2; their algorithm works over any field. [GKL11]

gave an average-case polynomial time reconstruction algorithm for multilinear formulas. [BSV20] gave a quasi-polynomial time reconstruction algorithm for depth-4 multilinear circuits with constant top fan-in. [BSV21] gave polynomial time reconstruction algorithms for multilinear depth-3 circuits with constant top fan-in for all fields of large enough or zero characteristic as well as for depth-3 powering circuits with constant top fan-in.

**ROFs and ROABPs.** A deterministic polynomial time reconstruction algorithm for read-once formulas is known [SV14, MV18]. Randomised polynomial time algorithms for learning ROABPs were obtained in [BBB+00, KS06]. [FS13] partially de-randomised these algorithms to obtain a quasi-polynomial time reconstruction algorithm for ROABPs.

**Other formulas and ABPs.** An average-case polynomial time reconstruction algorithm for fan-in 2 regular formulas (a.k.a. alternating normal form formulas) was given in [GKQ13]. [KNS19] gave an efficient average-case reconstruction algorithm for homogeneous constant width ABPs.

### 1.1.4 Geometric complexity theory, orbits and borders of polynomials

Geometric Complexity Theory (GCT) is an ambitious program pioneered by Mulmuley and Sohoni [MS01] to resolve the determinant versus permanent problem. The $n^2$-variate determinant polynomial $\mathsf{Det}_n$ is the determinant of an $n \times n$ symbolic matrix. Similarly, the $n^2$-variate permanent polynomial $\mathsf{Perm}_n$ is the determinant of an $n \times n$ symbolic matrix. It is conjectured that $\mathsf{Perm}_n$ is not in $\mathsf{aproj}(\mathsf{Det}_m)$ for any $m = \mathrm{poly}(n)$. The set $\mathsf{aproj}(\mathsf{Det}_m)$ is the set of affine projections of $\mathsf{Det}_m$, that is $\mathsf{aproj}(\mathsf{Det}_m) = \left\{\mathsf{Det}_m(A\mathbf{x} + \mathbf{b}) : A \in \mathbb{F}^{m^2 \times m^2}, \mathbf{b} \in \mathbb{F}^{m^2}\right\}$ where $\mathbf{x} = (x_1 \dots x_{m^2})^T$. This conjecture is a weaker version of Valiant's conjecture that $\mathsf{VP} \neq \mathsf{VNP}$. This is so because the determinant polynomial is complete for a sub-class of $\mathsf{VP}$ called $\mathsf{VBP}$ while the permanent polynomial is complete for $\mathsf{VNP}$. See Section 2.2 for a definition of $\mathsf{VBP}$.

Geometric complexity theory attempts to use ideas from algebraic geometry and representation theory to separate $\mathsf{Perm}_n$ and $\mathsf{aproj}(\mathsf{Det}_m)$ for any $m = \mathrm{poly}(n)$. It is conceivable that any proof of this separation that uses algebraic geometry would actually prove a stronger result; it will end up proving that $\mathsf{Perm}_n$ is not contained in the Zariski closure of $\mathsf{aproj}(\mathsf{Det}_m)$. The Zariski closure of any $X \subseteq \mathbb{F}^n$ is defined as follows: let $\mathcal{I}$ be the set of all polynomials in $\mathbb{F}[x_1, \dots, x_n]$ that vanish on $X$. Then, the Zariski closure of $X$ is the set of all points at which the polynomials in $\mathcal{I}$ vanish simultaneously. $\mathsf{aproj}(\mathsf{Det}_m)$ can be thought of as a subset of $\mathbb{F}^M$, where $M = \binom{m^2+m}{m}$, by identifying every polynomial in it with its co-

efficient vector. By the Zariski closure of $\text{aproj}(\text{Det}_m)$, we mean the Zariski closure of this subset of $\mathbb{F}^M$. The Zariski closure of any circuit class computing $n$-variate, degree $d$ polynomials can be defined in an analogous manner by identifying all polynomials computed by that circuit class with their coefficient vectors. The Zariski closure of a circuit class $\mathcal{C}$ is often called the *border* or just the closure of that circuit class and is denoted by $\overline{\mathcal{C}}$.

It can be shown that the Zariski closure of the affine projections of a polynomial is the same as the Zariski closure of its orbit (see Appendix E of [ST24] for an elementary proof over fields of characteristic 0). The orbit of a polynomial $f \in \mathbb{F}[x_1, \ldots, x_n]$ is defined as $\text{orb}(f) := \{f(A\mathbf{x} + \mathbf{b}) : A \in \text{GL}(n, \mathbb{F}), \mathbf{b} \in \mathbb{F}^n\}$. Thus, any proof of $\text{Perm}_n \notin \text{aproj}(\text{Det}_{\text{poly}(n)})$ using algebraic geometry must actually prove that $\text{Perm}_n \notin \overline{\text{orb}(\text{Det}_{\text{poly}(n)})}$, i.e. it must separate the permanent and the *orbit closure* of the determinant. In this way, GCT introduces the notions of orbits and borders into algebraic complexity theory. The introduction of these two notions not only opens up new directions of investigations for the old problems of lower bounds, PIT, and circuit reconstruction, but also introduces some interesting new problems. We shall now discuss two such problems, one related to orbits and another to borders.

#### 1.1.4.1 Polynomial Equivalence

Two $n$-variate polynomials $f, g$ are said to be equivalent, denoted by $f \sim g$, if $g \in \text{orb}(f)$. The Polynomial Equivalence or PE problem asks if the given polynomials $f, g$ are equivalent. Also, if $f \sim g$, then an algorithm for the PE problem should find an $A \in \text{GL}(n, \mathbb{F})$ and $\mathbf{b} \in \mathbb{F}^n$ such that $g = f(A\mathbf{x} + \mathbf{b})$. PE for a circuit class $\mathcal{C}$ can be defined by requiring that $f, g$ be computed by circuits in $\mathcal{C}$.

PE is a natural isomorphism-type problem for polynomials. However, despite decades of work, the exact complexity of the PE problems remains a mystery. Over finite fields, PE is known to be in $\text{NP} \cap \text{coAM}$ [Thi98, Sax06], but to date, no sub-exponential algorithm for this problem is known. Over $\mathbb{C}$ and $\mathbb{R}$, nothing better than a reduction to the problem of checking whether a system of polynomial equations has a solution is known. As a result, a special case of the PE problem called the Equivalence Test or ET problem has been studied. The ET problem is the version of the PE problem where $f$ is a fixed polynomial like the determinant or the permanent polynomial. One can also define the ET problem for a class $\mathcal{C}$ of circuits like ROFs by requiring $f$ to be some unknown polynomial computed by a circuit in $\mathcal{C}$. Then, an algorithm for the ET problem needs to determine if a given polynomial $g$ is equivalent to some circuit in $\mathcal{C}$. If yes, it must output a $\mathsf{C} \in \mathcal{C}$, $A \in \text{GL}(n, \mathbb{F})$ and $\mathbf{b} \in \mathbb{F}^n$ such that $g = \mathsf{C}(A\mathbf{x} + \mathbf{b})$. Notice that the ET problem for $\mathcal{C}$ is the circuit reconstruction problem

for the orbit of $\mathcal{C}$. As every formula is an affine projection of an ROF, ET for ROFs can be thought of as a natural special case of formula reconstruction.

**Prior work on PE and ET.** The PE problem for quadratic forms, also called the Quadratic Form Equivalence or QFE, has been long known to admit an efficient algorithm. However, even for cubic forms, PE is known to be a hard problem. More precisely, it is known that cubic form equivalence is at least as hard as the graph isomorphism problem [AS05]. However, no reduction from cubic form equivalence to graph isomorphism is known, so the former is possibly harder than graph isomorphism. The ET problem was first introduced in [Kay12a]. Since then, efficient equivalence testing algorithms have been developed for a host of polynomials like sum-product polynomials and elementary symmetric polynomials [Kay11], power symmetric polynomials [Kay11, KS21, KS23a], determinant [Kay12a, GGKS19], permanent [Kay12a, Gro12], continuant [MS21], IMM [KNST17, MNS20], and Nisan-Wigderson design polynomials [GS19, BDS24]. An equivalence test for the sums of univariates was given in [GKP18]. [BDSS24] showed that the equivalence test for sparse polynomials is NP-hard. Very recently, [BDGT24] have shown that the isomorphism testing problem for ROABPs is NP-hard. The isomorphism testing problem is the variant of the ET problem where $A$ is a permutation matrix.

#### 1.1.4.2 De-bordering

The de-bordering question for a circuit class $\mathcal{C}$ asks that if $f \in \overline{\mathcal{C}}$, then is $f$ also in $\mathcal{C}$? If not, then can we find a class $\mathcal{C}'$ such that $\overline{\mathcal{C}} \subseteq \mathcal{C}'$? It is known that if $f(x_1, \ldots, x_n) \in \overline{\mathcal{C}}$, then it can be *approximated* by circuits in $\mathcal{C}$ (see Section 2.10). Thus, the de-bordering question asks that if a polynomial can be approximated by circuits in a circuit class, can it also be computed exactly by some circuit in that class? This is not only a natural question, but the de-bordering problem for VP, i.e. $\mathsf{VP} \stackrel{?}{=} \overline{\mathsf{VP}}$ has the following connection to the $\mathsf{VP} \stackrel{?}{=} \mathsf{VNP}$ problem: Most known approaches for proving lower bounds not just prove lower bounds for a class, but also for the border of that class. Thus, if $\mathsf{VP} \neq \mathsf{VNP}$, but $\mathsf{VNP} \subseteq \overline{\mathsf{VP}}$, then all of the known approaches can never hope to separate VP and VNP. Another motivation for studying the de-bordering question comes from the problem of factoring polynomials given as arithmetic circuits. A conjecture about the efficiency of factoring algorithms called the factor conjecture says that if $f = g^e \cdot h$, then $g$ can be computed by a circuit of size $\mathrm{poly}(size(f), \deg(g))$, where $size(f)$ is the size of input circuit computing $f$. Burgisser showed in [Bür00] that if $f = g^e \cdot h$, then $g$ can be *approximated* by a circuit of size $\mathrm{poly}(size(f), \deg(g))$. So, if $\mathsf{VP} = \overline{\mathsf{VP}}$, then the factor conjecture would be true.

**Prior work on de-bordering.** It is folklore that the classes of sparse polynomials, ROFs, ROABPs, as well as non-commutative ABPs are closed under the border. The closure of the last two follows from the characterisation of ROABPs and non-commutative ABPs given by Nisan [Nis91]. It also follows from the duality trick [Sax08] that the border of depth-3 powering circuits is contained in the class of ROABPs. Also, the border of set-multilinear depth-3 circuits is contained in the class of ROABPs. [Kum20] showed that the border of $\Sigma\Pi\Sigma$ circuits with top fan-in 2 is universal, that is, it can compute all polynomials. [BIM+20] showed that the class of monotone ABPs is also closed under the border. [DDS21] proved that the border of depth-3 circuits with constant top fan-in is contained in VBP. They also gave a quasi-polynomial time hitting set construction for the border of this class. [DS22] proved that for every $k$, the border of $\Sigma\Pi\Sigma$ circuits with top fan-in $k$ is strictly contained in the border of $\Sigma\Pi\Sigma$ circuits with top fan-in $k+1$. [CGGR23] proved that the class of polynomials of the form $\det\left(\sum_{i\in[n]} A_i x_i\right)$ is closed under the border provided that all $A_i$ are rank-1. It was shown in [DGI+24] that any degree $d$ polynomial contained in the border of depth-3 powering circuits with top fan-in $k$ can be computed by a depth-3 powering circuit with top fan-in $4^k \cdot d$.

## 1.2 Our results

Having described the main directions of research in algebraic complexity theory, we now motivate and discuss our results. In Sections 1.2.1, 1.2.2, we discuss our results on the hitting sets and reconstruction problems related to orbits of constant read circuit classes. Section 1.2.4 has our results on the border of sums of ROFs. Our results on lower bounds for constant depth circuits can be found in Section 1.2.3.

### 1.2.1 Hitting sets for orbits of circuit classes

We give quasi-polynomial time hitting sets for orbits of low individual degree commutative ROABPs, multilinear constant width ROABPs, constant-depth constant-occur formulas, as well as occur-once formulas. These models have been formally defined in Section 2.1. As corollaries, we get quasi-polynomial time hitting sets for the orbits of various polynomials like the sum-product polynomials as well as the elementary and power symmetric polynomials. More importantly, we also get quasi-polynomial hitting sets for the orbits of $\{\mathsf{IMM}_{3,d}\}_{d\in\mathbb{N}}$ which is complete for arithmetic formulas. The contents of this section are from a joint work with Chandan Saha [ST24].

We have the following three reasons for studying hitting sets for orbits of ROABPs and

constant-occur formulas.

**The power of orbit closures.** Affine projections of polynomials computable by polynomial-size ROABPs or constant-occur formulas have great expressive power. For example, the iterated matrix multiplication polynomial $\mathsf{IMM}_{w,d}$ – the $(1,1)$-th entry of a product of $d$ generic $w \times w$ matrices – is computable by a linear-size ROABP, yet every polynomial computable by a size-$s$ *general* algebraic branching program[1] is in $\mathrm{aproj}(\mathsf{IMM}_{s,s})$. In fact, it was shown by Ben-Or and Cleve [BC92] that every polynomial computable by a size-$s$ arithmetic formula is in $\mathrm{aproj}(\mathsf{IMM}_{3,\mathrm{poly}(s)})$ . The sum-product polynomial $\mathsf{SP}_{s,d} := \sum_{i\in[s]} \prod_{j\in[d]} x_{i,j}$ is computable by a depth-2 read-once formula, yet every polynomial computable by a general depth-3 circuit with top fan-in $s$ and formal degree $d$ is in $\mathrm{aproj}(\mathsf{SP}_{s,d})$. As demonstrated by the depth reduction results in [GKKS16, Tav15, Koi12, AV08, VSBR83], depth-3 circuits are incredibly powerful. Also, affine projections of read-once formulas capture general arithmetic formulas. The orbit of $f$ being a mathematically interesting subset of $\mathrm{aproj}(f)$, it is natural to ask if we can give efficient hitting set constructions for the orbits of the above-mentioned polynomial families and circuit classes. Moreover, $\mathrm{orb}(f)$ is not 'much smaller' than $\mathrm{aproj}(f)$, as the latter is contained in the *orbit closure* of $f$ ($\overline{\mathrm{orb}(f)}$). The polynomials in $\overline{\mathrm{orb}(f)}$, and hence also the polynomials in $\mathrm{aproj}(f)$, can be approximated infinitesimally closely by the polynomials in $\mathrm{orb}(f)$ over $\mathbb{C}$.[2] In this sense, $\mathrm{orb}(f)$ is a dense subset of $\mathrm{aproj}(f)$.

**Geometry of the circuit classes.** Consider an $n$-variate polynomial $f \in \mathbb{R}[\mathbf{x}]$ that is computable by a polynomial-size ROABP or a polynomial-size constant-occur formula. Let $\mathbb{V}(f)$ be the variety (i.e., the zero locus) of $f$. The geometry of $\mathbb{V}(f)$ is preserved by any rigid transformation on $\mathbb{R}^n$. A rigid transformation $T$ is given by an orthogonal matrix $R \in O(n,\mathbb{R})$ (which stands for reflections and rotations) and a translation vector $\mathbf{b} \in \mathbb{R}^n$ such that every $\mathbf{x} \in \mathbb{R}^n$ maps to $T(\mathbf{x}) = R\mathbf{x} + \mathbf{b}$. Computation of a set $\mathcal{H} \subseteq \mathbb{R}^n$ that is not contained in $T(\mathbb{V}(f))$, for every rigid transformation $T$, would have to be "mindful" of the geometry of $\mathbb{V}(f)$ and oblivious to the choice of the coordinate system. Computing such an $\mathcal{H}$ is exactly the problem of constructing a hitting set for the polynomials $\{f(R\mathbf{x}+\mathbf{b}) \ : \ R \in O(n,\mathbb{R}) \text{ and } \mathbf{b} \in \mathbb{R}^n\}$. We can generalize the problem slightly by replacing $R \in O(n,\mathbb{R})$ with $A \in \mathrm{GL}(n,\mathbb{R})$.[3] A hitting set for ROABPs or constant-occur formulas

[1] Thanks to the depth reduction result of Valiant, Skyum, Berkowitz, and Rackoff [VSBR83], low-degree polynomials computable by arithmetic circuits are also computable by quasi-polynomially large algebraic branching programs.

[2] However, $\overline{\mathrm{orb}(f)}$ can be strictly larger than $\mathrm{aproj}(f)$.

[3] An invertible transformation $A$ is essentially an orthogonal transformation up to "scaling": from singular

does not immediately give a hitting set for $\{f(A\mathbf{x}+\mathbf{b}) : A \in \mathrm{GL}(n,\mathbb{R}) \text{ and } \mathbf{b} \in \mathbb{R}^n\}$, as the definitions of an ROABP and a constant-occur formula are tied to the choice of the coordinate system.

**Strengthening existing techniques.** Finally, it is worth investigating whether the techniques used to design hitting sets for ROABPs and constant-occur formulas can be applied or strengthened or combined to give hitting sets for the orbits of these circuit classes.

Indeed, the results of this section are obtained by building upon, strengthening and combining several tools and techniques from the literature, in particular the rank concentration by translation technique from [ASS13]; the merge-and-reduce idea from [FS13], [FSS14]; the algebraic independence based technique from [ASSS16, BMS13]; and the Shpilka-Volkovich or the $k$-wise independent generator from [SV15]. Our work here on hitting sets for the orbits of the above-mentioned circuit classes probes a line of research that – to our knowledge – has remained largely unexplored. In obtaining these results, we have highlighted the efficacy and the versatility of some of the existing tools and techniques. We now state our results.

**Theorem 1.1 (Hitting sets for the orbits of commutative ROABPs with low individual degree)** *Let $\mathcal{C}$ be the set of $n$-variate polynomials with individual degree at most $d$ that are computable by width-$w$ commutative ROABPs. If $|\mathbb{F}| > n^2d$, then a hitting set for $\mathrm{orb}(\mathcal{C})$ can be computed in $(nd)^{O(d\log w)}$ time.*

An interesting subclass of commutative ROABPs is the class of *sums of products of univariates*. This model, which is a broad generalization of the class of sparse polynomials, has found important applications in several other works [Sax08, SSS13, GKKS16]. We say an $n$-variate polynomial $f(x_1, x_2, \ldots, x_n)$ can be expressed as a sum of $s$ products of univariates if $f = \sum_{i\in[s]} \prod_{j\in[n]} f_{i,j}(x_j)$, where each $f_{i,j}(x_j)$ is a univariate polynomial in $x_j$. Theorem 1.2 below (which follows as a corollary from the above theorem) gives a quasi-polynomial time hitting set for the orbits of sums of products of *low degree* univariates.

**Theorem 1.2 (Hitting sets for the orbits of sums of products of low degree univariates)** *Let $\mathcal{C}$ be the set of $n$-variate polynomials that can be expressed as sums of $s$ products of univariates of degree at most $d$. If $|\mathbb{F}| > n^2d$, then a hitting set for $\mathrm{orb}(\mathcal{C})$ can be computed in $(nd)^{O(d\log s)}$ time.*

*Remarks.*

value decomposition, we have $A = UDV$, where $U, V$ are orthogonal matrices and $D$ is a diagonal matrix.

1. Even under the low individual degree restriction the above class remains reasonably natural and interesting. For example, the elementary symmetric polynomial $\mathsf{ESym}_{n,D} = \sum_{S \in \binom{[n]}{D}} \prod_{i \in S} x_i$ can be expressed as a sum of $n+1$ products of univariate affine forms. This is due to a nice interpolation trick attributed to Ben-Or in [NW97, Shp02]. The theorem then implies an $n^{O(\log n)}$-time hitting set for $\mathrm{orb}(\mathsf{ESym}_{n,D})$.

2. The theorem also implies a quasi-polynomial time hitting set for the orbits of multilinear sparse polynomials, and more generally, for the orbits of sparse polynomials with low individual degree. It is easy to see that the orbit of a multilinear sparse polynomial may contain a non-sparse polynomial. So, the existing hitting set constructions for sparse polynomials by Klivans and Spielman [KS01], Lipton and Vishnoi [LV03] (where the complexity depends polynomially on the sparsity parameter) may no longer remain efficient for the orbits of sparse polynomials.

3. It turns out though that for the particular case of sparse polynomials it is possible to remove the individual degree restriction from the above theorem. This is due to an independent and simultaneous work by Medini and Shpilka [MS21].

**Theorem 1.3 (Hitting sets for the orbits of multilinear constant-width ROABPs)** *Let $\mathcal{C}$ be the set of $n$-variate multilinear polynomials that are computable by width-$w$ ROABPs. If $|\mathbb{F}| > n^{O(w^4)}$, then a hitting set for $\mathrm{orb}(\mathcal{C})$ can be computed in $n^{O(w^6 \cdot \log n)}$ time.*

*Remarks.*

1. The theorem gives a quasi-polynomial time hitting set for $\mathrm{orb}(\mathsf{IMM}_{3,d})$, as $\mathsf{IMM}_{3,d}$ is computable by a width-9 ROABP. As mentioned before, the family $\{\mathsf{IMM}_{3,d}\}_{d \in \mathbb{N}}$ is complete for the class of arithmetic formulas under affine projections (in fact, under $p$-projections), as shown by Ben-Or and Cleve [BC92].

2. The set of affine projections of $\mathsf{IMM}_{2,d}$ is also quite rich, despite the fact that there are simple quadratic polynomials that are not in $\mathrm{aproj}(\mathsf{IMM}_{2,d})$ for *any* $d$, as shown by Allender and Wang [AW16], Saha, Saptharishi, and Saxena [SSS09]. This is because Saha, Saptharishi, and Saxena [SSS09] proved that hitting sets for $\mathrm{aproj}(\mathsf{IMM}_{2,d})$ give hitting sets for depth-3 circuits. Moreover, Bringmann, Ikenmeyer, and Zuiddam [BIZ18] have shown that $\overline{\mathrm{orb}(\mathsf{IMM}_{2,d})}$ captures the orbit closures of arithmetic formulas. The above theorem implies a quasi-polynomial time hitting set for $\mathrm{orb}(\mathsf{IMM}_{2,d})$.

**Theorem 1.4 (Hitting sets for the orbits of constant-depth, constant-occur formulas)** *Let $\mathcal{C}$ be the set of $n$-variate, degree-$D$ polynomials that are computable by depth-$\Delta$, occur-$k$ formulas of size $s$. Let $R := (2k)^{2\Delta \cdot 2^{\Delta}}$. If $\mathrm{char}(\mathbb{F}) = 0$ or $> (2ks)^{\Delta^3 R}$, then a hitting set for $\mathrm{orb}(\mathcal{C})$ can be computed in $(nRD)^{O(R(\log R + \Delta \log k + \Delta \log s) + \Delta R)}$ time. If the leaves are labelled by $b$-variate polynomials, then a hitting set for $\mathrm{orb}(\mathcal{C})$ can be computed in $(nRD)^{O(Rb + \Delta R)}$ time. In particular, if $\Delta$ and $k$ are constants, then the hitting sets can be constructed in time $(nD)^{O(\log s)}$ and $(nD)^{O(b)}$, respectively.*

*Remarks.*

1. The above theorem gives hitting sets for the orbits of two other interesting models that have been studied in the literature: There is a polynomial-time constructible hitting set for multilinear depth-4 circuits with constant top fan-in [SV18, KMSV13]. Theorem 1.4 implies a quasi-polynomial time hitting set for the orbit of this model, as a multilinear depth-4 circuit with constant top fan-in can be viewed as a depth-4 constant-occur formula. [BMS13] gave a polynomial-time hitting set for $\mathsf{C}(f_1, \ldots, f_m)$, where $\mathsf{C}$ is a low-degree circuit and $f_1, \ldots, f_m$ are sparse polynomials with bounded transcendence degree. The proof of the above theorem also implies a quasi-polynomial time hitting set for the orbit of this model.

2. The theorem yields polynomial-time hitting sets for the orbits of the power symmetric polynomial $\mathsf{PSym}_{n,D} = \sum_{i \in [n]} x_i^D$ and the sum-product polynomial $\mathsf{SP}_{n,D} = \sum_{i \in [n]} \prod_{j \in [D]} x_{i,j}$. This is because the polynomials PSym and SP are computable by constant-depth, occur-once formulas whose leaves are labelled by univariate polynomials. Prior to our work, [KS21] gave a polynomial-time hitting set for $\mathrm{orb}(\mathsf{PSym}_{n,D})$ using a different argument that involves the Hessian matrix.

**Theorem 1.5 (Hitting sets for the orbits of occur-once formulas)** *Let $\mathcal{C}$ be the set of $n$-variate, degree-$D$ polynomials that are computable by occur-once formulas whose leaves are labelled by $s$-sparse polynomials. If $|\mathbb{F}| > nD$ and $\mathrm{char}(\mathbb{F}) = 0$ or $> D$, then a hitting set for $\mathrm{orb}(\mathcal{C})$ can be computed in $(nD)^{O(\log n + \log s)}$ time. If the leaves are labelled by $b$-variate polynomials, then a hitting set for $\mathrm{orb}(\mathcal{C})$ can be computed in $(nD)^{O(\log n + b)}$ time.*

*Remark.* The independent and concurrent work by Medini and Shpilka [MS21] gave (among other results) a quasi-polynomial time hitting set construction for the orbits of read-once formulas. We note that this result also follows from the second part of the above theorem which is already present in the original version of our work [ST24].

**Simultaneous and subsequent work:** Independent of our work here, Medini and Shpilka [MS21] gave quasi-polynomial time hitting sets for the orbits of sparse polynomials and read-once formulas. In a subsequent work, Bhargava and Ghosh [BG21] improved some of the results of this work. They improved our $(nd)^{O(d \log w)}$ time hitting set for the orbits of width-$w$, individual degree $d$, commutative ROABPs to an $(ndw)^{O(\min\{w^2, 2d \log w\})}$ time hitting set for the orbits of width-$w$, individual degree $d$, any-order ROABPs. Any-order ROABPs are a slight generalisation of commutative ROABPs. Observe that when $w$ is a constant, [BG21] gives a hitting set even when the individual degree is large. They also improve our $(nd)^{O(w^6 \log n)}$ time hitting set for orbits of multilinear, width-$w$ ROABPs to an $(ndw)^{O(w^2 \log n \cdot \min\{w^2, 2d \log w\})}$ time hitting set for orbits of individual degree $d$, width-$w$ ROABPs.

### 1.2.2 Equivalence test for read-once arithmetic formulas

Here, our main result is a randomised polynomial time equivalence test for Read-Once Arithmetic Formulas (ROFs). The contents of this section are from a joint work with Nikhil Gupta and Chandan Saha [GST20]. Before stating our result, we give some motivation for studying this problem.

**Generalizing quadratic form equivalence.** PE for the class of quadratic forms or homogeneous polynomials of degree 2 is known. Are there bigger classes of polynomials for which PE is easy? An obvious way to generalize quadratic form equivalence is to solve PE for higher degree forms. Unfortunately, even cubic form equivalence is at least as hard as graph isomorphism and possibly harder. Another natural way to generalize quadratic form equivalence is as follows: Over $\mathbb{C}$, an $n$-variate quadratic form with *no redundant variables*[1] is in the orbit of $x_1x_2 + x_3x_4 + \ldots + x_{n-1}x_n$, if $n$ is even. The expression $x_1x_2 + x_3x_4 + \ldots + x_{n-1}x_n$ is a read-once arithmetic formula (ROF). The quadratic form equivalence problem (QFE) over $\mathbb{C}$ can thus be viewed as PE for orbits of *quadratic* ROFs. Thus, the following is a natural question. Can we solve PE for orbits of general ROFs efficiently? A typical algorithm to solve the search version of QFE over $\mathbb{C}$ finds invertible linear transformations that map the two input quadratic forms to the *canonical* ROF $x_1x_2 + x_3x_4 + \ldots + x_{n-1}x_n$ (see Section 2.6 for a definition of a canonical ROF). In other words, such an algorithm solves the ET problem for quadratic ROFs. So it is natural to ask if we can solve the ET problem for general ROFs efficiently.

[1] i.e., the number of variables cannot be reduced by applying an invertible linear map on the variables.

**Learning random or non-degenerate formulas** As mentioned before, a formula is an affine projection of an ROF. Learning formulas in the *worst-case* is a potentially hard problem, however, it may be possible to learn formulas in the average-case by formulating natural distributions under which formulas are learnable. A natural distribution on formulas is defined as follows: pick a tree of size $s$ *arbitrarily*, label the internal nodes by $+$ and $\times$ operations to form alternating layers of $+$ and $\times$ gates, and label the leaves by *random* linear forms in $n$ variables. The corresponding learning problem asks to reconstruct a random formula – picked according to this distribution – from black-box access to the formula. This problem was studied in [GKQ13] by *fixing* the underlying tree to be a complete binary tree; the formulas we thus get are called formulas in *alternating normal form* (ANF). [GKQ13] gave an efficient learning algorithm for random ANFs. On the other hand, an ET for ROFs gives a learning algorithm for random formulas, *irrespective of the underlying tree*, provided $n \geq s$. This is because a random formula is in the orbit of an ROF with high probability if $n \geq s$ and $|\mathbb{F}|$ is sufficiently large. Thus, ET for ROFs provides supporting evidence for efficient learnability of random formulas (that are not necessarily ANFs).

Having motivated the equivalence test problem for ROFs, we now state our results. Our results hold over any field $\mathbb{F}$ of characteristic 0 or of sufficiently large characteristic and size. As for the computation model, we assume that it allows basic field operations in unit time and univariate polynomial factoring in randomized polynomial time. We say an algorithm is efficient if it runs in randomized polynomial time.

For the ease of stating the theorems, we consider ROFs in *canonical* form (see Definition 2.23). The orbit of every ROF contains a canonical ROF. So, by removing redundant variables from the input polynomial we can assume *without any loss of generality* that the underlying ROF is canonical.

Our first result gives an efficient algorithm to solve ET for *general* ROFs. The algorithm is randomized and has oracle access to the search version of QFE. In subsequent discussions, we will mention "QFE" to mean "the search version of QFE". We will also identify an ROF with the polynomial it computes and denote the set of $n \times n$ matrices with entries in $\mathbb{F}$ by $M(n, \mathbb{F})$.

**Theorem 1.6 (ET for ROFs)** *Let $n \in \mathbb{N}$, $\mathrm{char}(\mathbb{F}) = 0$ or $\geq n^2$, and $|\mathbb{F}| \geq n^{13}$. There is a $\mathrm{poly}(n)$ time randomized algorithm (with oracle access to QFE over $\mathbb{F}$) that takes input black-box access to an $n$-variate polynomial $f \in \mathbb{F}[\mathbf{x}]$, which is in the orbit of an unknown canonical ROF $\mathtt{C}$, and outputs (with high probability) an $A \in \mathrm{GL}(n, \mathbb{F})$ such that $f(A\mathbf{x}) = \mathtt{C}(PS\mathbf{x} + \mathbf{b})$, where*

*$P \in M(n, \mathbb{F})$ and $S \in M(n, \mathbb{F})$ are permutation and scaling (i.e., diagonal) matrices respectively, and $\mathbf{b} \in \mathbb{F}^n$.*

*Remarks.*

1. As $\mathtt{C}(PS\mathbf{x} + \mathbf{b})$ is an ROF, we can apply any of the known polynomial-time ROF reconstruction algorithms [HH91, BHH95, SV14, MV18] to first get an ROF for $\mathtt{C}(PS\mathbf{x} + \mathbf{b})$, and then obtain a formula for $f$ by applying $A^{-1}$ on the variables of the reconstructed ROF.

2. QFE can be solved efficiently over $\mathbb{C}, \mathbb{R}, \mathbb{F}_q$ and also over $\mathbb{Q}$ with oracle access to integer factoring. Hence, ET for ROFs can also be solved efficiently over these fields.

3. Although ET has been studied for polynomial families like the determinant and IMM, to our knowledge no ET was known for any natural circuit class of unbounded *depth*, *degree* and *fan-in* (or even depth-4 ROFs) before this work.

4. Recently, [MS21] showed that ET for ROANFs and sum-product polynomials can be solved efficiently. As ROANFs are special fan-in 2 ROFs and sum-product polynomials are depth-2 ROFs, the theorem generalizes these two results considerably. Also, our proof approach is entirely different from the ones in [MS21].

5. The constraints on $\mathrm{char}(\mathbb{F})$ and $|\mathbb{F}|$ originate primarily (but not solely) from the use of the black-box multivariate polynomial factorization algorithm [KT90] in the equivalence test. We have not made an attempt to optimize these constraints.

The second result gives an efficient algorithm to solve PE for orbits of ROFs that are *additive-constant-free*. An ROF is additive-constant-free if no $\mathbb{F}$-constant appears as a child of a $+$-gate. For e.g., the canonical ROF $x_1x_2 + x_3x_4 + \ldots + x_{n-1}x_n$ is additive-constant-free. ROANFs and sum-product polynomials are also examples of additive-constant-free canonical ROFs. An additive-constant-free ROF is in the orbit of an additive-constant-free canonical ROF.

**Theorem 1.7 (PE for orbits of additive-constant-free ROFs)** *Let $n \in \mathbb{N}$, $\mathrm{char}(\mathbb{F}) = 0$ or $\geq n^2$, and $|\mathbb{F}| \geq n^{13}$. There is a $\mathrm{poly}(n)$ time randomized algorithm (with oracle access to QFE over $\mathbb{F}$) that takes input black-box access to two $n$-variate polynomials $f_1, f_2 \in \mathbb{F}[\mathbf{x}]$, which are in the orbits of two <u>unknown</u> additive-constant-free canonical ROFs, and checks if $f_1 \in \mathrm{orb}(f_2)$. Furthermore, if $f_1 \in \mathrm{orb}(f_2)$, then the algorithm outputs (with high probability) an $A \in \mathrm{GL}(n, \mathbb{F})$ and a $\mathbf{b} \in \mathbb{F}^n$ such that $f_1 = f_2(A\mathbf{x} + \mathbf{b})$.*

As mentioned before, the above result is a broad generalization of efficient QFE. We also show that the additive-constant-free restriction can be removed for the special case of the orbits of ROFs with product-depth 2.

**Theorem 1.8 (PE for orbits of product-depth 2 ROFs)** *Let $n \in \mathbb{N}$, $\mathrm{char}(\mathbb{F}) = 0$ or $\geq n^2$, and $|\mathbb{F}| \geq n^{13}$. There is a $\mathrm{poly}(n)$ time randomized algorithm (with oracle access to QFE over $\mathbb{F}$) that takes input black-box access to two $n$-variate polynomials $f_1, f_2 \in \mathbb{F}[\mathbf{x}]$, which are in the orbits of two unknown canonical ROFs with product-depth 2, and checks if $f_1 \in \mathrm{orb}(f_2)$. Furthermore, if $f_1 \in \mathrm{orb}(f_2)$, then the algorithm outputs (with high probability) an $A \in \mathrm{GL}(n, \mathbb{F})$ and a $\mathbf{b} \in \mathbb{F}^n$ such that $f_1 = f_2(A\mathbf{x} + \mathbf{b})$.*

### 1.2.3 Lower bounds for constant depth arithmetic circuits

In a recent breakthrough work, [LST21] proved super-polynomial lower bounds for constant-depth arithmetic formulas.[1] In this section, we give an alternative and more "direct" proof of their result. The contents of this section are from a joint work with Prashanth Amireddy, Ankit Garg, Neeraj Kayal, and Chandan Saha [AGK+23].

The lower bounds of [LST21] are of the form $n^{\Omega(\log(n)^{c_\Delta})}$ for a constant $0 < c_\Delta < 1$ depending on the depth $\Delta$ of the formula. We examine the lower bound proof in [LST21] at a high level. Their proof has two main steps: First, they reduce the problem of proving lower bounds for low-depth formulas to the problem of proving lower bounds for low-depth *set-multilinear* formulas; set-multilinear formulas are special homogeneous formulas with an underlying partition of the variables into subsets. [LST21] calls such reductions 'hardness escalation'. Second, they use an interesting adaptation of the rank of the partial derivatives matrix measure [Nis91] to prove a lower bound for low-depth set-multilinear formulas. They call this measure *relative rank* (relrk). The effectiveness of the relrk measure crucially depends on a certain 'imbalance' between the sizes of the sets used to define set-multilinear polynomials. The proof in [LST21] raises two natural questions:

**Question 1:** Can we bypass the hardness escalation, i.e., the set-multilinearization, step?

**Question 2:** Can we design a measure that exploits some weakness of homogeneous (but not necessarily set-multilinear) formulas directly?

[1] It is not difficult to see that if a polynomial has a size $s$, depth-$\Delta$ circuit, then it has a size $s^{O(\Delta)}$, depth-$\Delta$ formula. Thus, for $\Delta = O(1)$, the formula size and circuit size are polynomially related. Because of this reason, we can just work with constant depth formulas.

We answer both these question in affirmative by giving a direct lower bound for low-depth homogeneous formulas via the SP measure which was used in the series of works on homogeneous depth-4 exponential lower bounds. We also show that the lower bound can be obtianed using the affine projections of partials (APP) measure which has been used to prove lower bounds for of depth 4 circuits with bounded bottom fan-in [GKS20] . While our proof also yields lower bounds only in the low-degree setting, the hope is that it could potentially lead to a stronger lower bound in the future. We now state our result.

Consider the *shifted partials* (SP) and *affine projection of partials* (APP) measures for a polynomial $f$: $\mathsf{SP}_{k,\ell}(f) := \dim\langle \mathbf{x}^{\ell} \cdot \partial^{k}(f)\rangle$; $\mathsf{APP}_{k,n_0}(f) := \max_{L:\mathbf{x}\to\langle \mathbf{z}\rangle} \dim \left\langle \pi_L\left(\partial^{k} P\right)\right\rangle$, where $L : \mathbf{x} \to \langle \mathbf{z}\rangle$, $|\mathbf{z}| = n_0$, are linear maps and for any $g \in \mathbb{F}[\mathbf{x}]$, $\pi_L(g) := g(L(x_1), \ldots, L(x_n))$.

Also, for convenience, let us denote by $M(n,k) := \binom{n+k-1}{k}$ the number of monomials of degree $k$ in $n$ variables.

**Theorem 1.9 (Lower bound for low-depth homogeneous formulas via shifted partials)** *Let $\mathsf{C}$ be a homogeneous formula of size $s$ and product-depth $\Delta$ that computes a polynomial of degree $d$ in $n$ variables. Then for appropriate values of $k$ and $\ell$,*

$$\mathsf{SP}_{k,\ell}(\mathsf{C}) \leq \frac{s\, 2^{O(d)}}{n^{\Omega(d^{2^{1-\Delta}})}} \min\{M(n,k)M(n,\ell), M(n,d-k+\ell)\}.$$

*At the same time, there are homogeneous polynomials $f$ of degree $d$ in $n$ variables (e.g., an appropriate projection of iterated matrix multiplication polynomial, Nisan-Wigderson design polynomial, etc.) such that*

$$\mathsf{SP}_{k,\ell}(f) \geq 2^{-O(d)} \min\{M(n,k)M(n,\ell), M(n,d-k+\ell)\}.$$

*This gives a lower bound of $\frac{n^{\Omega(d^{2^{1-\Delta}})}}{2^{O(d)}}$ on the size of homogeneous product-depth $\Delta$ formulas for $f$.*

**Remark 1.1**

1. *The above lower bound is slightly better than the bound of [LST21]. Instead of the $d^{O(d)}$ loss incurred due to converting homogeneous to set-multilinear formulas, our analysis incurs a $2^{O(d)}$ loss; in fact, this loss can be brought down to $2^{O(k)}$, but we ignore this distinction as we set $k = \Theta(d)$ in the analysis. So, for example, for homogeneous product-depth 2 formulas, our super-polynomial lower bound continues to hold for a higher degree ($\log^2(n)$ vs $(\log(n)/\log\log(n))^2$ in [LST21]). While the improvement may be insignificant, this hints at something interesting going on with the direct approach.*

2. *[HLT24] uses the direct approach proposed in this work to strengthen some results about the ideal proof system (see [GP18]) proved in [GHT22].*

### 1.2.4 Border of the sums of two ROFs

The main result of this section is the de-bordering of the sum of two additive constant free ROFs. We also give a quasi-polynomial time hitting set construction for the border of sums of two homogeneous depth-5 ROFs. While ROF and sums of ROFs are simple circuit classes, studying the PIT and learning questions for these classes led to the development of techniques that turned out to be useful for other, more general classes of circuits. Thus one can certainly hope that studying the border of these classes would be a first step towards studying the borders of other more general circuit classes like sums of constantly many ROABPs and depth-4 multilinear circuits. It is not too difficult to see that ROFs are closed under the border. So the next natural step is to try and understand the power of the border of a sum of constantly many ROFs. Moreover, a lot of known de-bordering results are for some constant-depth circuit classes. The sum of constantly many ROFs is a natural model to explore de-bordering for circuits with larger depths.

Our first result shows that the class of sum of $k$ many ROFs, denoted by $\Sigma_k$ROF is not closed under the border. We use $\mathrm{ROF}(n)$ to denote the class of ROFs in $n$-variables.

**Theorem 1.10 (Sum of ROFs not closed under border)** *For any $n \in \mathbb{N}$, $n \geq 10$ and $2 \leq k \leq \frac{n}{5}$, $\sum_k \mathrm{ROF}(n) \subsetneq \overline{\sum_k \mathrm{ROF}(n)}$ over any field.*

Next we show that the sum of two additive constant free ROFs is contained in the sum of linearly many ROFs. We shall denote the class of additive constant free ROFs by $\mathrm{ROF}_0$.

**Theorem 1.11 (De-bordering the border of sum of 2 ROFs)** *Over fields of characteristic other than 2 and for any $n \in \mathbb{N}$, $\overline{\sum_2 \mathrm{ROF}_0(n)} \subseteq \sum_{O(n)} \mathrm{ROF}_0(n)$.*

Finally, we give a quasi-polynomial time hitting set for the border of sums of two homogeneous depth 5 ROFs.

**Theorem 1.12 (Hititng set for the border of sum of 2 homogeneous depth-5 ROFs)** *If $\mathbb{F}$ is a field of size $\mathrm{poly}(n)$ and characteristic other than 2, then there is an $n^{O(\log n)}$ time computable hitting set for the border of sums of two homogeneous depth-5 ROFs computing $n$-variate polynomials over $\mathbb{F}$.*

## 1.3 Organisation of the thesis

We first give all the definitions and some preliminary results required for this thesis in Chapter 2. The next two chapters are devoted to proving the results from Section 1.2.1. In Chapter

3 we prove Theorems 1.1, 1.2, and 1.3. Chapter 4 is devoted to the proofs of Theorems 1.4 and 1.5. Chapter 5 gives a proof of Theorem 1.6 while Chapter 6 gives a proof of Theorems 1.7 and 1.8. A proof of Theorem 1.9 can be found in Chapter 7. Theorems 1.10, 1.11, and 1.12 are proved in Chapter 8. Finally, we conclude by mentioning some directions for future work in Chapter 9.

# Chapter 2

# Preliminaries

This chapter describes the notations and conventions used throughout this thesis. It defines notations required to describe the results of this thesis as well as states some preliminary results used in the proofs of the main results of this work.

We begin by stating some of the notations that will be used throughout this work. Notations and conventions which are specific to a chapter will be defined at the beginning of that chapter.

$\mathbb{N}$ denotes the set of natural numbers. For $n \in \mathbb{N}$, $[n] = \{1, \ldots, n\}$ and $\mathbf{x} = \{x_1, \ldots, x_n\}$. We shall use symbols like $\mathbb{F}, \mathbb{K}$ to denote a field. For a polynomial $f \in \mathbb{F}[\mathbf{x}]$, $\mathrm{var}(f)$ shall denote the set of variables present in $f$. The space of polynomials having degree at most $d$ will be denoted as $\mathbb{F}[\mathbf{x}]_{\leq d}$. For $S \subseteq \mathbb{F}[\mathbf{x}]$, $\langle S \rangle$ is the $\mathbb{F}$-linear space spanned by $S$. $\mathrm{GL}(n, \mathbb{F})$ is the set of $n \times n$ invertible matrices over $\mathbb{F}$.

## 2.1 Arithmetic models of computation

We start by defining an arithmetic circuit.

**Definition 2.1 (Arithmetic circuit)** *An arithmetic circuit $\mathsf{C}$ over a field $\mathbb{F}$ and a set of variables $\mathbf{x} = (x_1, ..., x_n)$ is a directed acyclic graph. The vertices of $\mathsf{C}$ are called gates. Each gate with in-degree 0 is called an input gate and is labelled by either a variable or a field element. Every other gate is either labelled by a $\times$ (called a product gate) or a $+$ (called a sum gate). Every edge is labelled by a field element and every gate with out-degree 0 is called an output gate. An arithmetic circuit computes a polynomial in the natural way: an input gate computes the field element or variable it is labelled with. A sum gate computes the sum of polynomials computed by its inputs, each input scaled by the field element on the corresponding edge. Similarly, a product gate computes the product*

*of polynomials computed by its inputs, each input scaled by the field constant on the corresponding edge.*

*The size of an arithmetic circuit is equal to the number of edges in it and the depth of an arithmetic circuit is equal to the length of the longest directed path in it. The product-depth of a circuit is equal to the number of product gates on the longest directed path in it.*

The fan-in of a gate is equal to the number of edges entering the gate and the fan-out of a gate is the number of edges leaving the gate. An arithmetic formula is a special type of arithmetic circuit.

**Definition 2.2 (Arithmetic formula)** *An arithmetic formula is an arithmetic circuit whose underlying directed graph is a tree.*

Algebraic Branching Programs or ABPs are a model of computation that is known to be at least as powerful as arithmetic formulas.

**Definition 2.3 (Algebraic branching program)** *Let $M_1, \ldots, M_k$ be matrices whose entries are affine forms in $x_1, \ldots, x_n$ such that $M_2, \ldots, M_{k-1}$ are $w \times w$ matrices and $M_1, M_k$ are $1 \times w$ and $w \times 1$ dimensional vectors. Then, $M_1 \cdots M_k$ is an algebraic branching program (ABP). $w$ is called the width of the ABP. The size of the ABP is defined to be the number of entries in $M_1, \ldots, M_k$, that is, $(k-2)w^2 + 2w$.*

Some of the chapters of this work consider an extremely restricted class of arithmetic formulas called Read-Once Formulas or ROFs.

**Definition 2.4 (Read-Once Formula)** *An arithmetic formula $\mathsf{C}$ over a field $\mathbb{F}$ is a* read-once formula (ROF) *if every leaf in $\mathsf{C}$ is labelled by either a distinct variable or an $\mathbb{F}$ element.*

We describe ROFs and some of the facts about them in more detail in Section 2.6. A natural way to generalise ROFs is to allow variables to label constantly many leaves; such circuits are called read-$k$ formulas, and more generally occur-$k$ formulas as defined in [ASSS16].

**Definition 2.5 (Occur-$k$ formula)** *An occur-k formula is a rooted tree whose leaves are labelled by s-sparse polynomials and whose internal nodes are sum ($+$) gates or product-power ($\times^{\wedge}$) gates. Each variable appears in at most k of the sparse polynomials that label the leaves. The edges feeding into a $+$ gate are labelled by field elements and have 1 as* edge weights*, whereas the edges feeding into a $\times^{\wedge}$ gate have natural numbers as edge weights. A leaf node computes the s-sparse polynomial that labels it. A $+$ gate with inputs from nodes that compute $f_1, ..., f_m$ and with the corresponding input*

*edge labels $\alpha_1, ..., \alpha_m$, computes $\alpha_1 f_1 + \cdots + \alpha_m f_m$. A $\times$ gate with inputs from nodes that compute $f_1, ..., f_m$ and with the corresponding input edge weights $e_1, ..., e_m$, computes $f_1^{e_1} \cdots f_m^{e_m}$. The formula computes the polynomial that is computed by the root node.*

*The* size *of an occur-k formula is the weighted sum of all the edges in the formula (i.e., an edge feeding into a $\times$ gate is counted as many times as its edge weight, whereas an edge feeding into a $+$ gate is counted once) plus the sizes of the depth-2 circuits computing the s-sparse polynomials at the leaves. The* depth *of an occur-k formula is equal to the depth of the underlying tree plus 2, to account for the depth of the circuits computing the sparse polynomials at the leaves.*[1]

Read-$k$ formulas have been studied intensely in the literature. Occur-$k$ formulas generalize read-$k$ formulas in two ways – the leaves are labelled by arbitrary sparse polynomials instead of just variables, and powering gates are included along with the usual sum and product gates. These generalizations help make the occur-$k$ model complete[2], and capture other interesting circuit classes (such as multilinear depth-4 circuits with constant top fan-in [SV18, KMSV13]) and polynomial families (such as the power symmetric polynomials). Besides, there is no restriction of multilinearity on the model, unlike the case in some prior works [AvMV15, SV18, KMSV13].

Another natural generalisation of ROFs are Read-Once Algebraic Branching Programs or ROABPs (see [FS13]). They are algebraic analogues of Read-Once Branching Programs.

**Definition 2.6 (ROABP)** *An $n$-variate, width-$w$ read-once oblivious algebraic branching program (ROABP) is a product of the form $\mathbf{1}^T \cdot M_1(x_1)M_2(x_2)\cdots M_n(x_n) \cdot \mathbf{1}$, where $\mathbf{1}$ is the $w \times 1$ column vector of all ones, and for every $i \in [n]$, $M_i(x_i)$ is a $w \times w$ matrix whose entries are in $\mathbb{F}[x_i]$.*

In Chapter 3, we study a special class of ROABPs called commutative ROABPs.

**Definition 2.7 (Commutative ROABP)** *An $n$-variate, width-$w$* commutative *ROABP is an $n$-variate, width-$w$ ROABP $\mathbf{1}^T \cdot M_1(x_1)M_2(x_2)\cdots M_n(x_n) \cdot \mathbf{1}$, where for all $i, j \in [n]$, $M_i(x_i)$ and $M_j(x_j)$ commute with each other.*

A polynomial $f$ is *$s$-sparse* if it has at most $s$ monomials with non-zero coefficients; these monomials will be referred to as the *monomials of $f$*. It is easy to see that an $s$-sparse polynomial of degree $d$ can be computed by a depth-2 circuit of size at most $sd$. Also, observe that every $s$-sparse polynomial can be computed by a width-$s$ commutative ROABP.

[1] Observe that if $f$ is computable by a size-$s$, depth-$\Delta$, occur-$k$ formula, then it is also computable by a size-$s$, depth-$\Delta$ circuit that has only $+$ and $\times$ gates.

[2] For example, the power symmetric polynomial $x_1^n + \ldots + x_n^n$ cannot be computed by a read-$k$ formula for any $k < n$, but it can be computed by an occur-once formula.

## 2.2 Algebraic complexity classes

In his seminal work [Val79], Valiant defined the following two complexity classes for families of algebraic circuits. They can be thought of as the algebraic versions of P and NP.

**Definition 2.8 (Class VP)** $\mathsf{VP}_{\mathbb{F}}$ *is a class of all polynomial families* $\{f_n\}_{n\geq 1}$ *over a field* $\mathbb{F}$ *such that there exists a polynomial function* $t : \mathbb{N} \to \mathbb{N}$*, such that for all* $n \geq 1$*,* $f_n$ *is a polynomial in at most* $t(n)$ *variables, of degree at most* $t(n)$ *and computed by an arithmetic circuit of size at most* $t(n)$*.*

**Definition 2.9 (Class VNP)** $\mathsf{VNP}_{\mathbb{F}}$ *is a class of all polynomial families* $\{f_n\}_{n\geq 1}$ *over a field* $\mathbb{F}$ *such that there exist polynomial functions* $k, t : \mathbb{N} \to \mathbb{N}$ *and a family of polynomials* $\{g_n\}_{n\geq 1} \in \mathsf{VP}_{\mathbb{F}}$ *such that for all* $n \geq 1$*,*

$$f_n(x_1, ..., x_{k(n)}) = \sum_{w \in \{0,1\}^{t(n)}} g_{t(n)}(x_1, ..., x_{k(n)}, w_1, ..., w_{t(n)}).$$

Observe that $\mathsf{VP} \subseteq \mathsf{VNP}$, Valiant conjuctured that VP is in fact strictly contained in VNP. Proving Valiant's conjecture is the central open problem of algebraic complexity theory. Some of the polynomial families known to be in VP are the power and elementary symmetric polynomials, the iterated matrix multiplication polynomials and the determinant polynomial. The most well-known example of a polynomial family in VNP is the permanent polynomial family. In fact, the permanent polynomial family is known to be complete for VNP.

**Definition 2.10 (VNP completeness)** *A polynomial* $g(x_1 \ldots, x_m)$ *is said to be a p-projection of a polynomial* $f(y_1, \ldots, y_n)$ *if there exist* $r_1, \ldots, r_n \in \{x_1, \ldots, x_m\} \cap \mathbb{F}$ *such that* $g = f(r_1, \ldots, r_n)$*. A polynomial family* $\{f_n\}_{n\in\mathbb{N}} \in \mathsf{VNP}$ *is said to be* VNP*-complete if for every polynomial family* $\{g_n\}_{n\in\mathbb{N}} \in \mathsf{VNP}$*, there exists a polynomial function* $t : \mathbb{N} \to \mathbb{N}$ *such that* $g_n$ *is a p-projection of* $f_{t(n)}$*.*

It is not difficult to see that to prove $\mathsf{VP} \neq \mathsf{VNP}$, it is sufficient to show that there exists a VNP-complete polynomial family that is not in VP. In this way, VNP-completeness is analogous to NP-completeness.

A class related to VP is VBP.

**Definition 2.11 (Class VBP)** $\mathsf{VBP}_{\mathbb{F}}$ *is a class of all polynomial families* $\{f_n\}_{n\geq 1}$ *over a field* $\mathbb{F}$ *such that there exists a polynomial function* $t : \mathbb{N} \to \mathbb{N}$*, such that for all* $n \geq 1$*,* $f_n$ *is a polynomial in at most* $t(n)$ *variables, of degree at most* $t(n)$ *and computed by an algebraic branching program with size at most* $\mathrm{poly}(n)$*.*

It is known that both IMM and determinant polynomial families are complete for VBP.

## 2.3 Polynomial families

In this section, we describe some of the polynomial families that the reader may encounter in this work. We start by defining two well-known families of symmetric polynomials.

**Definition 2.12** *For any $n \in \mathbb{N}$ and $1 \leq d \leq n$, the $d$-th elementary symmetric polynomial in $n$ variables is*

$$\mathsf{ESym}_{n,d} = \sum_{S \in \binom{[n]}{d}} \prod_{i \in S} x_i.$$

*The $d$-th power symmetric polynomial in $n$ variables is*

$$\mathsf{PSym}_{n,d} = \sum_{i \in [n]} x_i^d.$$

The sum-product polynomials are an extremely simple polynomial family.

**Definition 2.13** *For any $s \in \mathbb{N}$ and $1 \leq d \leq n$, the $d$-th sum-product polynomial in $sd$ variables is,*

$$\mathsf{SP}_{s,d} := \sum_{i \in [s]} \prod_{j \in [d]} x_{i,j}.$$

The following two polynomial families and their variants have been used extensively in proving arithmetic circuit lower bounds [NW97, Raz10, KLSS17, KS17b, KS16, KST16a, KST16b, FKS16, CLS19, KS19b, GST20, LST21, KS22].

**Definition 2.14 (Nisan-Wigderson design polynomial)** *For a prime power $q$ and $d \in \mathbb{N}$, let $\mathbf{x} = \{x_{1,1}, \ldots, x_{1,q}, \ldots, x_{d,1}, \ldots, x_{d,q}\}$. For any $k \in [d]$, the Nisan-Wigderson design polynomial on $qd$ variables, denoted by $\mathsf{NW}_{q,d,k}$ or simply $\mathsf{NW}$, is defined as follows:*

$$\mathsf{NW}_{q,d,k} = \sum_{\substack{h(z) \in \mathbb{F}_q[z]: \\ \deg(h) < k}} \prod_{i \in [d]} x_{i,h(i)}.$$

**Definition 2.15 (Iterated Matrix Multiplication polynomial)** *For any $n \in \mathbb{N}$ and $1 \leq d \leq n$, the iterated matrix multiplication, $\mathsf{IMM}_{n,d}$ is a polynomial in $N = d \cdot n^2$ variables defined as the $(1,1)$-th entry of the matrix product of $d$ many $n \times n$ matrices whose entries are distinct variables.*

In a recent breakthrough [LST21], Limaye, Srinivasan, and Tavenas introduced what they call the word polynomial which is a projection of the IMM.

**Definition 2.16 (Word polynomial $P_{\mathbf{w}}$ [LST21])** *Given a word $\mathbf{w} = (w_1, \ldots, w_d) \in \mathbb{Z}^d$, let $\mathbf{x}(\mathbf{w})$ be a tuple of $d$ pairwise disjoint sets of variables $(\mathbf{x}_1(\mathbf{w}), \ldots, \mathbf{x}_d(\mathbf{w}))$ with $|\mathbf{x}_i(\mathbf{w})| = 2^{|w_i|}$ for all $i \in [d]$. $\mathbf{x}_i(\mathbf{w})$ will be called negative if $w_i < 0$ and positive otherwise. As the set sizes are powers of 2, we can map the variables in a set $\mathbf{x}_i(\mathbf{w})$ to Boolean strings of length $|w_i|$. Let $\sigma : \mathbf{x} \to \{0,1\}^*$ be such a mapping.[1] We extend the definition of $\sigma$ from variables to set-multilinear monomials as follows: Let $X = x_1 \cdots x_r$ be a set-multilinear monomial where $x_i \in \mathbf{x}_{\phi(i)}(\mathbf{w})$ and $\phi : [r] \to [d]$ be an increasing function. Then, we define a Boolean string $\sigma(X) := \sigma(x_1) \circ \cdots \circ \sigma(x_r)$, where $\circ$ denotes the concatenation of bits. Let $\mathcal{M}_+(\mathbf{w})$ and $\mathcal{M}_-(\mathbf{w})$ denote the set of all (monic) set-multilinear monomials over all the positive sets and all the negative sets, respectively. For two Boolean strings $a, b$, we say $a \sim b$ if $a$ is a prefix of $b$ or vice versa. For a word $\mathbf{w}$, the corresponding word polynomial $P_{\mathbf{w}}$ is defined as*

$$P_{\mathbf{w}} := \sum_{\substack{m_+ \in \mathcal{M}_+(\mathbf{w}),\, m_- \in \mathcal{M}_-(\mathbf{w}) \\ \sigma(m_+) \sim \sigma(m_-)}} m_+ \cdot m_-.$$

Notice that if $\sum_{w_i \geq 0} |w_i| \leq \sum_{w_i < 0} |w_i|$, then for any $m_+ \in \mathcal{M}_+$ and $m_- \in \mathcal{M}_-$, $\sigma(m_+)$ will be a prefix of $\sigma(m_-)$ and if $\sum_{i \in [d]: w_i \geq 0} |w_i| > \sum_{i \in [d]: w_i < 0} |w_i|$, then for any $m_+ \in \mathcal{M}_+$ and $m_- \in \mathcal{M}_-$, $\sigma(m_-)$ will be a prefix of $\sigma(m_+)$. In Chapter 7, we will make use of the following lemma from [LST21] which shows that $IMM$ is at least as hard as $P_{\mathbf{w}}$. For this, we recall the notion of *unbiased*-ness of $\mathbf{w} = (w_1, \ldots, w_d)$ from [LST21] – we say that $\mathbf{w}$ is *$h$-unbiased* if $\max_{i \in [d]} |w_1 + \cdots + w_i| \leq h$.

**Lemma 2.1 (Lemma 7 in [LST21])** *Let $\mathbf{w} \in [-h..h]^d$ be $h$-*unbiased*. If for some $n \geq 2^h$, $IMM_{n,d}$ has a formula $\mathsf{C}$ of product-depth $\Delta$ and size $s$, then $P_{\mathbf{w}}$ has a formula $\mathsf{C}'$ of product-depth at most $\Delta$ and size at most $s$. Moreover, if $\mathsf{C}$ is homogeneous, then so is $\mathsf{C}'$.*

## 2.4 Complexity measures

This section describes the two complexity measures that we use in Chapter 7.

Let $n$ and $n_0$ be positive integers. Define variable sets $\mathbf{x} := \{x_1, \ldots, x_n\}$ and $\mathbf{z} := \{z_1, \ldots, z_{n_0}\}$. For a monic monomial $m$ and an $f \in \mathbb{F}[\mathbf{x}]$, we define $\partial_m f \in \mathbb{F}[\mathbf{x}]$ to be the polynomial obtained by successively taking partial derivatives with respect to all the variables of $m$ (counted with their multiplicities). For an integer $\ell \geq 0$, $\mathbf{x}^\ell := \{{x_1}^{e_1} \cdots {x_n}^{e_n} : e_1, \ldots, e_n \in \mathbb{Z}_{\geq 0} \text{ and } \sum_{i \in [n]} e_i = \ell\}$. For an integer $k \geq 0$ and $f \in \mathbb{F}[\mathbf{x}]$, $\partial^k f := \{\partial_m f : m \in \mathbf{x}^k\}$. For a $f \in \mathbb{F}[\mathbf{x}]$, a map $L : \mathbf{x} \to \langle \mathbf{z} \rangle$, and $\mathcal{S} \subseteq \mathbb{F}[\mathbf{x}]$, $\pi_L(f) \in \mathbb{F}[\mathbf{z}]$ and $\pi_L(\mathcal{S}) \subseteq \mathbb{F}[\mathbf{z}]$ are defined as $\pi_L(f) := f(L(x_1), \ldots, L(x_n))$ and $\pi_L(\mathcal{S}) := \{\pi_L(f) : f \in \mathcal{S}\}$, respectively.

[1] Note that $\sigma$ may map a variable from $\mathbf{x}_i(\mathbf{w})$ and a variable from $\mathbf{x}_j(\mathbf{w})$ to the same string if $i \neq j$.

For $\mathcal{S}, \mathcal{T} \subseteq \mathbb{F}[\mathbf{x}]$, $\mathcal{S} \cdot \mathcal{T} := \{f \cdot g : f \in \mathcal{S} \text{ and } g \in \mathcal{T}\}$ and $\mathcal{S} + \mathcal{T} := \{f + g : f \in \mathcal{S} \text{ and } g \in \mathcal{T}\}$. For a $\mathcal{S} \subseteq \mathbb{F}[\mathbf{x}]$, we define its *span* as $\langle \mathcal{S} \rangle \subseteq \mathbb{F}[\mathbf{x}]$ to be the set of all polynomials which can be expressed as $\mathbb{F}$-linear combinations of elements in $\mathcal{S}$. For a $\mathcal{S} \subseteq \mathbb{F}[\mathbf{x}]$, its *dimension*, denoted by $\dim \mathcal{S}$, refers to the maximum number of *linearly independent* polynomials in $\mathcal{S}$.

We can now define the complexity measures for polynomials that we use to prove our lower bounds: the *shifted partials* (SP) measure and the *affine projections of partials* (APP) measure. We remark here that both these measures (with different parameters) have been used in the literature prior to our work – for example, the shifted partials measure in [GKKS14, Kay12b] and the affine projections of partials in [GKS20, KNS20].

**Definition 2.17 (SP and APP measures)** *For a polynomial $f \in \mathbb{F}[\mathbf{x}]$, non-negative integers $k, \ell$, and $n_0 \in [n]$, we define* $\mathsf{SP}_{k,\ell}(f) := \dim \left\langle \mathbf{x}^{\ell} \cdot \boldsymbol{\partial}^{k} f \right\rangle$ *and* $\mathsf{APP}_{k,n_0}(f) := \max_{L:\mathbf{x}\to\langle \mathbf{z}\rangle} \dim \left\langle \pi_L \left( \boldsymbol{\partial}^{k} f \right) \right\rangle$.

SP and APP are *sub-additive*. That is, for any $f, g \in \mathbb{F}[\mathbf{x}]$, $\mathsf{SP}(f+g) \leq \mathsf{SP}(f) + \mathsf{SP}(g)$ and $\mathsf{APP}(f+g) \leq \mathsf{APP}(f) + \mathsf{APP}(g)$.

**Remark 2.1** *The lower bounds that we prove in Chapter 7 can also be obtained using the skewed partials measure (*SkewP*) [KNS20], which is a special case of* APP*. [KNS20] used the* SkewP *measure to prove an optimal "non-FPT"*[1] *lower bound of $n^{\Omega(d)}$ for multilinear depth-3 circuits computing* $\mathsf{IMM}_{n,d}$*. However, we use the more general* APP *measure for several reasons: Firstly,* APP *has the geometrically appealing feature that it is invariant under the application of invertible linear transformations on the variables. Secondly, there are models for which* APP *gives lower bounds but* SkewP *does not (see [AGK+23]). The third reason is that for reconstruction of circuits using the recently proposed learning from lower bounds framework [KS19a, GKS20],* APP *might give weaker non-degeneracy conditions than* SkewP*. Thus using* APP*, we might be able to learn more circuits from a circuit class than we can learn using the* SkewP *measure.*

*Also, there is a close connection between* APP *and the relative rank (*relrk*) measure used in [LST21]: Both of them are variants of the evaluation dimension (*evalDim*) measure with the added feature of 'imbalance'. It is natural to wonder to what extent the imbalance is required. The* relrk *measure works with an imbalance between the sizes of the sets involved in a set-multilinear partition. An imbalance or skew between the sizes of variable sets also appears in* APP*, albeit at a* gross level*:* APP *uses two sets – one for taking derivatives, the other for affine projections – and there is an imbalance between the sizes of these two sets. Drawing analogy with* evalDim*, one may also view these two sets as the variables used for evaluations ($\mathbf{y}$) and the remaining variables ($\mathbf{z}$). It turns out that (for*

[1] Borrowing terminology from [LST21].

*set-multilinear polynomials) the "finer" imbalance used in the* relrk *measure implies an imbalance – at a gross level – between* $\mathbf{y}$ *and* $\mathbf{z}$.[1] *One may naturally ask if an imbalance at a gross level, like in* APP *and its precursor* SkewP, *is sufficient to prove lower bounds for low-depth circuits.*

## 2.5 Hitting sets

**Definition 2.18 (Hitting set)** *Let $\mathcal{C}$ be a set of $n$-variate polynomials. A set of points $\mathcal{H} \subseteq \mathbb{F}^n$ is a hitting set for $\mathcal{C}$ if for every non-zero $f \in \mathcal{C}$, there is a point $\mathbf{a} \in \mathcal{H}$ such that $f(\mathbf{a}) \neq 0$.*

By a 'T-time hitting set', we mean that the hitting set can be computed in $T$ time. Typically, $T$ is a function of the input parameters such as the number of variables, the size of the input circuit, and the degree or the individual degree of the input polynomial. The *individual degree* of a monomial is the largest of the exponents of the variables that appear in it. The individual degree of a polynomial is the largest of the individual degrees of its monomials.

**Definition 2.19 (Hitting set generator)** *Let $\mathcal{C}$ be a set of $n$-variate polynomials and $t \in \mathbb{N}$. A polynomial map $\mathcal{G} : \mathbb{F}^t \to \mathbb{F}^n$ is a hitting set generator for $\mathcal{C}$ if for every non-zero $f \in \mathcal{C}$, we have $f \circ \mathcal{G} \neq 0$.*

We say the number of variables of $\mathcal{G}$ is $t$, and the degree of $\mathcal{G}$ – denoted by $\deg(\mathcal{G})$ – is the maximum of the degrees of the $n$ polynomials that define $\mathcal{G}$. We will denote the $t$-variate polynomial $f \circ \mathcal{G}$ by $f(\mathcal{G})$. By treating a matrix $A \in \mathbb{F}^{n \times n}$ as a linear transformation from $\mathbb{F}^n$ to $\mathbb{F}^n$, we will denote the polynomial map $A \circ \mathcal{G}$ by $A\mathcal{G}$ and the $t$-variate polynomial $f \circ A\mathcal{G}$ by $f(A\mathcal{G})$. If the defining polynomials of $\mathcal{G}$ have degree $d_0$ and the degree of the polynomials in $\mathcal{C}$ is at most $D$, then the degree of $f(\mathcal{G})$ is at most $d_0 D$. Thus, if we are given the defining polynomials of $\mathcal{G}$, then we can construct a hitting set for $\mathcal{C}$ in time $\text{poly}(n, (d_0 D)^t)$ using the Schwartz-Zippel lemma, provided also that $|\mathbb{F}| > d_0 D$.

### 2.5.1 The Shpilka-Volkovich generator

**Definition 2.20 (The Shpilka-Volkovich or SV hitting set generator [SV15])** *Assume that $|\mathbb{F}| \geq n$ and let $\alpha_1, \ldots, \alpha_n$ be distinct elements of $\mathbb{F}$. For $i \in [n]$, let*

$$L_i(y) := \prod_{j \in [n], j \neq i} \frac{y - \alpha_j}{\alpha_i - \alpha_j}$$

[1] [LST21] talks about the (relative) rank of the partial derivatives matrix. The rank of this matrix is evalDim with respect to an appropriate set $\mathbf{y}$.

*be the i-th Lagrange interpolation polynomial. Then, for $t \in \mathbb{N}$, the Shpilka-Volkovich (SV) generator $\mathcal{G}^{SV}_t : \mathbb{F}^{2t} \to \mathbb{F}^n$ is defined as $\mathcal{G}^{SV}_t := \left(\mathcal{G}^{(1)}_t, ..., \mathcal{G}^{(n)}_t\right)$ where,*

$$\mathcal{G}^{(i)}_t(y_1, \ldots, y_t, z_1, ..., z_t) = \sum_{k=1}^{t} L_i(y_k) \cdot z_k.$$

The SV generator is also sometimes called the $t$-wise independent generator or a $t$-wise independent monomial map. Notice that $\deg\left(\mathcal{G}^{(i)}_t\right) = n$, and $\mathcal{G}^{SV}_{t+1|(y_{t+1}=\alpha_i)} = \mathcal{G}^{SV}_t + \mathbf{e}_i \cdot z_{t+1}$, where $\mathbf{e}_i$ is the $i$-th standard basis vector of $\mathbb{F}^n$. Thus, $\text{Img}\left(\mathcal{G}^{SV}_t\right) \subseteq \text{Img}\left(\mathcal{G}^{SV}_{t+1}\right)$ and, continuing in this manner, $\text{Img}\left(\mathcal{G}^{SV}_t\right) \subseteq \text{Img}\left(\mathcal{G}^{SV}_{t'}\right)$ for any $t' \geq t$. We say that a polynomial $f \in \mathbb{F}[\mathbf{x}]$ depends only on variables $x_1, \ldots, x_b$ if $f \in \mathbb{F}[x_1, \ldots, x_b]$.

**Observation 2.1** *Let $f \in \mathbb{F}[\mathbf{x}]$ be a non-zero polynomial that depends on only $b$ of the $\mathbf{x}$ variables, and $g \in \text{orb}(f)$. Then, $g$ has a monomial of support at most $b$ and $g(\mathcal{G}^{SV}_b) \neq 0$.*

**Proof:** Suppose that $f$ depends on only the variables $x_1, \ldots, x_b$. Let $g = f(A\mathbf{x}) \neq 0$, where $A \in \text{GL}(n, \mathbb{F})$. Suppose that $A$ maps $x_i \mapsto \ell_i(\mathbf{x})$ for all $i \in [n]$. As $A$ is invertible, $\ell_1, \ldots, \ell_n$ are $\mathbb{F}$-linearly independent. Let $B$ be the $b \times n$ matrix whose $i$-th row is the coefficient vector of $\ell_i$ for all $i \in [b]$. Then, $\text{rank}(B) = b$ and there are $b$ columns $j_1, \ldots, j_b$ of $B$ that are also linearly independent. This means the linear forms $\ell'_1, \ldots, \ell'_b$ obtained from $\ell_1, \ldots, \ell_b$ after setting the variables other than $x_{j_1}, \ldots, x_{j_b}$ to 0 are also linearly independent. Thus, $g(\ell'_1, \ldots, \ell'_b) \neq 0$ which is only possible if $g$ has a monomial whose support is contained in $\{x_{j_1}, \ldots, x_{j_b}\}$. Now observe that $g\left(\mathcal{G}^{SV}_{b\,|(y_1=\alpha_{j_1}, y_2=\alpha_{j_2}, ..., y_b=\alpha_{j_b})}\right) \neq 0$. □

The following observation, which allows us to construct a hitting set generator for a polynomial $f$ from a hitting set generator for $\frac{\partial f}{\partial x_i}$ will be used crucially in the proofs of Theorems 3.1, 1.4 and 1.5.

**Observation 2.2** *Let $f \in \mathbb{F}[\mathbf{x}]$ be an $n$-variate, degree $d$ polynomial, and for some $m \in \mathbb{N}$, let $\mathcal{G} : \mathbb{F}^m \to \mathbb{F}^n$ be a polynomial map of degree at most $d'$. If $|\mathbb{F}| > dd'$ and there is an $i \in [n]$ such that $\frac{\partial f}{\partial x_i}(\mathcal{G}) \neq 0$, then $f(\mathcal{G} + \mathcal{G}^{SV}_1)$ is not a constant.*

**Proof:** If $\frac{\partial f}{\partial x_i}(\mathcal{G}) \neq 0$, then the Schwartz-Zippel lemma implies that there is a $(\beta_1, ..., \beta_n) \in \text{Img}(\mathcal{G})$ such that

$$\frac{\partial f}{\partial x_i}(\beta_1, ..., \beta_n) \neq 0,$$

because $\deg\left(\frac{\partial f}{\partial x_i}(\mathcal{G})\right) \leq dd'$ and $|\mathbb{F}| > dd'$. Let $r(z_1) := f(\beta_1, ..., \beta_{i-1}, \beta_i + z_1, \beta_{i+1}, ..., \beta_n)$. Then,

$$\frac{\partial r}{\partial z_1}(0) = \frac{\partial f}{\partial x_i}(\beta_1, ..., \beta_n) \neq 0,$$

and so, $f(\beta_1, ..., \beta_{i-1}, \beta_i + z_1, \beta_{i+1}, ..., \beta_n)$ is not a constant. Now, $\mathcal{G} + \mathcal{G}_1^{SV}{}_{|(y_1=\alpha_i)} = \mathcal{G} + \mathbf{e}_i \cdot z_1$. Let $\text{Img}_{z_1}(\mathcal{G} + \mathcal{G}_1^{SV})$ be the "partial image" of $\mathcal{G} + \mathcal{G}_1^{SV}$ obtained by keeping the $z_1$ variable alive and setting all other variables to field elements. This means that $(\beta_1, ..., \beta_{i-1}, \beta_i + z_1, \beta_{i+1}, ..., \beta_n) \in \text{Img}_{z_1}(\mathcal{G} + \mathcal{G}_1^{SV})$, and hence, $f(\mathcal{G} + \mathcal{G}_1^{SV})$ is not a constant. □

**Observation 2.3** *Let $f \in \mathbb{F}[\mathbf{x}]$ be an $n$-variate, degree $d$ polynomial, and for some $m \in \mathbb{N}$, let $\mathcal{G} : \mathbb{F}^m \to \mathbb{F}^n$ be a polynomial map of degree at most $d'$. If $|\mathbb{F}| > dd'$, then for any affine form $\ell \in \mathbb{F}[\mathbf{x}]$ and variable sets $\mathbf{u}$ and $\{y, z\}$ such that $|\mathbf{u}| = m$ and $\mathbf{u} \cap \{y, z\} = \emptyset$, $f(\mathcal{G}(\mathbf{u}) + \mathcal{G}_1^{SV}(y, z)) = 0$ only if $g(\mathcal{G}) = 0$, where $g := f(x_i = \ell)$.*

**Proof:** Suppose that $g(\mathcal{G}) \neq 0$. Then there exists $(\alpha_1, \ldots, \alpha_n) \in \text{Img}(\mathcal{G})$ such that

$$f(\alpha_1, \ldots, \alpha_{i-1}, \beta, \alpha_{i+1} \ldots, \alpha_n) = g(\alpha_1, \ldots, \alpha_n) \neq 0,$$

where $\beta := \ell(\alpha_1, \ldots, \alpha_n)$. It is easy to see that $(\beta - \alpha_i)\mathbf{e}_i$, where $\mathbf{e}_i \in \mathbb{F}^n$ is the $i$-th basis vector is in the image of $\mathcal{G}_1^{SV}$. So $(\alpha_1, \ldots, \alpha_{i-1}, \beta, \alpha_{i+1} \ldots, \alpha_n) \in \text{Img}(\mathcal{G} + \mathcal{G}_1^{SV})$ and $f(\mathcal{G}(\mathbf{u}) + \mathcal{G}_1^{SV}(y, z)) \neq 0$. □

### 2.5.2 Low support rank concentration

We now define the notion of low support rank concentration, which is a crucial component of this work. The support of a monomial is defined to be the number of distinct variables in it. For a monomial $\mathbf{x}^{\boldsymbol{\alpha}}$, we will denote its support by $\text{Supp}(\mathbf{x}^{\boldsymbol{\alpha}})$.

**Definition 2.21** *Let $F$ be a polynomial in $\mathbf{x}$-variables with coefficients from $\mathbb{K}^{w \times w}$, where $\mathbb{K}$ is a field and $w \in \mathbb{N}$. For an $m \in \mathbb{N}$, we say that $F$ has* support-$m$ rank concentration *over $\mathbb{K}$ if the coefficient of every monomial in $F$ is in the $\mathbb{K}$-span of the coefficients of the monomials of support at most $m$ in $F$.*

**Observation 2.4** *Let $f = \mathbf{1}^T \cdot M_1(x_1)M_2(x_2) \cdots M_n(x_n) \cdot \mathbf{1} \in \mathbb{F}[\mathbf{x}]$ be computable by an ROABP of width $w$, and $F = M_1(x_1)M_2(x_2) \cdots M_n(x_n)$. For an $m \in \mathbb{N}$ and $t_1(\mathbf{z}), \ldots, t_n(\mathbf{z}) \in \mathbb{F}[\mathbf{z}]$, where $\mathbf{z}$ is a set of variables different from $\mathbf{x}$, suppose that $F(\mathbf{x} + \mathbf{t}(\mathbf{z})) := M_1(x_1 + t_1(\mathbf{z}))M_2(x_2 + t_2(\mathbf{z})) \cdots M_n(x_n + t_n(\mathbf{z})) \in \mathbb{F}(\mathbf{z})^{w \times w}[\mathbf{x}]$ has support-$m$ rank concentration over $\mathbb{F}(\mathbf{z})$. Then, $f(x_1 + t_1(\mathbf{z}), \ldots, x_n + t_n(\mathbf{z}))$, when viewed as a polynomial in $\mathbf{x}$-variables with coefficients from $\mathbb{F}[\mathbf{z}]$, has an $\mathbf{x}$-monomial of support at most $m$, provided $f \neq 0$.*

**Proof:** Let $F(\mathbf{x}+\mathbf{t}(\mathbf{z})) = \sum_{\boldsymbol{\alpha}} C_{\boldsymbol{\alpha}} \mathbf{x}^{\boldsymbol{\alpha}}$, where $C_{\boldsymbol{\alpha}} \in \mathbb{F}[\mathbf{z}]^{w\times w}$. Then, $f(x_1+t_1(\mathbf{z}),\ldots,x_n+t_n(\mathbf{z})) = \sum_{\boldsymbol{\alpha}} \left(\mathbf{1}^T\cdot C_{\boldsymbol{\alpha}}\cdot\mathbf{1}\right)\mathbf{x}^{\boldsymbol{\alpha}}$. If $f\neq 0$, then there is an $\boldsymbol{\alpha}$ such that $\mathbf{1}^T\cdot C_{\boldsymbol{\alpha}}\cdot\mathbf{1}\neq 0$. If $\text{Supp}\left(\mathbf{x}^{\boldsymbol{\alpha}}\right)\leq m$, then there is nothing to prove. Otherwise, as $F(\mathbf{x}+\mathbf{t}(\mathbf{z}))$ has support-$m$ rank concentration over $\mathbb{F}(\mathbf{z})$, $C_{\boldsymbol{\alpha}}$ is in the $\mathbb{F}(\mathbf{z})$-span of $\left\{C_{\boldsymbol{\beta}}:\ \text{Supp}\left(\mathbf{x}^{\boldsymbol{\beta}}\right)\leq m\right\}$. Thus, there is a $\boldsymbol{\beta}$ with $\text{Supp}\left(\mathbf{x}^{\boldsymbol{\beta}}\right)\leq m$ such that $\mathbf{1}^T\cdot C_{\boldsymbol{\beta}}\cdot\mathbf{1}$ is non-zero, as $\mathbf{1}^T\cdot C_{\boldsymbol{\alpha}}\cdot\mathbf{1}$ is non-zero. □

### 2.5.3 Algebraic rank and faithful homomorphisms

We say that polynomials $f_1,\ldots,f_m\in\mathbb{F}[\mathbf{x}]$ are algebraically independent over $\mathbb{F}$, if they do not satisfy any non-trivial polynomial equation over $\mathbb{F}$, i.e., for any $p\in\mathbb{F}[y_1,\ldots,y_m]$, $p(f_1,\ldots,f_m)=0$ only if $p=0$. For $\mathbf{f}=(f_1,\ldots,f_m)$, the transcendence degree (i.e., the algebraic rank) of $\mathbf{f}$ over $\mathbb{F}$ is the cardinality of any maximal algebraically independent subset of $\{f_1,\ldots,f_m\}$ over $\mathbb{F}$. The notion of algebraic rank is well defined as algebraic independence satisfies the matroid properties.

For $\mathbf{f}=(f_1,\ldots,f_m)\in\mathbb{F}[\mathbf{x}]^m$, let

$$J_{\mathbf{x}}(\mathbf{f}) := \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \cdots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \frac{\partial f_m}{\partial x_2} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix}_{m\times n}$$

denote the Jacobian matrix of $\mathbf{f}$. The following well-known lemma relates the transcendence degree of $\mathbf{f}$ over $\mathbb{F}$ – denoted by $\text{tr-deg}_{\mathbb{F}}(\mathbf{f})$ – to the rank of the Jacobian.

**Lemma 2.2 (The Jacobian criterion)** *Let $\mathbf{f}=(f_1,\ldots,f_m)\in\mathbb{F}[\mathbf{x}]^m$ be a tuple of polynomials of degree at most $D$ and $\text{tr-deg}_{\mathbb{F}}(\mathbf{f})=r$. If $\text{char}(\mathbb{F})=0$ or $\text{char}(\mathbb{F})>D^r$, then $\text{tr-deg}_{\mathbb{F}}(\mathbf{f})=\text{rank}_{\mathbb{F}(\mathbf{x})}J_{\mathbf{x}}(\mathbf{f})$.*

**Definition 2.22 (Faithful homomorphisms)** *A homomorphism $\phi:\mathbb{F}[\mathbf{x}]\to\mathbb{F}[\mathbf{z}]$ is said to be* faithful *to $\mathbf{f}=(f_1,\ldots,f_m)\in\mathbb{F}[\mathbf{x}]^m$ if $\text{tr-deg}_{\mathbb{F}}(\mathbf{f})=\text{tr-deg}_{\mathbb{F}}(\phi(\mathbf{f}))$.*

**Lemma 2.3 (Theorem 2.4 in [ASSS16])** *If a homomorphism $\phi:\mathbb{F}[\mathbf{x}]\to\mathbb{F}[\mathbf{z}]$ is faithful to $\mathbf{f}=(f_1,\ldots,f_m)\in\mathbb{F}[\mathbf{x}]^m$, then for any $p\in\mathbb{F}[y_1,\ldots,y_m]$, $p(\mathbf{f})=0$ if and only if $p(\phi(\mathbf{f}))=0$.*

The following lemma was proved in [ASSS16, BMS13].

**Lemma 2.4 (Lemma 2.7 of [ASSS16])** *Let* $\mathbf{f} = (f_1, ..., f_m)$ *be a tuple of polynomials of degree at most* $D$, $\text{tr-deg}_{\mathbb{F}}(\mathbf{f}) \leq r$, *and* $\text{char}(\mathbb{F}) = 0$ *or* $> D^r$. *Let* $\psi : \mathbb{F}[\mathbf{x}] \to \mathbb{F}[\mathbf{z}]$ *be a homomorphism such that* $\text{rank}_{\mathbb{F}(\mathbf{x})} J_{\mathbf{x}}(\mathbf{f}) = \text{rank}_{\mathbb{F}(\mathbf{z})} \psi(J_{\mathbf{x}}(\mathbf{f}))$. *Then, the map* $\phi : \mathbb{F}[\mathbf{x}] \to \mathbb{F}[\mathbf{z}, t, y_1, ..., y_r]$ *that, for all* $i \in [n]$, *maps*

$$x_i \to \left( \sum_{j=1}^{r} y_j t^{ij} \right) + \psi(x_i)$$

*is faithful to* $\mathbf{f}$.

We will also need the following observation in our proofs.

**Observation 2.5** *Let* $\mathbf{f} = (f_1, \ldots, f_m) \in \mathbb{F}[\mathbf{x}]^m$ *be a tuple of polynomials with* $\text{tr-deg}_{\mathbb{F}}(\mathbf{f}) = r$. *For any* $A \in \text{GL}(n, \mathbb{F})$, *let* $g_i = f_i(A\mathbf{x})$ *for all* $i \in [m]$ *and* $\mathbf{g} = (g_1, \ldots, g_m)$. *Then,* $\text{tr-deg}_{\mathbb{F}}(\mathbf{g}) = r$.

**Proof:** Assume without loss of generality that $f_1, \ldots, f_r$ is a transcendence basis of $\mathbf{f}$. We will show that $g_1, \ldots, g_r$ is a transcendence basis of $\mathbf{g}$. For contradiction, let $p \in \mathbb{F}[y_1, \ldots, y_r]$ be such that $p(g_1, \ldots, g_r) = 0$. Then, $p(g_1, \ldots, g_r) = p(f_1, \ldots, f_r)(A\mathbf{x}) = 0$. As $A$ is invertible, $p(f_1, \ldots, f_r) = 0$. Because $f_1, \ldots, f_r$ are algebraically independent, this implies that $p = 0$, and so, $g_1, \ldots, g_r$ are algebraically independent. Also, if there exists a $j \in [r+1, m]$ such that $g_1, \ldots, g_r, g_j$ are algebraically independent, then for all non-zero $p \in \mathbb{F}[y_1, \ldots, y_{r+1}]$, $p(g_1, \ldots, g_r, g_j) \neq 0$. But, as $p(g_1, \ldots, g_r, g_j) = p(f_1, \ldots, f_r, f_j)(A\mathbf{x})$ and $A$ is invertible, for all $p \neq 0$, $p(f_1, \ldots, f_r, f_j) \neq 0$. This means that $\text{tr-deg}_{\mathbb{F}}(\mathbf{f}) > r$, which contradicts the hypothesis of the observation. □

[GKST17] gave a quasi-polynomial time hitting set construction for the sum of constantly many ROABPs. As ROFs are a sub-class of ROABPs, we have the following theorem which is used in Chapter 8.

**Theorem 2.1** *[GKST17] Let* $\mathbb{F}$ *be a field of size at least* $\text{poly}(n)$. *There exists a hitting set generator* $\mathcal{G} : \mathbb{F}^m \to \mathbb{F}^n$ *with degree* $\text{poly}(n)$ *and* $m = O(\log n)$ *for polynomials computed by ROABPs or sum of constantly many ROFs.*

## 2.6 Read-once arithmetic formulas

Recall that an arithmetic formula $\mathsf{C}$ is said to be read-once if every variable appears at most once in $\mathsf{C}$. We will identify a gate with the polynomial it computes and $\mathsf{C}$ with the polynomial computed by its top-most gate. Without loss of generality, $\mathsf{C}$ has alternate layers of $+$ and $\times$ gates, every non-leaf gate has fan-in at least two, and every child of a $\times$ gate computes

a non-constant polynomial. Furthermore, we assume that no leaf nor any wire of an ROF is labelled by 0. This ensures that if $\mathtt{C}$ computes $f$, then all variables present in $\mathtt{C}$ are also present in $f$, that is, they are in $\mathrm{var}(f)$. We shall assume that all ROFs in this work satisfy this property.

The product-depth of $\mathtt{C}$, denoted $\Delta$, is the number of $\times$ gates in a longest path in $\mathtt{C}$ from the root to a leaf. We call $\mathtt{C}$ a $+$-rooted (similarly, a $\times$-rooted) ROF if the root of $\mathtt{C}$ is a $+$ gate (respectively, a $\times$ gate). The following fact is easy to verify.

**Fact 2.1 (Irreducibility of an ROF)** *The polynomial computed by a $+$-rooted ROF is irreducible over $\mathbb{F}$.*

If $f \in \mathbb{F}[\mathbf{x}]$ is computed by an ROF, we shall denote this by $f \in \mathrm{ROF}$. In Chapters 5 and 6, we consider ROFs with some additional structure.

**Definition 2.23 (Canonical ROF)** *An ROF $\mathtt{C}$ is* canonical *if it satisfies the following properties:*

1. *$\mathtt{C}$ has alternate layers of $+$ and $\times$ gates.*
2. *Every non-leaf gate in $\mathtt{C}$ has fan-in at least 2.*
3. *Every child of a $\times$ gate computes a non-constant polynomial.*
4. *There are no labels on the edges of $\mathtt{C}$.*
5. *A $+$ gate has at most one constant and at most one variable among its children, but not both.*
6. *Suppose there is a $+$ gate that has among its children a variable and a $\times$ gate $v$ such that $v$ has two children – a variable and a $+$ gate $v'$. Then, $v'$ has no constant among its children.*

Let $\mathtt{C}$ be a $+$-rooted canonical ROF over $\mathbb{F}$. The equation $\mathtt{C} = T_1 + \cdots + T_s + \gamma$ means that $T_1, \ldots, T_s$ are the non-constant children and $\gamma \in \mathbb{F}$ is the constant child of the root $+$ gate. Note that a constant in a canonical ROF $\mathtt{C}$ only appears as a child of a $+$ gate. Thus, all constants present in $\mathtt{C}$ are additive-constants. An example of a canonical ROF is an ROANF.

**Definition 2.24 (ROANF)** *A canonical ROF $\mathtt{C}$ is in the* read-once alternating normal form (ROANF) *if it is a complete binary tree, the root of $\mathtt{C}$ is a $+$ gate, the bottom-most layer of $\mathtt{C}$ contains $\times$ gates, and all the leaves are labelled with distinct variables.*

**Definition 2.25** *An ROF is* additive-constant-free *if it has no additive-constants. We shall denote the class of additive-constant-free ROFs by* $\text{ROF}_0$. *If* $R \in \text{ROF}_0$ *has* $n$ *variables, then we shall say that* $\mathtt{C} \in \text{ROF}_0(n)$.

For an ROF $\mathtt{C}$ and $x, y \in \mathbf{x}$, $\text{fca}(x, y)$ denotes the first common ancestor gate of $x, y$ in $R$. Because $R$ is an ROF, the first common ancestor is unique for any pair of variables.

The product-depth of any gate $G$ in $\mathtt{C}$ is the number of $\times$ gate on the path from the root to $G$ excluding $G$.

**Definition 2.26** *For an ROF* $\mathtt{C} \in \mathbb{F}[\mathbf{x}]$, *its gate graph* $G = (|\mathbf{x}|, E)$, *where*

$$E := \{\{x, y\} : \text{fca}(x, y) \textit{ is a } \times \textit{ gate}\}.$$

[SV15] showed that $E = \left\{\{x, y\} : \frac{\partial^2 R}{\partial x \partial y} \neq 0\right\}$.

We shall denote the class of polynomials computed by sum of $k$ many ROFs as $\sum_k \text{ROF}$. In addition, we shall use $\sum_k \text{ROF}(n)$ to denote the set of all polynomials computed by the sum of $k$ many ROFs in $n$ variables. The following theorem is required in Chapter 8.

**Theorem 2.2** *[MT18] For each* $n \in \mathbb{N}$, $\text{ESym}_{n,n-1} \notin \sum_{\lceil \frac{n}{2} \rceil - 1} \text{ROF}$.

## 2.7 The Hessian of a polynomial

The Hessian matrix of an ROF is crucially used to obtain an equivalence test for ROFs in Chapter 5. We now define the Hessian of a polynomial.

**Definition 2.27 (Hessian of a polynomial)** *The* Hessian *of* $g \in \mathbb{F}[\mathbf{x}]$, *denoted as* $H_g$, *is the* $n \times n$ *matrix whose* $(i, j)$*-th entry is* $\frac{\partial^2 g}{\partial x_i \partial x_j}$. *The determinant of* $H_g$ *is called the* Hessian determinant *of* $g$.

The Hessian matrix appears naturally in the Taylor expansion of a polynomial and has important applications in optimization, second derivative tests, etc. In algebraic complexity, the rank of the Hessian plays a crucial role in the best known lower bound on the determinantal complexity of the permanent [MR04, CCL10]. The Hessian determinant is an effective tool for designing equivalence tests for the sum-product polynomial, the power symmetric polynomial [Kay11], and the sum of univariates model [GKP18]. A suitable 4-th order generalization of the Hessian has been used in [GKP18] to study the Waring decomposition problem in the average case. In this work, we focus on understanding the essential variables of the Hessian determinant of a general ROF (see Section 5.4) to devise an equivalence test for ROFs.

A few basic properties of the Hessian are given below.

**Observation 2.6** *Let $\mathtt{C} = T_1 + \cdots + T_s + \gamma$ be an ROF over $\mathbb{F}$, where for every $l \in [s], T_l$ is a $\times$-rooted sub-ROF of $\mathtt{C}$ and $\gamma \in \mathbb{F}$. Suppose the rows and columns of $H_{\mathtt{C}}$ are labelled by $\mathrm{var}(T_1), \ldots, \mathrm{var}(T_s)$ in order. Then, $H_{\mathtt{C}}$ is a block diagonal matrix, where for $l \in [s]$, the $l$-th block on the diagonal is $H_{T_l}$.*

**Fact 2.2 (Chain rule)** *Let $g \in \mathbb{F}[\mathbf{x}]$ and $h = g(A\mathbf{x})$ for $A \in M(|\mathbf{x}|, \mathbb{F})$. Then, $H_h = A^T \cdot H_g(A\mathbf{x}) \cdot A$.*

**Fact 2.3** *Let $g \in \mathbb{F}[\mathbf{x}]$ and $\mathbf{b} \in \mathbb{F}^{|\mathbf{x}|}$. Then, $H_{g(\mathbf{x}+\mathbf{b})} = H_g(\mathbf{x}+\mathbf{b})$.*

**Fact 2.4** *Let $g \in \mathbb{F}[\mathbf{x}]$, $A \in \mathrm{GL}(|\mathbf{x}|, \mathbb{F})$, $\mathbf{b} \in \mathbb{F}^{|\mathbf{x}|}$ and $h = g(A\mathbf{x}+\mathbf{b})$. Then, $\det(H_h) = \alpha^2 \cdot \det(H_g)(A\mathbf{x}+\mathbf{b})$, where $\alpha = \det(A)$.*

## 2.8 Essential and redundant variables

The contents of this section are some definitions and preliminary results for the results in Chapters 5 and 6 which are part of a joint work with Nikhil Gupta and Chandan Saha [GST23]. A special case of the result in Chapter 5 is included in [Gup22]. As such, some of the preliminary results mentioned in this section also appear in [Gup22].

**Definition 2.28 (Essential and redundant variables)** *The number of* essential variables *of an $n$-variate $g \in \mathbb{F}[\mathbf{x}]$ is $s := \min_{A \in \mathrm{GL}(n,\mathbb{F})} |\mathrm{var}(g(A\mathbf{x}))|$. The number of* redundant variables *of $g$ is $(n-s)$.*

Following [Car06], we denote the number of essential variables of $g$ by $N_{ess}(g)$. [Car06] gave a polynomial-time algorithm that takes input the coefficient vector of $g$ and finds an $A \in \mathrm{GL}(n, \mathbb{F})$ such that $|\mathrm{var}(g(A\mathbf{x}))| = N_{ess}(g)$. [Kay11] gave a randomized polynomial-time algorithm that does the same given black-box access to $g$. These algorithms use a neat relation between $N_{ess}(g)$ and $\dim \left\langle \frac{\partial g}{\partial x} : x \in \mathbf{x} \right\rangle$. See Claim 2.3 in [KNST17] for a proof of the following fact.

**Fact 2.5 (Essential variables and partials)** *Let $d \in \mathbb{N}$ and $\mathrm{char}(\mathbb{F}) = 0$ or $> d$. If $g \in \mathbb{F}[\mathbf{x}]_{\leq d}$, then $N_{ess}(g) = \dim \left\langle \frac{\partial g}{\partial x} : x \in \mathbf{x} \right\rangle$. For $\mathbf{z} \subseteq \mathbf{x}$, $\left\{ \frac{\partial g}{\partial z} : z \in \mathbf{z} \right\}$ is a basis of $\left\langle \frac{\partial g}{\partial x} : x \in \mathbf{x} \right\rangle$ if and only if there is an $A \in \mathrm{GL}(|\mathbf{x}|, \mathbb{F})$ that maps every variable in $\mathbf{x} \setminus \mathbf{z}$ to itself, $\mathrm{var}(g(A\mathbf{x})) = \mathbf{z}$, and $N_{ess}(g(A\mathbf{x})) = |\mathbf{z}|$.*

In the claim below, we mention a slightly general version of an algorithm to remove redundant variables. Its proof follows from the above fact.

**Claim 2.1 (Elimination of redundant variables)** *Let $d \in \mathbb{N}$, $\mathrm{char}(\mathbb{F}) = 0$ or $> d$, and $|\mathbb{F}| \geq 2|\mathbf{x}|d$. There is a randomized $\mathrm{poly}(|\mathbf{x}|, d)$ time algorithm* Remove-Redundant-Vars *that takes input black-box access to a $g \in \mathbb{F}[\mathbf{x}]_{\leq d}$ and a set $\mathbf{y} \subseteq \mathbf{x}$ s.t. $\mathbf{x} \setminus \mathbf{y}$ contains a set of essential variables of $g$, and outputs an $A \in \mathrm{GL}(|\mathbf{x}|, \mathbb{F})$ s.t. $A$ maps every $\mathbf{y}$-variable to itself and $g(A\mathbf{x})$ is $\mathbf{y}$-free and has no redundant variables.*

We say a set $\mathbf{z} \subseteq \mathbf{x}$ is a *set of essential variables* of $g$ if $\left\{ \frac{\partial g}{\partial z} : z \in \mathbf{z} \right\}$ is a basis of $\left\langle \frac{\partial g}{\partial x} : x \in \mathbf{x} \right\rangle$; variables in $\mathbf{x} \setminus \mathbf{z}$ are *redundant* for $g$. We categorize the essential variables further as follows.

**Definition 2.29 (Truly and ordinary essential variables)** *An $x \in \mathbf{x}$ is a* truly essential variable *of $g \in \mathbb{F}[\mathbf{x}]$ if for every $A \in \mathrm{GL}(|\mathbf{x}|, \mathbb{F})$ that maps $x$ to itself, $x \in \mathrm{var}(g(A\mathbf{x}))$. If $\mathbf{z}$ is a set of essential variables of $g$, then a $z \in \mathbf{z}$ that is not truly essential is an* ordinary essential variable *of $g$ in $\mathbf{z}$.*

**Observation 2.7 (Characterizing truly essential variables using partials)** *Let $d \in \mathbb{N}$ and $\mathrm{char}(\mathbb{F}) = 0$ or $> d$. If $g \in \mathbb{F}[\mathbf{x}]_{\leq d}$, then $x \in \mathbf{x}$ is a truly essential variable of $g$ if and only if $\sum_{x' \in \mathbf{x}} \alpha_{x'} \frac{\partial g}{\partial x'} = 0$ for $\alpha_{x'} \in \mathbb{F}$ implies $\alpha_x = 0$, i.e., no $\mathbb{F}$-linear dependence of $\left\{ \frac{\partial g}{\partial x'} : x' \in \mathbf{x} \right\}$ involves $\frac{\partial g}{\partial x}$.*

It follows that every set of essential variables of $g$ contains all the truly essential variables.

**Proof:** Let $x$ be a truly essential variable of $g$. Suppose there exists $\alpha_{x'} \in \mathbb{F}$, for every $x' \in \mathbf{x}$, such that $\sum_{x' \in \mathbf{x} \setminus \{x\}} \alpha_{x'} \frac{\partial g}{\partial x'} = \alpha_x \frac{\partial g}{\partial x}$, where $\alpha_x \neq 0$. Then, it follows from Fact 2.5 that there is an $A \in \mathrm{GL}(|\mathbf{x}|, \mathbb{F})$ which maps $x$ to itself and $g(A\mathbf{x})$ is $x$-free. Hence, by Definition 2.29, $x$ is not truly essential.

Suppose $x$ is not a truly essential variable of $g$. Then, there exists an $A \in \mathrm{GL}(|\mathbf{x}|, \mathbb{F})$ that maps $x$ to itself and $g(A\mathbf{x})$ is $x$-free. Suppose the column of $A$ labelled by $x$ is $(\alpha_{y,x})^T_{y \in \mathbf{x}}$. Then, it follows from the chain rule of derivatives that $0 = \frac{\partial g(A\mathbf{x})}{\partial x} = \sum_{y \in \mathbf{x}} \alpha_{y,x} \frac{\partial g}{\partial y}(A\mathbf{x})$, which implies $\sum_{y \in \mathbf{x}} \alpha_{x,y} \frac{\partial g}{\partial y} = 0$. As $A$ maps $x$ to itself, $\alpha_{x,x} = 1 \neq 0$. This completes the proof. □

The next fact follows from the proof of Fact 2.5.

**Fact 2.6 (Structure of a matrix for removing redundant variables)** *Let $d \in \mathbb{N}, \mathrm{char}(\mathbb{F}) = 0$ or $> d$, and $g \in \mathbb{F}[\mathbf{x}]_{\leq d}$. Let $\mathbf{z}$ be a set of essential variables of $g$, $\mathbf{z}_1$ the set of truly essential variables of $g$, $\mathbf{z}_2 = \mathbf{z} \setminus \mathbf{z}_1$, and $\mathbf{y} = \mathbf{x} \setminus \mathbf{z}$. Then, there is an $A \in \mathrm{GL}(|\mathbf{x}|, \mathbb{F})$ that maps every variable in $\mathbf{z}_1 \uplus \mathbf{y}$ to itself, maps every $z \in \mathbf{z}_2$ to a linear form in $\mathbf{y} \uplus \{z\}$, and $\mathrm{var}(g(A\mathbf{x})) = \mathbf{z}$.*

**Observation 2.8 (Truly essential variables map to linear forms in essential variables)** *Let $d \in \mathbb{N}, \text{char}(\mathbb{F}) = 0$ or $> d$, $\mathbf{x}$ and $\mathbf{y}$ be disjoint sets of variables, and $h \in \mathbb{F}[\mathbf{x}]_{\leq d}$. Let $\mathbf{z} \subseteq \mathbf{x} \uplus \mathbf{y}$ and $A \in \text{GL}(|\mathbf{x}| + |\mathbf{y}|, \mathbb{F})$ such that $|\mathbf{z}| = N_{ess}(h)$ and $h(A \cdot (\mathbf{x}, \mathbf{y})^T) \in \mathbb{F}[\mathbf{z}]$ (where we pretend that $h$ is a polynomial in $\mathbf{x} \uplus \mathbf{y}$). Then, $A$ maps every truly essential variable of $h$ to a linear form in $\mathbf{z}$.*

**Proof:** Assume that $\mathbf{z} = \mathbf{x}$ and all $\mathbf{x}$-variables are essential for $h$. Suppose, $A = \begin{bmatrix} A_{\mathbf{x}} & A_1 \\ A_2 & A_{\mathbf{y}} \end{bmatrix}$, where the rows and columns of $A_{\mathbf{x}}, A_{\mathbf{y}}$ are labelled by $\mathbf{x}$ and $\mathbf{y}$, respectively. It is sufficient to show that $A_1 = 0$. Let $g = h(A(\mathbf{x}, \mathbf{y})^T) \in \mathbb{F}[\mathbf{x}]$. Pretend that $g, h$ are polynomials in $\mathbf{x} \uplus \mathbf{y}$. Let $\nabla g = ([\nabla g]_{\mathbf{x}}, [\nabla g]_{\mathbf{y}})^T$, where $[\nabla g]_{\mathbf{x}} = \left(\frac{\partial g}{\partial x}\right)_{x \in \mathbf{x}}$ and $[\nabla g]_{\mathbf{y}} = \left(\frac{\partial g}{\partial y}\right)_{y \in \mathbf{y}} = 0$. Similarly, let $\nabla h = ([\nabla h]_{\mathbf{x}}, [\nabla h]_{\mathbf{y}})^T$, where $[\nabla h]_{\mathbf{y}} = 0$. By the chain rule,

$$\nabla g = A^T \cdot [\nabla h](A(\mathbf{x}, \mathbf{y})^T). \tag{2.1}$$

Since $\mathbf{x}$ is the set of essential variables of $h$, the entries in $[\nabla h]_{\mathbf{x}}$ are $\mathbb{F}$-linearly independent, and thus, the entries in $[\nabla h]_{\mathbf{x}}(A(\mathbf{x}, \mathbf{y})^T)$ are also $\mathbb{F}$-linearly independent. Now, it is easy to see from Equation (2.1) and the structure of $A$ that $A_1^T = 0$. Otherwise, we get a non-zero linear combination of $\frac{\partial h}{\partial x}(A(\mathbf{x}, \mathbf{y})^T), x \in \mathbf{x}$, which is equal to 0 (as $[\nabla g]_{\mathbf{y}} = [\nabla h]_{\mathbf{y}} = 0$), and this leads to a contradiction.

Now, suppose $\mathbf{x} \neq \mathbf{z}$. Let $P \in \text{GL}(|\mathbf{x}| + |\mathbf{y}|, \mathbb{F})$ be a permutation matrix that maps $\mathbf{x}$ to $\mathbf{z}$, $\mathbf{z}$ to $\mathbf{x}$ and every other variable to itself. We know $h(A(\mathbf{x}, \mathbf{y})^T) \in \mathbb{F}[\mathbf{z}]$. Then, note that $h(AP(\mathbf{x}, \mathbf{y})^T) \in \mathbb{F}[\mathbf{x}]$. It follows from the above argument that $AP$ maps every $\mathbf{x}$-variable to a linear form in $\mathbf{x}$. This implies $A$ maps every $\mathbf{x}$-variable to a linear form in $\mathbf{z}$.

Now, suppose that not all $\mathbf{x}$-variables are necessarily essential for $h$. Let $C \in \text{GL}(|\mathbf{x}| + |\mathbf{y}|, \mathbb{F})$ be such that $h(C(\mathbf{x}, \mathbf{y})^T)$ has no redundant variables. Then, Fact 2.6 implies that every truly essential variable of $h$ is in $\text{var}(h(C(\mathbf{x}, \mathbf{y})^T))$. The argument in the above paragraph implies that $C^{-1}A$ maps all variables in $\text{var}(h(C(\mathbf{x}, \mathbf{y})^T))$ to linear forms in $\mathbf{z}$. Because $C$ maps every truly essential variable of $h$ to itself, $A$ maps every truly essential variable to a linear form in $\mathbf{z}$. □

**Observation 2.9 (Truly essential variables from factors)** *Let $d \in \mathbb{N}, \text{char}(\mathbb{F}) = 0$ or $> d$, and $\mathbf{x}$ and $\mathbf{y}$ be disjoint sets of variables. Let $h(\mathbf{x}, \mathbf{y}) = g(\mathbf{x})^e \cdot p(\mathbf{x}, \mathbf{y}) \in \mathbb{F}[\mathbf{x}, \mathbf{y}]$, where $g(\mathbf{x}), p(\mathbf{x}, \mathbf{y})$ are coprime, $\deg(h) \leq d$, and $e \geq 1$. If $N_{ess}(g) = |\mathbf{x}|$, then every $\mathbf{x}$-variable is truly essential for $h$.*

**Proof:** Suppose, $\sum_{x \in \mathbf{x}} \alpha_x \frac{\partial h}{\partial x} + \sum_{y \in \mathbf{y}} \beta_y \frac{\partial h}{\partial y} = 0$. Since $h = g(\mathbf{x})^e \cdot p(\mathbf{x}, \mathbf{y})$ and $e \geq 1$, we get

$$\sum_{x \in \mathbf{x}} \alpha_x \left( g^e \frac{\partial p}{\partial x} + e \cdot g^{e-1} \cdot p \frac{\partial g}{\partial x} \right) + \sum_{y \in \mathbf{y}} \beta_y g^e \frac{\partial p}{\partial y} = 0.$$

On dividing the above equation by $g^{e-1}$ and rearranging the terms we get

$$g\left(\sum_{x\in\mathbf{x}}\alpha_x\frac{\partial p}{\partial x}+\sum_{y\in\mathbf{y}}\beta_y\frac{\partial p}{\partial y}\right)+e\cdot p\left(\sum_{x\in\mathbf{x}}\alpha_x\frac{\partial g}{\partial x}\right)=0.$$

As $g$ and $p$ are coprime, and $\deg\left(\sum_{x\in\mathbf{x}}\alpha_x\frac{\partial g}{\partial x}\right)<\deg(g)$, we get $\sum\limits_{x\in\mathbf{x}}\alpha_x\frac{\partial g}{\partial x}=0$. But, this implies $\alpha_x=0$ for every $x\in\mathbf{x}$, since $N_{ess}(g)=|\mathbf{x}|$. Hence, every $\mathbf{x}$-variable is truly essential for $h$. □

**Observation 2.10 (Truly essential pairs of variables)** *Let $d\in\mathbb{N}, \mathrm{char}(\mathbb{F})=0$ or $>d$, and $\{x_1,x_2\}$ and $\mathbf{y}$ be disjoint sets of variables. Let $h(x_1,x_2,\mathbf{y})=\sum_{i\geq 0}p_i(\mathbf{y})\cdot(x_1x_2)^i$ be a polynomial of degree at most $d$ such that $p_i(\mathbf{y})\neq 0$ for some $i\geq 1$. Then, $x_1$ and $x_2$ are truly essential for $h$.*

**Proof:** Let $\alpha_1\frac{\partial h}{\partial x_1}+\alpha_2\frac{\partial h}{\partial x_2}+\sum_{y\in\mathbf{y}}\beta_y\frac{\partial h}{\partial y}=0$, where $\alpha_1,\alpha_2,\beta_y\in\mathbb{F}$ for $y\in\mathbf{y}$. Since $h=\sum_{i\geq 0}p_i(\mathbf{y})(x_1x_2)^i$,

$$\alpha_1\left(\sum_{i\geq 1}i\cdot p_i\cdot x_1^{i-1}x_2^i\right)+\alpha_2\left(\sum_{i\geq 1}i\cdot p_i\cdot x_1^ix_2^{i-1}\right)+\sum_{y\in\mathbf{y}}\beta_y\left(\sum_{i\geq 0}(x_1x_2)^i\frac{\partial p_i}{\partial y}\right)=0.$$

Notice that $\alpha_1\left(\sum_{i\geq 1}i\cdot p_i\cdot x_1^{i-1}x_2^i\right)$, $\alpha_2\left(\sum_{i\geq 1}i\cdot p_i\cdot x_1^ix_2^{i-1}\right)$, and $\sum_{y\in\mathbf{y}}\beta_y\left(\sum\limits_{i\geq 0}(x_1x_2)^i\frac{\partial p_i}{\partial y}\right)$ are monomial disjoint. Thus, each of these three polynomials is zero. Suppose $\alpha_1\neq 0$. Then, $\sum\limits_{i\geq 1}i\cdot p_i\cdot x_1^{i-1}x_2^i=0$. As $\mathrm{char}(\mathbb{F})=0$ or $>d$, we get $p_i=0$ for every $i\geq 1$, which is a contradiction. Thus, $\alpha_1=0$. Similarly, $\alpha_2=0$. Hence, $x_1$ and $x_2$ are truly essential for $h$.

□

**Essential variables modulo affine forms.** Let $g\in\mathbb{F}[\mathbf{x}]$ and $\ell=\sum_{i\in[n]}\alpha_ix_i+\beta$, a non-constant affine form in $\mathbb{F}[\mathbf{x}]$. Let $I=\{i\in[n]:\alpha_i\neq 0\}$, and $\prec$ a monomial ordering on $\mathbb{F}[\mathbf{x}]$. Suppose, $x_j$ has the highest precedence among $\{x_i:i\in I\}$ according to $\prec$. There is a natural ring isomorphism between the quotient ring $\mathbb{F}[\mathbf{x}]/\langle\ell\rangle$ and $\mathbb{F}[\mathbf{x}\setminus\{x_j\}]$, where $\langle\ell\rangle$ is the ideal generated by $\ell$. We define $g$ modulo $\ell$ (denoted $g_\ell$) as:

$$g_\ell:=g\left(x_1,\ldots,x_{j-1},-{\alpha_j}^{-1}\left(\sum_{i\in[n]\setminus\{j\}}\alpha_ix_i+\beta\right),x_{j+1},\ldots,x_n\right).$$

**Definition 2.30** *The number of essential variables of $g$ modulo $\ell$ is defined as $N_{ess}(g_\ell)$.*

Notice that the ordering $\prec$ is implicit in the above definition. But, the following observation shows that the exact choice of $\prec$ is unimportant here.

**Observation 2.11 (Soundness of Definition 2.30)** *For $j \in I$, let $W_j = \left\langle \frac{\partial \varphi_j(g)}{\partial x} \ : \ x \in \mathbf{x} \right\rangle$, where $\varphi_j(g)$ is obtained by substituting $x_j$ in $g$ by $-{\alpha_j}^{-1}\left(\sum_{i\in[n]\setminus\{j\}} \alpha_i x_i + \beta\right)$. Then, for $j_1, j_2 \in I$, $\dim W_{j_1} = \dim W_{j_2}$.*

**Proof:** For $j \in I$, let $\varphi_j$ be the substitution map defined as: $\varphi_j(x_j) = -{\alpha_j}^{-1}\left(\sum_{i\in[n]\setminus\{j\}} \alpha_i x_i + \beta\right)$, and $\varphi(x) = x$ for every $x \in \mathbf{x} \setminus \{x_j\}$. For $j_1, j_2 \in I$, let $g_{j_1} := \varphi_{j_1}(g)$ and $g_{j_2} := \varphi_{j_2}(g)$. Observe that $g_{j_1} - g_{j_2} \in \langle \ell \rangle$, which implies $g_{j_1} = \varphi_{j_1}(g_{j_2})$, as $\varphi_{j_1}(\ell) = 0$ and $g_{j_1}$ is $x_{j_1}$-free. Hence, by chain rule, $\frac{\partial g_{j_1}}{\partial x_i} = \varphi_{j_1}\left(\frac{\partial g_{j_2}}{\partial x_i}\right) - \alpha_{j_1}^{-1}\alpha_i \cdot \varphi_{j_1}\left(\frac{\partial g_{j_2}}{\partial x_{j_1}}\right)$. As $g_{j_2}$ is $x_{j_2}$-free,

$$\frac{\partial g_{j_1}}{\partial x_{j_2}} = -\alpha_{j_1}^{-1}\alpha_{j_2} \cdot \varphi_{j_1}\left(\frac{\partial g_{j_2}}{\partial x_{j_1}}\right) \implies \frac{\partial g_{j_1}}{\partial x_i} - \alpha_{j_2}^{-1}\alpha_i \cdot \frac{\partial g_{j_1}}{\partial x_{j_2}} = \varphi_{j_1}\left(\frac{\partial g_{j_2}}{\partial x_i}\right).$$

Notice that the space spanned by $\left\{\frac{\partial g_{j_1}}{\partial x_i} - \alpha_{j_2}^{-1}\alpha_i \cdot \frac{\partial g_{j_1}}{\partial x_{j_2}} \ : \ x_i \in \mathbf{x}\right\}$ is $W_{j_1}$, which (by the above equations) is $U := \left\langle \varphi_{j_1}\left(\frac{\partial g_{j_2}}{\partial x_i}\right) \ : \ x_i \in \mathbf{x} \right\rangle$. As $\varphi_{j_1}$ is linear, $\dim U \leq \dim W_{j_2}$, implying $\dim W_{j_1} \leq \dim W_{j_2}$. Similarly, we can show that $\dim W_{j_2} \leq \dim W_{j_1}$. Therefore, $\dim W_{j_1} = \dim W_{j_2}$. □

## 2.9 The orbit of a polynomial

For $A \in \mathrm{GL}(n, \mathbb{F})$, $A\mathbf{x}$ denotes the column vector obtained by multiplying $A$ with $(x_1 \cdots x_n)^T$.

**Definition 2.31 (Orbit of a polynomial)** *The* orbit of $g \in \mathbb{F}[\mathbf{x}]$, *denoted* $\mathrm{orb}(g)$, *is defined as* $\mathrm{orb}(g) := \{g(A\mathbf{x} + \mathbf{b}) \ : \ A \in \mathrm{GL}(|\mathbf{x}|, \mathbb{F}),\ \mathbf{b} \in \mathbb{F}^{|\mathbf{x}|}\}$.

*Remark.* Some results in this work, like those in Chapters 3, 4, 5, 6, holds even if we consider a more general definition of the orbit of a polynomial: Let $\mathbf{x}, \mathbf{y}$ be sets of $n$ and $m$ variables, where $n \geq m$, and $h \in \mathbb{F}[\mathbf{x}]$, $g \in \mathbb{F}[\mathbf{y}]$. Then, $h \in \mathrm{orb}(g)$ if there is exist a $A \in \mathbb{F}^{m\times n}$ of rank $m$ and $\mathbf{b} \in \mathbb{F}^m$ such that $h = g(A\mathbf{x} + \mathbf{b})$.

By the 'orbit of a circuit class $\mathcal{C}$', we mean the union of the orbits of the polynomials computable by the circuits in the class $\mathcal{C}$.

**Definition 2.32 (PS orbit of a polynomial)** *Let $g, h \in \mathbb{F}[\mathbf{x}]$. We say that $h$ is in the* PS orbit of $g$, *denoted* $\mathrm{PS}(g)$, *if there exist a $|\mathbf{x}| \times |\mathbf{x}|$ permutation matrix $P$, a $|\mathbf{x}| \times |\mathbf{x}|$ invertible diagonal (scaling) matrix $S$, and a $\mathbf{b} \in \mathbb{F}^{|\mathbf{x}|}$ such that $h = g(PS\mathbf{x} + \mathbf{b})$.*

The following facts are easy to prove.

**Fact 2.7** *If $h \in \text{orb}(g)$, then $N_{ess}(g) = N_{ess}(h)$.*

**Fact 2.8** *Let $d \in \mathbb{N}$, $\text{char}(\mathbb{F}) = 0$ or $> d$, $g \in \mathbb{F}[\mathbf{x}]_{\leq d}$, $\ell \in \mathbb{F}[\mathbf{x}]$ a non-constant affine form, $A \in \text{GL}(|\mathbf{x}|, \mathbb{F})$, $\mathbf{b} \in \mathbb{F}^{|\mathbf{x}|}$, $g' = g(A\mathbf{x} + \mathbf{b})$, and $\ell' = \ell(A\mathbf{x} + \mathbf{b})$. Then, $N_{ess}(g'_{\ell'}) = N_{ess}(g_\ell)$.*

## 2.10 The border of a circuit class

Recall from Chapter 1 that the closure or border of a circuit class $\mathcal{C}$, denoted by $\overline{\mathcal{C}}$ is the Zariski closure of $\mathcal{C}$ obtained by identifying each $n$-variate degree-$d$ polynomial computed by some circuit in $\mathcal{C}$ with its coefficient vector. We now give an alternative and equivalent definition of $\overline{\mathcal{C}}$. The equivalence between these two notions of border was proved in [Bür01].

Let $\epsilon$ be a variable distinct from $\mathbf{x}$. For $f \in \mathbb{F}[\mathbf{x}]$ and $g \in \mathbb{F}[\epsilon, \mathbf{x}]$, we say that $g$ approximates $f$ if there exists an $h \in \mathbb{F}[\epsilon, \mathbf{x}]$ such that $f + \epsilon h = g$. Alternatively, (over $\mathbb{C}$ and $\mathbb{R}$), $f = \lim_{\epsilon \to 0} g$. Throughout this work, we shall denote a polynomial in $\epsilon \cdot \mathbb{F}[\epsilon, \mathbf{x}]$ by $O(\epsilon)$. Thus, $f$ is approximated by $g$ if $f + O(\epsilon) = g$.

If $\mathsf{C}$ is a circuit over $\mathbb{F}(\epsilon)$ that computes $g$, that is, $\mathsf{C}$ can have "constants" from $\mathbb{F}(\epsilon)$ and these constants come for "free" and do not count towards the size of $\mathsf{C}$, then we say that $f$ is in the border of $\mathsf{C}$ denoted by $f \in \overline{\mathsf{C}}$. For a circuit class $\mathcal{C}$, its border denoted by $\overline{\mathcal{C}}$ is the union of the border of all circuits in it. For example, the class $\overline{\mathsf{VP}}$ contains all polynomial families of polynomial degree that can be approximated by polynomial sized arithmetic circuits over $\mathbb{F}(\epsilon)$.

Note that for $\gamma(\epsilon) \in \mathbb{F}(\epsilon)$, $\gamma$ can be written as $\epsilon^{-N} \cdot \gamma'$ for some $N \in \mathbb{N}$ and a formal power series $\gamma'$. A folklore result states that if $f$ is in the border of $\mathsf{C}$, then every $\mathbb{F}(\epsilon)$-element appearing in $\mathsf{C}$ can be written as $\epsilon^{-N} \cdot \gamma$ for some $\gamma \in \mathbb{F}[\epsilon]$; i.e. every $\mathbb{F}(\epsilon)$-element appearing in $\mathsf{C}$ is actually a Laurent polynomial. Throughout this work, whenever we say that $\mathsf{C}$ is a circuit over $\mathbb{F}(\epsilon)$, we assume that all $\mathbb{F}(\epsilon)$-elements in it are Laurent series. For an $\alpha \neq 0 = \sum_{i \in a}^{b} \alpha_i \epsilon^i \in \mathbb{F}(\epsilon)$, $\text{val}(\alpha)$ is the smallest $i \in \{a, \ldots, b\}$ such that $\alpha_i \neq 0$. $\text{val}(0) := \infty$. For a $g \in \mathbb{F}(\epsilon)[\mathbf{x}]$, $\text{val}(g)$ is the minimum valuation of a coefficient of any monomial of $g$. For an $\alpha \in \mathbb{F}[\epsilon]$, $\alpha(0) \in \mathbb{F}$ denotes the constant term of $\alpha$.

The following folklore claim follows from Nisan's characterisation of ROABPs [Nis91].

**Claim 2.2** $\overline{ROABP} = ROABP$.

It is also known that $\overline{\text{ROF}} = \text{ROF}$. In Chapter 8, we need a similar statement for additive constant free ROFs that we now prove.

**Claim 2.3** $\overline{\text{ROF}_0} = \text{ROF}_0$.

**Proof:** By induction on the number of variables $n$ in a $\mathtt{C} \in \overline{\text{ROF}_0}$. For $n = 1$, $f = \alpha x + O(\epsilon)$ for some $\alpha \in \mathbb{F}(\epsilon)$. Since $\alpha x + O(\epsilon) \in \mathbb{F}[\epsilon]$, $\alpha \in \mathbb{F}[\epsilon]$. Thus, $f = \alpha(0) \cdot x$ and the claim is true. Let the claim be true for all $f' \in \overline{\text{ROF}_0(n-1)}$ and let $f \in \overline{\text{ROF}_0(n)}$; say it is $\mathtt{C} + O(\epsilon)$. Note that $\mathtt{C} \in \mathbb{F}[\epsilon][\mathbf{x}]$.

If the top gate of $\mathtt{C}$ is a $+$ gate then $\mathtt{C} = T_1 + T_2$, where $T_i$s are ROFs over $\mathbb{F}(\epsilon)$. They are also non-constant, variable disjoint, and additive constant-free. Hence, they must be ROFs over $\mathbb{F}[\epsilon]$. As they have fewer than $n$ variables, from the induction hypothesis $T_i(\epsilon = 0)$ is in $\text{ROF}_0$ and $f = T_1(\epsilon = 0) + T_2(\epsilon = 0) \in \text{ROF}_0$. Otherwise, $\mathtt{C} = Q_1 Q_2$ where $Q_i$ are non-constant, variable disjoint, additive constant-free ROFs over $\mathbb{F}(\epsilon)$. Without loss of generality, $\mathsf{val}(Q_1) \leq \mathsf{val}(Q_2)$. As $\mathtt{C} \in \mathbb{F}[\epsilon][\mathbf{x}]$ and $Q_1, Q_2$ are variable disjoint, $\mathsf{val}(Q_1) + \mathsf{val}(Q_2) \geq 0$. Let $\ell := \max\{0, -\mathsf{val}(Q_1)\}$ and $Q_1' := \epsilon^{\ell} Q_1$, $Q_2' := \epsilon^{-\ell} Q_2'$. Note that $Q_1', Q_2' \in \mathbb{F}[\epsilon][\mathbf{x}]$. As they have fewer than $n$ variables, the induction hypothesis gives that, $Q_1'(\epsilon = 0)$ and $Q_2'(\epsilon = 0)$ are in $\text{ROF}_0$. Thus, $f = Q_1'(\epsilon = 0) Q_2'(\epsilon = 0)$ is also in $\text{ROF}_0$. □

# Chapter 3

# Hitting sets for orbits of ROABPs

This section gives hitting set constructions for orbits of low individual degree commutative ROABPs and multilinear constant-width ROABPs. The contents of this chapter are from a joint work with Chandan Saha [ST24].

## 3.1 Introduction

A polynomial $f \in \mathbb{F}[\mathbf{x}]$ where $\mathbf{x} = (x_1, \ldots, x_n)$ the orbit of $f$, is defined as

$$\text{orb}(f) := \{f(A\mathbf{x} + \mathbf{b}) : A \in \text{GL}(n, \mathbb{F}), \mathbf{b} \in \mathbb{F}^n\};$$

see Definition 2.31. Recall that for a circuit class $\mathcal{C}$, its orbit $\text{orb}(\mathcal{C})$ is defined as $\cup_{f \in \mathcal{C}} \text{orb}(f)$. In this section, we describe quasi-polynomial time hitting set constructions for two subclasses of ROABPs: commutative ROABPs with low individual degree and multilinear ROABPs with constant width (see Definition 2.7). As corollaries we obtain quasi-polynomial time hitting sets for orbits of sums of products of low degree univariate polynomials and sparse polynomials with low individual degree. We construct these hitting sets using the rank concentration by translation technique introduced in [ASS13] and later used in multiple works like [FSS14] to obtain hitting sets for various sub-classes of ROABPs. In this section, we show how this technique can be extended to the setting of orbits.

This section contains the proofs of the following theorems.

**Theorem 1.1 (Hitting sets for the orbits of commutative ROABPs with low individual degree)** *Let $\mathcal{C}$ be the set of $n$-variate polynomials with individual degree at most $d$ that are computable by width-$w$ commutative ROABPs. If $|\mathbb{F}| > n^2 d$, then a hitting set for $\text{orb}(\mathcal{C})$ can be computed in $(nd)^{O(d \log w)}$ time.*

**Theorem 1.3 (Hitting sets for the orbits of multilinear constant-width ROABPs)** *Let $\mathcal{C}$ be the set of $n$-variate multilinear polynomials that are computable by width-$w$ ROABPs. If $|\mathbb{F}| > n^{O(w^4)}$, then a hitting set for $\mathrm{orb}(\mathcal{C})$ can be computed in $n^{O(w^6 \cdot \log n)}$ time.*

As corollaries to Theorem 1.1, we obtain the following two theorems.

**Theorem 1.2 (Hitting sets for the orbits of sums of products of low degree univariates)** *Let $\mathcal{C}$ be the set of $n$-variate polynomials that can be expressed as sums of $s$ products of univariates of degree at most $d$. If $|\mathbb{F}| > n^2 d$, then a hitting set for $\mathrm{orb}(\mathcal{C})$ can be computed in $(nd)^{O(d \log s)}$ time.*

**Theorem 3.1 (Hitting sets for orbits of sparse polynomials)** *Let $\mathcal{C}$ be the set of $n$-variate, $s$-sparse polynomials with individual degree at most $d$. If $|\mathbb{F}| > nd$ and $\mathrm{char}(\mathbb{F}) = 0$ or $> d$, then a hitting set for $\mathrm{orb}(\mathcal{C})$ can be computed in $(nd)^{O(d \log s)}$ time.*

Theorems 1.1, 1.2, and 3.1 are proved in Section 3.2 while Theorem 1.3 is proved in Section 3.3. Since both low individual degree commutative ROABPs and constant width multilinear ROABPs are closed under translation, we just need to prove Theorems 1.1 and 1.3 for polynomials of the form $f(A\mathbf{x})$ where $f$ is computed by a low individual degree commutative ROABP or constant width multilinear ROABP.

## 3.2 Hitting sets for the orbits of commutative ROABPs

**The strategy.** Theorem 1.1 is proved by adapting the rank concentration by translation technique of Agrawal, Saha, and Saxena [ASS13][1] to work for the orbits of commutative ROABPs. Let $f = \mathbf{1}^T \cdot M_1(x_1)M_2(x_2)\cdots M_n(x_n) \cdot \mathbf{1}$ be a width-$w$ commutative ROABP; here $M_i(x_i) \in \mathbb{F}^{w \times w}[x_i]$ for all $i \in [n]$. Also, let $F = M_1(x_1)M_2(x_2)\cdots M_n(x_n)$. For any $A \in \mathrm{GL}(n, \mathbb{F})$, let $g = f(A\mathbf{x})$ and $G = F(A\mathbf{x})$. For $i \in [n]$, suppose that $A$ maps $x_i \mapsto \ell_i(\mathbf{x})$, where $\ell_i$ is a linear form, and let $y_i = \ell_i(\mathbf{x})$ and $\mathbf{y} = \{y_1, \ldots, y_n\}$. Then, $g = \mathbf{1}^T \cdot M_1(y_1)M_2(y_2)\cdots M_n(y_n) \cdot \mathbf{1}$ and $G = M_1(y_1)M_2(y_2)\cdots M_n(y_n)$. We will show that if $g \neq 0$, then there exist explicit "low" degree polynomials $t_1(\mathbf{z}), \ldots, t_n(\mathbf{z})$, where $\mathbf{z}$ is a "small" set of variables such that $g(x_1 + t_1(\mathbf{z}), \ldots, x_n + t_n(\mathbf{z}))$ has a "low" support monomial. This will be done by proving that $G(x_1 + t_1(\mathbf{z}), \ldots, x_n + t_n(\mathbf{z}))$ has low support rank concentration in the "$\mathbf{y}$-variables". Applying Observation 2.4, we will get that $g(x_1 + t_1(\mathbf{z}), \ldots, x_n + t_n(\mathbf{z}))$ has a low support $\mathbf{y}$-monomial. This will then imply that $g(x_1 + t_1(\mathbf{z}), \ldots, x_n + t_n(\mathbf{z}))$ has a low support $\mathbf{x}$-monomial, provided $f$ has low individual degree. Finally, we will plug in the

[1] [ASS13] proved their result for products of univariate polynomials over a Hadamard algebra which form a subclass of commutative ROABPs. However, their analysis also works for general commutative ROABPs.

SV generator (defined in Section 2.5.1) to preserve the non-zeroness of $g$. More precisely, we will prove the following theorem at the end of Section 3.2.2.

**Theorem 3.2** *Let $f$ be an $n$-variate polynomial with individual degree at most $d$ that is computable by a width-$w$ commutative ROABP. If $|\mathbb{F}| \geq n$, then $\mathcal{G}^{SV}_{(2\lceil \log w^2 \rceil (d+1)+1)}$ is a hitting set generator for* $\mathrm{orb}(f)$.

Our analysis differs from that in [ASS13] at a crucial point: In [ASS13], it was shown that $F(\mathbf{x}+\mathbf{t}) = M_1(x_1+t_1)M_2(x_2+t_2)\cdots M_n(x_n+t_n)$ has low support rank concentration over $\mathbb{F}(\mathbf{t})$ if the nonzeroness of every polynomial in a certain collection of polynomials – each in a "small" set of $\mathbf{t}$-variables – is preserved. As each polynomial in the collection has "few" $\mathbf{t}$-variables, a substitution $t_i \leftarrow t_i(\mathbf{z})$ that preserves its nonzeroness is relatively easy to construct. But the collection of polynomials that we need to preserve to show low support rank concentration for $G(\mathbf{x}+\mathbf{t})$ is such that every polynomial in the collection has potentially all the $\mathbf{t}$-variables. However, we are able to argue that each of these polynomials still has a low support $\mathbf{t}$-monomial. This then helps us construct a substitution $t_i \mapsto t_i(\mathbf{z})$ that preserves the nonzeroness of these polynomials.

**Notations and conventions.** In the analysis, we will treat $t_1(\mathbf{z}), \ldots, t_n(\mathbf{z})$ as formal variables $\mathbf{t} = (t_1, \ldots, t_n)$ while always keeping in mind the substitution map $t_i \mapsto t_i(\mathbf{z})$. For $i \in [n]$, let $r_i = \ell_i(\mathbf{t})$. For $S \subseteq [n]$, define $\mathbf{r}_S = \{r_i : i \in S\}$. The $\mathbb{F}$-linear independence of $\ell_1, \ldots, \ell_n$ allows us to treat $\mathbf{y}$ and $\mathbf{r}$ as sets of formal variables. Notice that in this notation, $G(\mathbf{x}+\mathbf{t}) = M_1(y_1+r_1)M_2(y_2+r_2)\cdots M_n(y_n+r_n)$. Let $\mathbb{A}$ denote the matrix algebra $\mathbb{F}^{w\times w}$. For $i \in [n]$, let $M_i(y_i) = \sum_{e_i=0}^{d} u_{i,e_i} y_i^{e_i}$, where $u_{i,e_i} \in \mathbb{A}$ and $M_i(y_i+r_i) = \sum_{b_i=0}^{d} v_{i,b_i} y_i^{b_i}$, where $v_{i,b_i} \in \mathbb{A}[r_i] \subset \mathbb{A}[\mathbf{t}]$. As $f$ is a commutative ROABP, $M_1(y_1), \ldots, M_n(y_n)$ commute with each other and hence $u_{i,e_i}$ and $u_{j,e_j}$ also commute for $i \neq j$. The following observation, implies that $v_{i,e_i}$ and $v_{j,e_j}$ also commute for $i \neq j$.

**Observation 3.1** *For every $i \in [n]$ and $b_i, e_i \in \{0, \ldots, d\}$,*

1. $v_{i,b_i} = \sum_{e_i=0}^{d} \binom{e_i}{b_i} \cdot r_i^{e_i-b_i} \cdot u_{i,e_i}$,
2. $u_{i,e_i} = \sum_{b_i=0}^{d} \binom{b_i}{e_i} \cdot (-r_i)^{b_i-e_i} \cdot v_{i,b_i}$,

*where $\binom{a}{b} = 0$ if $a < b$.*

**Proof:** The proof of Observation 3.1 follows from the following which we prove below: Let $p(y) = \sum_{e=0}^{d} w_e y^e$, where $w_e \in \mathbb{A}$ and $p(y+r) = \sum_{b=0}^{d} \widetilde{w}_b y^b$. Then, $\widetilde{w}_b = \sum_{e=0}^{d} \binom{e}{b} r^{e-b} w_e$.

$$\begin{aligned} p(y+r) &= \sum_{e=0}^{d} w_e (y+r)^e \\ &= \sum_{e=0}^{d} w_e \sum_{b=0}^{d} \binom{e}{b} r^{e-b} y^b \\ &= \sum_{b=0}^{d} \left( \sum_{e=0}^{d} \binom{e}{b} r^{e-b} w_e \right) y^b. \end{aligned}$$

Thus, $\widetilde{w}_b = \sum_{e=0}^{d} \binom{e}{b} r^{e-b} w_e$. □

For a set $S = \{i_1, i_2, \ldots, i_{|S|}\} \subseteq [n]$, where $i_1 < i_2 < \ldots < i_{|S|}$, the vector $(b_{i_1}, b_{i_2}, \ldots, b_{i_{|S|}})$ will be denoted by $(b_i : i \in S)$. Let $\mathrm{Supp}\,(\mathbf{b})$ denote the support of the vector $\mathbf{b}$ which is defined as the number of non-zero elements in it. We also define a parameter $m := 2\lceil \log w^2 \rceil + 1$.

### 3.2.1 The goal: low support rank concentration

We set ourselves the goal of proving that there exist explicit degree-$n$ polynomials $t_1(\mathbf{z}), \ldots, t_n(\mathbf{z})$, where $|\mathbf{z}| = 2m$, such that $G(x_1 + t_1(\mathbf{z}), \ldots, x_n + t_n(\mathbf{z})) = M_1(y_1 + r_1) M_2(y_2 + r_2) \cdots M_n(y_n + r_n) \in \mathbb{A}[r_1, \ldots, r_n][\mathbf{y}]$ has support-$(m-1)$ rank concentration over $\mathbb{F}(\mathbf{z})$ in the $\mathbf{y}$-variables. We will show in this and the next section that this happens if all polynomials in a certain collection of non-zero polynomials $\left\{ h_S(\mathbf{r}_S) : S \subseteq \binom{[n]}{m} \right\} \subseteq \mathbb{F}[r_1, \ldots, r_n]$, where $\deg_{\mathbf{r}_S}(h_S(\mathbf{r}_S)) \leq m d^{m+1}$, remain non-zero under the substitution $t_i \mapsto t_i(\mathbf{z})$.[1] The following lemma will help us achieve this goal.

**Lemma 3.1** *Let $G, \mathbf{t}, \mathbf{z}, \mathbf{y}$ and $\mathbf{r}_S$ be as defined above. Suppose that the following two conditions are satisfied:*

1. *For every $S \subseteq \binom{[n]}{m}$ and $(b_i : i \in S) \in \{0, \ldots, d\}^m$, there is a non-zero polynomial $h_S(\mathbf{r}_S)$ such that*

$$h_S(\mathbf{r}_S) \cdot \prod_{i \in S} v_{i,b_i} \in \mathbb{F}[\mathbf{t}]\text{-span} \left\{ \prod_{i \in S} v_{i,b'_i} \ : \ \mathrm{Supp}\,(b'_i : i \in S) < m \right\}.$$

[1] We do not really need the degree bound on $h_S(\mathbf{r}_S)$.

2. *There exists a substitution $t_i \mapsto t_i(\mathbf{z})$ that keeps $h_S(\mathbf{r}_S)$ non-zero for all $S \subseteq \binom{[n]}{m}$.*

*Then, for every $\mathbf{b} = (b_i : i \in [n]) \in \{0, \ldots, d\}^n$,*

$$\prod_{i\in[n]} v_{i,b_i} \in \mathbb{F}(\mathbf{z})\text{-span}\left\{\prod_{i\in[n]} v_{i,b'_i} \;:\; \text{Supp}\left(b'_i : i \in [n]\right) < m\right\}, \tag{3.1}$$

*and $G(x_1 + t_1(\mathbf{z}), \ldots, x_n + t_n(\mathbf{z}))$ has support-$(m-1)$ rank concentration in the $\mathbf{y}$-variables over $\mathbb{F}(\mathbf{z})$.*

**Proof:** Consider a $\mathbf{b} = (b_i : i \in [n]) \in \{0, \ldots, d\}^n$ with $\text{Supp}(\mathbf{b}) \geq m$. Pick an $S \subseteq \binom{[n]}{m}$ such that $\text{Supp}\left(b_i : i \in S\right) = m$. As $h_S(\mathbf{r}_S)$ is a non-zero polynomial and the substitution $t_i \mapsto t_i(\mathbf{z})$ keeps it non-zero,

$$\prod_{i\in S} v_{i,b_i} \in \mathbb{F}(\mathbf{z})\text{-span}\left\{\prod_{i\in S} v_{i,b'_i} : \text{Supp}\left(b'_i : i \in S\right) < m\right\}.$$

Also, as $v_{i,b_i}$ and $v_{j,b_j}$ commute when $i \neq j$,

$$\begin{aligned}
\prod_{i\in[n]} v_{i,b_i} &\in \mathbb{F}(\mathbf{z})\text{-span}\left\{\prod_{i\in S} v_{i,b'_i} \cdot \prod_{j\in[n]\setminus S} v_{j,b_j} : \text{Supp}\left(b'_i : i \in S\right) < m\right\} \\
&= \mathbb{F}(\mathbf{z})\text{-span}\left\{\prod_{i\in[n]} v_{i,b'_i} : \text{Supp}\left(b'_i : i \in S\right) < m \text{ and } b'_i = b_i \;\forall i \in [n] \setminus S\right\} \\
&\subseteq \mathbb{F}(\mathbf{z})\text{-span}\left\{\prod_{i\in[n]} v_{i,b'_i} : \text{Supp}\left(b'_i : i \in [n]\right) < \text{Supp}(\mathbf{b})\right\}.
\end{aligned}$$

Repeat the above argument for every $\mathbf{b}' \in \{0, \ldots, d\}^n$ such that $m \leq \text{Supp}(\mathbf{b}') < \text{Supp}(\mathbf{b})$. Continuing in this manner yields (3.1) for all $\mathbf{b} \in \{0, \ldots, d\}^n$. Since $\prod_{i\in[n]} v_{i,b_i}$ is the coefficient of the monomial $\mathbf{y}^{\mathbf{b}} := y_1^{b_1} \cdots y_n^{b_n}$ in $G(x_1 + t_1(\mathbf{z}), \ldots, x_n + t_n(\mathbf{z}))$, the polynomial $G(x_1 + t_1(\mathbf{z}), \ldots, x_n + t_n(\mathbf{z}))$ has support-$(m-1)$ rank concentration in the $\mathbf{y}$-variables over $\mathbb{F}(\mathbf{z})$. □

### 3.2.2 Achieving rank concentration

We will now see how to satisfy conditions 1 and 2 of Lemma 3.1 such that $\deg_{\mathbf{r}_S}\left(h_S(\mathbf{r}_S)\right) \leq md^{m+1}$, $t_i(\mathbf{z})$ is an explicit degree-$n$ polynomial, and $|\mathbf{z}| = 2m$. It follows from Lemma

3.1 that to achieve rank concentration, we only need focus on sets $S$ of size exactly $m$. Assume without loss of generality that $S = [m]$. For $\mathbf{b} = (b_1, \ldots, b_m)$ and $\mathbf{e} = (e_1, \ldots, e_m)$ in $\{0, \ldots, d\}^m$, define $\binom{\mathbf{b}}{\mathbf{e}} := \prod_{i \in [m]} \binom{b_i}{e_i}$, where, as before, $\binom{b_i}{e_i} = 0$ if $b_i < e_i$. Also, let $v_{\mathbf{b}} := \prod_{i \in [m]} v_{i,b_i}$ and $u_{\mathbf{e}} := \prod_{i \in [m]} u_{i,e_i}$. Define $\mathbf{r} := (-r_1, \ldots, -r_m)$, $\mathbf{r}^{\mathbf{b}} := \prod_{i \in [m]} (-r_i)^{b_i}$ and $\mathbf{r}^{-\mathbf{e}} := \prod_{i \in [m]} (-r_i)^{-e_i}$. We now define some vectors and matrices by fixing an arbitrary order on the elements of $\{0, \ldots, d\}^m$.

Let $V := \left(v_{\mathbf{b}} : \mathbf{b} \in \{0, \ldots, d\}^m\right)$ and $U := \left(u_{\mathbf{e}} : \mathbf{e} \in \{0, \ldots, d\}^m\right)$; $V$ is a row vector in $\mathbb{A}[\mathbf{r}]^{(d+1)^m}$ whereas $U$ is a row vector in $\mathbb{A}^{(d+1)^m}$. Let $C := \mathrm{diag}(\mathbf{r}^{\mathbf{b}} : \mathbf{b} \in \{0, \ldots, d\}^m)$ and $D := \mathrm{diag}(\mathbf{r}^{-\mathbf{e}} : \mathbf{e} \in \{0, \ldots, d\}^m)$; both $C$ and $D$ are $(d+1)^m \times (d+1)^m$ diagonal matrices. Finally, let $M$ be a $(d+1)^m \times (d+1)^m$ numeric matrix whose rows and columns are indexed by $\mathbf{b} \in \{0, \ldots, d\}^m$ and $\mathbf{e} \in \{0, \ldots, d\}^m$ respectively. The entry of $M$ indexed by $(\mathbf{b}, \mathbf{e})$ contains $\binom{\mathbf{b}}{\mathbf{e}}$. We now make the following claim.

**Claim 3.1** *Let $U$, $V$, $C$, $M$ and $D$ be as defined above. Then, $U = VCMD$.*

**Proof:** The entry indexed by $\mathbf{e} \in \{0, \ldots, d\}^m$ of $U$ is $u_{\mathbf{e}}$. Observe that

$$\begin{aligned}
u_{\mathbf{e}} &= \prod_{i \in [m]} u_{i,e_i} \\
&= \prod_{i \in [m]} \left( \sum_{b_i = 0}^{d} \binom{b_i}{e_i} \cdot (-r_i)^{b_i - e_i} \cdot v_{i,b_i} \right) && \text{(from Observation 3.1)} \\
&= \sum_{\mathbf{b} = (b_1, \ldots, b_n) \in \{0, \ldots, d\}^m} \binom{\mathbf{b}}{\mathbf{e}} \prod_{i \in [m]} (-r_i)^{b_i} \cdot \prod_{i \in [m]} v_{i,b_i} \cdot \prod_{i \in [m]} (-r_i)^{-e_i} \\
&= \sum_{\mathbf{b} \in \{0, \ldots, d\}^m} \binom{\mathbf{b}}{\mathbf{e}} \cdot \mathbf{r}^{\mathbf{b}} \cdot v_{\mathbf{b}} \cdot \mathbf{r}^{-\mathbf{e}} \\
&= \sum_{\mathbf{b} \in \{0, \ldots, d\}^m} v_{\mathbf{b}} \cdot \mathbf{r}^{\mathbf{b}} \cdot \binom{\mathbf{b}}{\mathbf{e}} \cdot \mathbf{r}^{-\mathbf{e}}.
\end{aligned}$$

The equation $U = VCMD$ now follows easily from the definitions of these matrices. □

In [ASS13], a very similar equation was called the *transfer equation* and we will refer to $U = VCMD$ by the same name. Let $F := \left\{\mathbf{b} \in \{0, \ldots, d\}^m : \mathrm{Supp}(\mathbf{b}) = m\right\}$; clearly, $|F| = d^m$. [1] Also, let us call the set of all vectors $\left(n_{\mathbf{e}} : \mathbf{e} \in \{0, \ldots, d\}^m\right) \in \mathbb{F}^{(d+1)^m}$ for which $\sum_{\mathbf{e} \in \{0, \ldots, d\}^m} n_{\mathbf{e}} u_{\mathbf{e}} = 0$ the *null space* of $U$. Then, we have the following lemma.

[1] There is a slight overloading of notation here: We have used $F$ before at the beginning of Section 3.2 to denote the product $M_1(x_1)M_2(x_2)\cdots M_n(x_n)$. However, since all our arguments involve only $G = F(A\mathbf{x})$ and not $F$, we would use $F$ in this section to denote the set that is mentioned here.

**Lemma 3.2** *There are vectors $\{\mathbf{n_b} : \mathbf{b} \in F\}$ in the null space of $U$ such that the following holds: Let $N$ be the $(d+1)^m \times d^m$ matrix whose rows are indexed by $\mathbf{e} \in \{0,\dots,d\}^m$ and whose columns are indexed by $\mathbf{b} \in F$ and whose column indexed by $\mathbf{b}$ is $\mathbf{n_b}$. Then, the square matrix $[CMDN]_F$ is invertible, where $[CMDN]_F$ is the sub-matrix of $CMDN$ consisting of only those rows of $CMDN$ that are indexed by $\mathbf{b} \in F$.*

The proof of the above lemma is similar to the proof of Theorem 13 in [ASS13]. At a high level, the proof proceeds as follows: We first impose an order on the monomials in $\mathbf{r}$ variables and observe that this induces an order on the rows and columns of the matrices $U, V, C, M, D, N$. Then we use this order and a certain greedy argument to construct an $N$ whose columns are in the null space of $U$. We again use the order along with the Cauchy-Binet identity for the determinant to argue that $\det([M]_F DN)$ has an $\mathbf{r}$ monomial with the maximum order. As $[C]_F$ is a diagonal matrix, this proves the lemma.

**Proof:** The entries of $U$, the columns of $M$, the rows and columns of $D$, and the rows of $N$ are indexed by $\mathbf{e} \in \{0,\dots,d\}^m$. Impose an order $\prec$, say the lexicographical order, on the indices $\mathbf{e} \in \{0,\dots,d\}^m$ of $U$ and the other three matrices. Pick the *minimal* basis of the space spanned by the entries of $U$ according to this order, i.e., consider the entries of $U$ in the order dictated by $\prec$ while forming the basis. Let $\mathcal{B} := \{\mathbf{e} \in \{0,\dots,d\}^m : u_{\mathbf{e}} \text{ is in the minimal basis of } U \text{ w.r.t. } \prec\}$.

**Construction of the matrix $N$.** The columns of $N$ are indexed by $\mathbf{b} \in F$. We will now specify a set of column vectors $\{\mathbf{n_b} : \mathbf{b} \in F\}$ in the null space of $U$ such that the column of $N$ indexed by $\mathbf{b} \in F$ is $\mathbf{n_b}$. There are two cases for $\mathbf{b} \in F$:

**Case 1: $\mathbf{b} \in F \setminus \mathcal{B}$.** In this case, $u_{\mathbf{b}}$ is dependent on $\{u_{\mathbf{e}} : \mathbf{e} \in \mathcal{B} \text{ and } \mathbf{e} \prec \mathbf{b}\}$. Pick this dependence vector as $\mathbf{n_b}$.
**Case 2: $\mathbf{b} \in F \cap \mathcal{B}$.** Let there be $p$ such $\mathbf{b}$, where $p \leq |\mathcal{B}| \leq w^2$. For a set $E \subseteq [m]$ and $\mathbf{b} \in \{0,\dots,d\}^m$, let $(\mathbf{b})_E$ denote the vector obtained by projecting $\mathbf{b}$ to the coordinates in $E$. Roughly speaking, the following claim says that each of these $p$ vectors has a "small signature" that differentiates it from the other $p-1$ vectors.

**Claim 3.2** *There exists a way of numbering all $\mathbf{b} \in F \cap \mathcal{B}$ as $\mathbf{b}_1,\dots,\mathbf{b}_p$ and there exist non-empty sets $E_1,\dots,E_p \subseteq [m]$, each of size at most $\log p \leq \log w^2$ such that for all $k \in [p-1]$,*

$$(\mathbf{b}_k)_{E_k} \neq (\mathbf{b}_\ell)_{E_k} \quad \forall \ell \in \{k+1,\dots,p\} \tag{3.2}$$

**Proof:** Suppose that we have already identified $\mathbf{b}_1, \ldots, \mathbf{b}_{k-1}$ for some $k \in [p-1]$ and have constructed $E_1, \ldots, E_{k-1}$ satisfying (3.2). We will show how to identify $\mathbf{b}_k$ and construct $E_k$ *greedily*.

Initially $E_k = \emptyset$. Let $T$ be the set of the $\mathbf{b}$ vectors that have not been numbered yet; $|T| \leq p$. As each vector in $T$ is unique, there exists an index $i_1 \in [m]$ such that the $i_1$-th entry is not the same for all $\mathbf{b} \in T$. In fact, there must exist a $j_1 \in [d]$ such that the number of $\mathbf{b}$ whose $i_1$-th entry is $j_1$ is at least 1 and at most $|T|/2$. Add $i_1$ to $E_k$ and remove from $T$ all those $\mathbf{b}$ whose $i_1$-th entry is not $j_1$. Again, as each vector in $T$ is unique, there exists an index $i_2 \in [m] \setminus E_k$ and a $j_2 \in [d]$ such that the number of $\mathbf{b} \in T$ whose $i_2$-th entry is $j_2$ is at least 1 and at most $|T|/2$. Again, add $i_2$ to $E_k$ and remove from $T$ all those $\mathbf{b}$ whose $i_2$-th entry is not $j_2$. Continuing in this fashion, in $\log p$ or fewer iterations, $|T| = 1$; call the only vector in $T$, $\mathbf{b}_k$ and stop. It is clear that $|E_k| \leq \log p$ and that $\mathbf{b}_k$ and $E_k$ satisfy (3.2).

After having identified $\mathbf{b}_1, \ldots, \mathbf{b}_{p-1}$, call the last remaining vector $\mathbf{b}_p$ and pick $E_p$ to be any arbitrary singleton set. □

We will call $E_k$ the *signature* of $\mathbf{b}_k$ for $k \in [p]$. The following claim tells us that for each vector $\mathbf{b}_k$, there is a vector that is not in $\mathcal{B}$ and has support at most $m-1$, but agrees with $\mathbf{b}_k$ on its signature and so in some sense can be used as a proxy for $\mathbf{b}_k$.

**Claim 3.3** *For every $k \in [p]$, there exists a vector $\mathbf{b}'_k \in \{0, \ldots, d\}^m \setminus (F \cup \mathcal{B})$ such that $(\mathbf{b}'_k)_{E_k} = (\mathbf{b}_k)_{E_k}$ and also $\mathbf{b}'_k$ and $\mathbf{b}_k$ agree on all locations where $\mathbf{b}'_k$ is non-zero.*

**Proof:** As $|E_k| \leq \log w^2$ and $m = 2\lceil \log w^2 \rceil + 1$, for any vector $\mathbf{b}' \in \{0, \ldots, d\}^m$ satisfying $(\mathbf{b}')_{E_k} = (\mathbf{b}_k)_{E_k}$, there are still at least $\lceil \log w^2 \rceil + 1$ coordinates whose values we are free to choose. For each such free coordinate, we choose its value to be either 0 or the value at the same coordinate in $\mathbf{b}_k$. There are $2^{\lceil \log w^2 \rceil + 1} \geq 2w^2$ such $\mathbf{b}'$, one of which is $\mathbf{b}_k$ and the remaining $2w^2 - 1$ are in $\{0, \ldots, d\}^m \setminus F$. As $|\mathcal{B}| \leq w^2$, at least one of these $2w^2 - 1$ vectors is in $\{0, \ldots, d\}^m \setminus (F \cup \mathcal{B})$. Pick any such vector and call it $\mathbf{b}'_k$. □

We will now use the above two claims to construct $\mathbf{n}_{\mathbf{b}_k}$ for all $k \in [p]$. We will use $\mathbf{b}'_k$ from Claim 3.3 as a proxy for $\mathbf{b}_k$. Notice that $u_{\mathbf{b}'_k}$ is dependent on $\{u_{\mathbf{e}} : \mathbf{e} \in \mathcal{B} \text{ and } \mathbf{e} \prec \mathbf{b}'_k\}$. Let this dependence vector be $\mathbf{n}_{\mathbf{b}_k}$. This completes the construction of $N$. We will now show that $[CMDN]_F$ is an invertible matrix.

**$[CMDN]_F$ is invertible.** As $C$ is a diagonal matrix with non-zero entries, it is sufficient to show that $[MDN]_F = [M]_F DN$ is an invertible matrix, where $[M]_F$ is the sub-matrix of $M$ consisting of only those rows of $M$ that are indexed by $\mathbf{b} \in F$. The following claim lets us simplify the structure of $[M]_F$ so that it becomes easier to argue that $[M]_F DN$ is invertible.

**Claim 3.4** *There is a row operation matrix $R \in \mathrm{GL}(d^m, \mathbb{F})$ having determinant 1 such that $R[M]_F$ has the following structure: The rows of $R[M]_F$ are indexed by $\mathbf{b} = (b_1, \ldots, b_m) \in F$ and its columns by $\mathbf{e} = (e_1, \ldots, e_m) \in \{0, \ldots, d\}^m$. Its entry indexed by $(\mathbf{b}, \mathbf{e})$ is non-zero if and only if for all $i \in [m]$, $b_i = e_i$ if $e_i \neq 0$. All the non-zero entries of $R[M]_F$ are either 1 or $-1$.*

**Proof:** We prove the claim by induction on $m$. For $m = 1$,

$$[M]_F = \begin{pmatrix} 1 & \binom{d}{d-1} & \binom{d}{d-2} & \cdots & \binom{d}{1} & 1 \\ 0 & 1 & \binom{d-1}{d-2} & \cdots & \binom{d-1}{1} & 1 \\ 0 & 0 & 1 & \cdots & \binom{d-2}{1} & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 1 \end{pmatrix}.$$

Let $R_1$ be the row operation matrix that multiplies the last row of $[M]_F$ by $\binom{2}{1}$ and subtracts it from the second to last row; then it multiplies the last row by $\binom{3}{1}$, the second to last row by $\binom{3}{2}$ and subtracts them from the third to last row, and so on. Then, the first $d$ columns of $R_1[M]_F$ form a $d \times d$ identity matrix. Also, it is not hard to see that the entry in the last column of the row of $R_1[M]_F$ indexed by $e \in [d]$ is $1 - \binom{e}{1} + \binom{e}{2} - \cdots + (-1)^{e-1}\binom{e}{e-1} = (-1)^{e-1}$. Let $R_1$ be $R$. Also, ignoring the last column of $R[M]_F$ and $[M]_F$, the remaining sub-matrices of both the matrices are upper triangular with ones on the diagonal. Thus both of them have determinant 1. As $R$ relates them, it also has determinant 1.

Assume that the claim is true for all values of $m'$ up to, but not including $m \geq 2$. Let the matrix $M$ for $m'$ be denoted by $M_{m'}$ and $R$ for $m'$ be denoted by $R_{m'}$. Then, $[M_m]_F = [M_{m-1}]_F \otimes [M_1]_F$. Let $R_m := R_{m-1} \otimes R_1$. Then, $R_m[M_m]_F = (R_{m-1} \otimes R_1)\left([M_{m-1}]_F \otimes [M_1]_F\right) = (R_{m-1}[M_{m-1}]_F) \otimes (R_1[M_1]_F)$. Thus, the claim that $R_m[M_m]_F$ has the desired structure follows from the induction hypothesis. Further, as both $R_{m-1}$ and $R_1$ have determinant 1, $\det(R_m) = 1$. □

Because of the above claim, showing that $R[M]_F DN$ is invertible would suffice. Just like we did with $M$, we also impose the order $\prec$ on the columns of $R[M]_F$ that are indexed by $\mathbf{e} \in \{0, \ldots, d\}^m$. Recall that the rows of $R[M]_F$ and the columns of $N$ are indexed by $\mathbf{b} \in F$. We order these indices as follows: we keep the indices $\mathbf{b} \in F \setminus \mathcal{B}$ before $\mathbf{b}_1, \ldots, \mathbf{b}_p$. We will treat $\mathbf{r}^{-\mathbf{e}}$ as a monomial in $(-r_1)^{-1}, \ldots, (-r_m)^{-1}$ "variables" and impose the order $\prec$ on the monomials in these variables. Let $A := \{\mathbf{b} : \mathbf{b} \in F \setminus \mathcal{B}\} \cup \left\{\mathbf{b}'_1, \ldots, \mathbf{b}'_p\right\}$; notice that $|A| = |F|$. Also, the elements of $A$ are ordered as the elements of $F$ but with $\mathbf{b}'_k$ replacing $\mathbf{b}_k$ for $k \in [p]$. Then, from the Cauchy-Binet formula and the construction of the matrix $N$,

$\det(R[M]_F DN)$ equals

$$\det\left([R[M]_F]_{\bullet,A}\right)[N]_A \cdot \prod_{\mathbf{e}\in A} \mathbf{r}^{-\mathbf{e}} + \text{ lower order monomials in the } (-r_1)^{-1},\ldots,(-r_m)^{-1} \text{ variables.}$$

Here $[R[M]_F]_{\bullet,A}$ denotes the restriction of $R[M]_F$ to the columns indexed by $\mathbf{e} \in A$, and $[N]_A$ denotes the restriction of $N$ to the rows indexed by $\mathbf{e} \in A$. Thus to show that $R[M]_F DN$ (and therefore $[CMDN]_F$) is invertible, it is sufficient to prove the following two claims.

**Claim 3.5** *$[N]_A$ is an identity matrix.*

**Proof:** This basically follows from the construction of $N$: Consider a $\mathbf{b} \in F \setminus \mathcal{B}$. As $A$ does not contain any element of $\mathcal{B}$, the column of $[N]_A$ indexed by $\mathbf{b}$ has only one non-zero entry (which is 1) in the row indexed by $\mathbf{b}$. Similarly, the column of $[N]_A$ indexed by $\mathbf{b}_k$ for any $k \in [p]$ has only one non-zero entry (which is 1) in the row indexed by $\mathbf{b}'_k$. The claim then follows from the fact that the elements of $A$ are ordered as the elements of $F$ but with $\mathbf{b}'_k$ replacing $\mathbf{b}_k$ for all $k \in [p]$. □

**Claim 3.6** *The matrix $[R[M]_F]_{\bullet,A}$ is an upper triangular matrix with 1 or $-1$ entries on the diagonal.*

**Proof:** Consider the column of $[R[M]_F]_{\bullet,A}$ indexed by some $\mathbf{b} \in F \setminus \mathcal{B}$. From Claim 3.4, the only non-zero entry in this column is in the row indexed by $\mathbf{b}$ itself. Now consider a column of $[R[M]_F]_{\bullet,A}$ indexed by $\mathbf{b}'_k$ for some $k \in [p]$. From Claims 3.2 and 3.3, $(\mathbf{b}'_k)_{E_k} = (\mathbf{b}_k)_{E_k} \neq (\mathbf{b}_\ell)_{E_k}$ for all $\ell > k$. As every coordinate of $\mathbf{b}_k$ is non-zero, it follows from Claim 3.4 that the entry in the row indexed by $\mathbf{b}_\ell$ must be 0 for every $\ell > k$. Also, from Claim 3.3, $\mathbf{b}_k$ and $\mathbf{b}'_k$ agree at all coordinates $\mathbf{b}'_k$ is non-zero. So, from Claim 3.4, the entry in the row indexed by $\mathbf{b}_k$ must be non-zero. Also, recall from Claim 3.4 that the non-zero entries of $R[M]_F$ are either 1 or $-1$. The claim then follows from the fact that the elements of $A$ are ordered same as elements of $F$ but with $\mathbf{b}'_k$ replacing $\mathbf{b}_k$ for all $k \in [p]$. □

This finishes the proof of the lemma. □

Observe that $\det([CMDN]_F) \in \mathbb{F}[\mathbf{r}]$: Every entry of $[CMDN]_F$ is a $\mathbb{F}$-linear combination of some entries of the matrix $CMD$. The entry of $CMD$ indexed by $(\mathbf{b}, \mathbf{e})$ is $\binom{\mathbf{b}}{\mathbf{e}} \cdot \mathbf{r}^{\mathbf{b}} \cdot \mathbf{r}^{-\mathbf{e}}$, which is non-zero only if $b_i \geq e_i$ for all $i \in [m]$. In this case, $\mathbf{r}^{\mathbf{b}} \cdot \mathbf{r}^{-\mathbf{e}}$ is a monomial in the $\mathbf{r}$-variables. Thus, $\det([CMDN]_F)$ – which is a polynomial in the entries of $[CMDN]_F$ – is a polynomial in the $\mathbf{r}$-variables. This observation leads to the following corollary of the above lemma, which immediately gives a way to satisfy condition 1 of Lemma 3.1.

**Corollary 3.1** *Let $h(\mathbf{r}) := \det([CMDN]_F)$. Then, $\deg_{\mathbf{r}}(h(\mathbf{r})) \leq md^{m+1}$. Also, for every $\mathbf{b} \in F$,*

$$h(\mathbf{r}) \cdot v_{\mathbf{b}} \in \mathbb{F}[\mathbf{t}]\text{-span}\left\{v_{\mathbf{b}'} : \mathbf{b}' \in \{0,\ldots,d\}^m \text{ and Supp}\left(\mathbf{b}'\right) < m\right\}.$$

**Proof:** As mentioned in the previous paragraph, every entry of $[CMDN]_F$ is an $\mathbb{F}$-linear combination of the entries of $CMD$ which themselves are of the form $\binom{\mathbf{b}}{\mathbf{e}} \cdot \mathbf{r}^{\mathbf{b}} \cdot \mathbf{r}^{-\mathbf{e}}$. As, $\mathbf{b}, \mathbf{e} \in \{0,\ldots,d\}^m$ and $\mathbf{r}$ has $m$ variables, the degree of $\mathbf{r}^{\mathbf{b}} \cdot \mathbf{r}^{-\mathbf{e}}$ in the $\mathbf{r}$-variables is at most $md$. Since $[CMDN]_F$ is a $d^m \times d^m$ matrix, the degree of $\det([CMDN]_F)$ in the $\mathbf{r}$-variables is at most $md^{m+1}$.

$U = VCMD$ implies that $VCMDN = 0$. Let $V_F$ be the sub-vector of $V$ consisting solely of the entries indexed by $\mathbf{b} \in F$. As $VCMDN = 0$, every entry of $V_F\,[CMDN]_F$ is in

$$\mathbb{F}[\mathbf{t}]\text{-span}\left\{v_{\mathbf{b}'} : \mathbf{b}' \in \{0,\ldots,d\}^m \setminus F\right\} = \mathbb{F}[\mathbf{t}]\text{-span}\left\{v_{\mathbf{b}'} : \mathbf{b}' \in \{0,\ldots,d\}^m \text{ and Supp}(\mathbf{b}') < m\right\}.$$

So by multiplying $V_F\,[CMDN]_F$ by the adjoint of $[CMDN]_F$, we get that every entry of $V_F$ times $\det([CMDN]_F)$, i.e., $h(\mathbf{r}) \cdot v_{\mathbf{b}}$ where $\mathbf{b} \in F$ is in

$$\mathbb{F}[\mathbf{t}]\text{-span}\left\{v_{\mathbf{b}'} : \mathbf{b}' \in \{0,\ldots,d\}^m \text{ and Supp}(\mathbf{b}') < m\right\}.$$

□

The following claim about $h(\mathbf{r})$ gives us a way to satisfy condition 2 of Lemma 3.1.

**Claim 3.7** *The polynomial $h(\mathbf{r})$, when viewed as a polynomial in the $\mathbf{t}$-variables after setting $r_i = \ell_i(\mathbf{t})$, has a $\mathbf{t}$-monomial of support at most $m$.*

**Proof:** The polynomial $h(\mathbf{r}) = h(\ell_1(\mathbf{t}),\ldots,\ell_m(\mathbf{t})) \neq 0$ as $[CMDN]_F$ is an invertible matrix and $\ell_1,\ldots,\ell_m$ are $\mathbb{F}$-linearly independent. Then, as there are only $m$ $\mathbf{r}$-variables, the claim follows immediately from Observation 2.1. □

Thus, by substituting $\mathcal{G}_m^{SV}$ for $\mathbf{t}$, the polynomial $h(\mathbf{r})$ remains non-zero, satisfying condition 2. Note that the number of variables in $\mathcal{G}_m^{SV}$, i.e., $|\mathbf{z}| = 2m$ and its degree is $n$. We are now in a position to prove Theorem 3.2.

### 3.2.3 Proof of Theorem 3.2

Let $f = \mathbf{1}^T \cdot M_1(x_1)M_2(x_2)\cdots M_n(x_n) \cdot \mathbf{1}$ be a width-$w$ commutative ROABP having individual degree at most $d$; here $M_i \in \mathbb{F}^{w\times w}[x_i]$ for all $i \in [n]$. Also, let $F = M_1(x_1)M_2(x_2)\cdots M_n(x_n)$. For any $A \in \mathrm{GL}(n,\mathbb{F})$, let $g = f(A\mathbf{x})$ and $G = F(A\mathbf{x})$. Suppose that $A$ maps $x_i \mapsto \ell_i(\mathbf{x})$ and let $y_i = \ell_i(\mathbf{x})$ for all $i \in [n]$. Then, $g = \mathbf{1}^T \cdot M_1(y_1)M_2(y_2)\cdots M_n(y_n) \cdot \mathbf{1}$ and $G =$

$M_1(y_1)M_2(y_2)\cdots M_n(y_n)$. In Sections 3.2.1 and 3.2.2, we have shown that $G\left(\mathbf{x}+\mathcal{G}_m^{SV}\right)$ has support-$(m-1)$ rank concentration (for $m = 2\left\lceil \log w^2\right\rceil + 1$) over $\mathbb{F}(\mathbf{z})$ in the $\mathbf{y}$-variables; the $\mathbf{z}$-variables are the variables introduced by the $\mathcal{G}_m^{SV}$ generator. From Observation 2.4, if $g(\mathbf{x}) \neq 0$, then $g\left(\mathbf{x}+\mathcal{G}_m^{SV}\right)$, when viewed as a polynomial over $\mathbb{F}[\mathbf{z}]$ in the $\mathbf{y}$-variables[1], has a $\mathbf{y}$-monomial of support at most $m-1$. Let the $\mathbf{y}$-degree of this monomial be $D'$. As the individual degree of every $\mathbf{x}$-variable in $f$ is at most $d$, the individual degree of every $\mathbf{y}$-variable in $g$ is also at most $d$. Thus, $D' \leq (m-1)d$. As the homogeneous component of $g\left(\mathbf{x}+\mathcal{G}_m^{SV}\right)$ of $\mathbf{y}$-degree $D'$ is non-zero, the homogeneous component of $g\left(\mathbf{x}+\mathcal{G}_m^{SV}\right)$ (now viewed as polynomial over $\mathbb{F}[\mathbf{z}]$ in the $\mathbf{x}$-variables) of $\mathbf{x}$-degree $D'$ must also be non-zero, since $\ell_1,\ldots,\ell_n$ are linearly independent. This means that $g(\mathbf{x}+\mathcal{G}_m^{SV})$, when viewed as a polynomial over $\mathbb{F}[\mathbf{z}]$ in the $\mathbf{x}$-variables, has an $\mathbf{x}$-monomial of support (in fact, degree) at most $D' \leq (m-1)d$. Thus, $g\left(\mathcal{G}_{(m-1)d}^{SV}+\mathcal{G}_m^{SV}\right) \neq 0$. Now, it follows directly from the definition of the SV generator that $\mathcal{G}_{(m-1)d}^{SV}+\mathcal{G}_m^{SV} = \mathcal{G}_{m+(m-1)d}^{SV}$ and so $g\left(\mathcal{G}_{m+(m-1)d}^{SV}\right) \neq 0$. Replacing $m$ by its value $2\left\lceil \log w^2\right\rceil + 1$ proves the theorem. Note that the SV generator needs $|\mathbb{F}| \geq n$.

### 3.2.4 Proofs of Theorem 1.1

Let $f$ be an $n$-variate polynomial computed by a width-$w$ commutative ROABP of individual degree at most $d$, and $g \in \text{orb}\,(f)$. Then, from Theorem 3.2, $g\left(\mathcal{G}_{(2\lceil \log w^2\rceil (d+1)+1)}^{SV}\right) \neq 0$ whenever $g \neq 0$. Now, $\mathcal{G}_{(2\lceil \log w^2\rceil (d+1)+1)}^{SV}$ has $2\left(2\left\lceil \log w^2\right\rceil (d+1)+1\right)$ variables, and is of degree $n$. So $g\left(\mathcal{G}_{(2\lceil \log w^2\rceil (d+1)+1)}^{SV}\right)$ also has $2\left(2\left\lceil \log w^2\right\rceil (d+1)+1\right)$ variables. Since the individual degree of $f$ is at most $d$, the $\deg(f) = \deg(g) \leq nd$. So the degree of $g\left(\mathcal{G}_{(2\lceil \log w^2\rceil (d+1)+1)}^{SV}\right)$ is at most $n^2d$. Thus, as $|\mathbb{F}| > n^2d$, a hitting set for $g$ can be computed in time $\left(n^2d+1\right)^{2\left(2\left\lceil \log w^2\right\rceil (d+1)+1\right)} = (nd)^{O(d\log w)}$.

### 3.2.5 Proofs of Theorem 1.2

Let $f$ be an $n$-variate polynomial such that $f = \sum_{i\in[s]}\prod_{j\in[n]} f_{i,j}(x_j)$, where each $f_{i,j}(x_j)$ is a univariate polynomial in $x_j$. For all $j \in [n]$, define the matrix $M_j = \text{diag}(f_{1,j},\ldots,f_{s,j})$. Then $f = \mathbf{1}^T \cdot M_1(x_1)M_2(x_2)\cdots M_n(x_n)\cdot \mathbf{1}$. Moreover, as the matrices $M_1,\ldots,M_n$ are diagonal, they commute with each other. Hence, $f$ is computed by a width-$s$ commutative ROABP and the theorem follows from Theorem 1.1.

[1]This we can do as $g\left(\mathbf{x}+\mathcal{G}_m^{SV}\right) = \mathbf{1}^T \cdot G\left(\mathbf{x}+\mathcal{G}_m^{SV}\right)\cdot \mathbf{1}$, and $G\left(\mathbf{x}+\mathcal{G}_m^{SV}\right)$ can be viewed as a polynomial over $\mathbb{A}[\mathbf{z}]$ in the $\mathbf{y}$-variables.

### 3.2.6 Proof of Theorem 3.1

Let $f = \sum_{j\in[s]} c_j x_1^{e_{j,1}} \cdots x_n^{e_{j,n}}$ be a sparse polynomial, where $c_j \in \mathbb{F}$ for $j \in [s]$. Observe that $f$ can be computed by a commutative ROABP as follows: Let $M_1(x_1) := \mathrm{diag}(c_1 x_1^{e_{1,1}}, \ldots c_s x_1^{e_{s,1}})$ and, for $2 \leq i \leq n$, let $M_i(x_i) := \mathrm{diag}(x_i^{e_{1,i}}, \ldots x_i^{e_{s,i}})$. Then, $f = \mathbf{1}^T \cdot M_1(x_1) \cdots M_n(x_n) \cdot \mathbf{1}$. Notice that, as all matrices $M_i$ are diagonal, it is a commutative ROABP and its width is $s$. Thus, if the individual degree of $f$ is at most $d$, then Theorem 1.1 implies a hitting set that can be computed in time $(nd)^{O(d \log s)}$.

A parallel and independent work [MS21] shows that for the case of sparse polynomials the low individual degree restriction can be removed. They prove the following theorem.

**Theorem 3.3 ([MS21])** *Let $f$ be an $n$-variate, $s$-sparse polynomial of degree $d$ and $g \in \mathrm{orb}(f)$. Also, let $|\mathbb{F}| > nd$ and $\mathrm{char}(\mathbb{F}) = 0$ or $> d$. Then, $g \neq 0$ implies $g\left(\mathcal{G}^{SV}_{\lceil \log s \rceil + 1}\right) \neq 0$. In fact, if $g$ is not a constant, then neither is $g\left(\mathcal{G}^{SV}_{\lceil \log s \rceil + 1}\right)$.*

The above theorem yields an $(nd)^{\log s}$ size hitting set for the orbits of $n$-variate, degree-$d$ polynomials. We will make use of the above theorem in Sections 4.3 and 4.4 to prove Theorems 1.4 and 1.5, respectively.

## 3.3 Hitting sets for the orbits of multilinear constant-width ROABPs

**The strategy.** Theorem 1.3 is proved by combining the rank concentration by translation technique of Agrawal, Saha, and Saxena [ASS13] with the merge-and-reduce idea from Forbes and Shpilka [FS13], Forbes, Saptharishi, and Shpilka [FSS14]. Let

$$f = \mathbf{1}^T \cdot M_1(x_1) M_2(x_2) \cdots M_n(x_n) \cdot \mathbf{1}$$

be a multilinear, width-$w$ ROABP; here $M_i(x_i) \in \mathbb{F}^{w\times w}[x_i]$ for all $i \in [n]$. Also, let $F = M_1(x_1)M_2(x_2)\cdots M_n(x_n)$. For any $A \in \mathrm{GL}(n, \mathbb{F})$, let $g = f(A\mathbf{x})$ and $G = F(A\mathbf{x})$. For $i \in [n]$, suppose that $A$ maps $x_i \mapsto \ell_i(\mathbf{x})$, where $\ell_i$ is a linear form, and let $y_i = \ell_i(\mathbf{x})$ and $\mathbf{y} = \{y_1, \ldots, y_n\}$. Then, $g = \mathbf{1}^T \cdot M_1(y_1)M_2(y_2)\cdots M_n(y_n)\cdot \mathbf{1}$ and $G = M_1(y_1)M_2(y_2)\cdots M_n(y_n)$. Much like in the case of commutative ROABPs, we show that if $g \neq 0$, then there exist explicit "low" degree polynomials $t_1(\mathbf{z}), \ldots, t_n(\mathbf{z})$, where $\mathbf{z}$ is a "small" set of variables such that $G(x_1 + t_1(\mathbf{z}), \ldots, x_n + t_n(\mathbf{z}))$ has "low" support rank concentration in the "$\mathbf{y}$-variables". While in the rank concentration argument for commutative ROABPs the $\mathbf{x}$-variables were

translated only once, here the translations can be thought of as happening sequentially and in stages. There will be $\lceil \log n \rceil$ stages with each stage also consisting of multiple translations. After the $p$-th stage, the product of any $2^p$ consecutive matrices in $G$ will have low support rank concentration in the $\mathbf{y}$-variables. Thus, after $\lceil \log n \rceil$ stages, we will have low support rank concentration in the $\mathbf{y}$-variables for $G(x_1 + t_1(\mathbf{z}), \ldots, x_n + t_n(\mathbf{z}))$. As in the case of commutative ROABPs, we show that $G(\mathbf{x} + \mathbf{t})$ has low support rank concentration if each polynomial in a certain collection of non-zero polynomials in the $\mathbf{t}$-variables is kept non-zero by the substitution $t_i \mapsto t_i(\mathbf{z})$. However, in this case, it is trickier to show that these polynomials have low support $\mathbf{t}$-monomials. We do this by arguing that each such polynomial can be expressed as a ratio of a polynomial that contains a low support $\mathbf{t}$-monomial and a product of linear forms in the $\mathbf{t}$-variables.

*Remark.* A quasi-polynomial time hitting set for general ROABPs was given by Agrawal, Gurjar, Korwar, and Saxena [AGKS15] using an elegant generalization of the monomial isolation method of Klivans and Spielman [KS01], namely the *basis isolation method*. As shown in Gurjar, Korwar, Saxena, and Thierauf [GKST17], Forbes, Ghosh, and Saxena [FGS18], designing a basis isolating weight assignment is a stronger objective than achieving rank concentration by translation. It is not immediately clear how to obtain efficient constructions of basis isolating weight assignments for the orbits of ROABPs, even under additional restrictions such as commutativity, constant-width or low individual degree. However, our work here shows that the weaker objective of rank concentration by translation can be achieved for the orbits of the above-mentioned subclasses of ROABPs.

**Notations and conventions.** Much like in the previous section, we will first translate the $\mathbf{x}$-variables by the $\mathbf{t}$-variables and then substitute the $\mathbf{t}$-variables by low degree polynomials in a small set of variables. We will translate the $\mathbf{x}$-variables by $\lceil \log n \rceil$ groups of $\mathbf{t}$-variables, $\mathbf{t}_1, \ldots, \mathbf{t}_{\lceil \log n \rceil}$. For all $p \in \lceil \log n \rceil$, the group $\mathbf{t}_p$ will have $\mu := w^2 + \lceil \log w^2 \rceil$ sub-groups of $\mathbf{t}$-variables, $\mathbf{t}_{p,1}, \ldots, \mathbf{t}_{p,\mu}$. For all $p \in \lceil \log n \rceil$ and $q \in [\mu]$, $\mathbf{t}_{p,q} := \{t_{p,q,1}, \ldots, t_{p,q,n}\}$. Thus, finally the translation will look like

$$x_i \to x_i + \sum_{\substack{p \in \lceil \log n \rceil, \\ q \in [\mu]}} t_{p,q,i}$$

for all $i \in [n]$. Finally, we will substitute the $\mathbf{t}$-variables as $t_{p,q,i} \mapsto s_{p,q} \cdot z_{p,q}^{\beta_{p,q}(i)}$, where $\beta_{p,q}(i)$ will be fixed later in the analysis. Let $r_{p,q,i} := \ell_i\left(\mathbf{t}_{p,q}\right)$; notice that for all $i \in [n]$, $y_i$ is

translated as

$$y_i \to y_i + \sum_{\substack{p\in\lceil \log n\rceil,\\ q\in[\mu]}} \ell_i\left(\mathbf{t}_{p,q}\right) = y_i + \sum_{\substack{p\in\lceil \log n\rceil,\\ q\in[\mu]}} r_{p,q,i}.$$

For the purpose of analysis, we will think of the translation as happening sequentially in the order $\mathbf{t}_{1,1},\ldots,\mathbf{t}_{1,\mu},\mathbf{t}_{2,1},\ldots,\mathbf{t}_{2,\mu},\ldots,\mathbf{t}_{n,1},\ldots\mathbf{t}_{n,\mu}$, i.e., we will first translate by $\mathbf{t}_{1,1}$, then by $\mathbf{t}_{1,2}$, and so on. Let us denote the order thus imposed on the set $\{(p,q) : p \in [\lceil \log n\rceil], q \in [\mu]\}$ by $\prec$.

For a set $S = \{i_1, i_2, \ldots, i_{|S|}\} \subseteq [n]$, where $i_1 < i_2 < \ldots < i_{|S|}$, the vector $(b_{i_1}, b_{i_2}, \ldots, b_{i_{|S|}})$ will be denoted by $(b_i : i \in S)$. Let $\mathrm{Supp}\,(\mathbf{b})$ denote the support of the vector $\mathbf{b}$ which is defined as the number of non-zero elements in it.

The inductive argument given on the next two subsections is inspired by the "merge-and-reduce" idea from [FS13, FSS14].

### 3.3.1 Low support rank concentration: an inductive argument

In this and the next sections, we will prove the following lemma. Let $\mathbb{A} := \mathbb{F}^{w\times w}$.

**Lemma 3.3** *There exist* $\left\{\beta_{p,q}(i) : p \in [\lceil \log n\rceil], q \in [\mu], i \in [n]\right\} \subset \mathbb{Z}_{\geq 0}$, *such that*

$$G\left(x_1 + \sum_{\substack{p\in\lceil \log n\rceil,\\ q\in[\mu]}} s_{p,q}\cdot z_{p,q}^{\beta_{p,q}(1)}, \ldots, x_n + \sum_{\substack{p\in\lceil \log n\rceil,\\ q\in[\mu]}} s_{p,q}\cdot z_{p,q}^{\beta_{p,q}(n)}\right),$$

*when treated as a polynomial in the* $\mathbf{y}$*-variables over* $\mathbb{A}[r_{p,q,i} : p \in [\lceil \log n\rceil], q \in [\mu], i \in [n]]$, *has support-*$\mu$ *rank concentration in the* $\mathbf{y}$*-variables over* $\mathbb{F}\left(s_{p,q}, z_{p,q} : p \in [\lceil \log n\rceil], q \in [\mu]\right)$. *Moreover,* $\left\{\beta_{p,q}(i) : p \in [\lceil \log n\rceil], q \in [\mu], i \in [n]]\right\}$ *can be found in time* $n^{O(w^4)}$ *and each* $\beta_{p,q}(i) \leq n^{O(w^4)}$.

We will prove this lemma by induction on $(p,q)$. Let us call

$$\left\{\beta_{p,q}(i) : p \in [\lceil \log n\rceil], q \in [\mu], i \in [n]]\right\}$$

*efficiently computable and good* if they can be found in time $n^{O(w^4)}$ and each $\beta_{p,q}(i) \leq n^{O(w^4)}$. Precisely, the induction hypothesis is as follows.

**Induction hypothesis.** Just before translating by $\mathbf{t}_{p^*,q^*}$-variables, we will assume that the following is true: there exist efficiently computable and good $\left\{\beta_{p,q}(i) : (p,q) \prec (p^*,q^*)\right\}$ such that the product of any $2^{p^*}$ consecutive matrices in

$$G\left(x_1 + \sum_{(p,q)\prec(p^*,q^*)} s_{p,q}\cdot z_{p,q}^{\beta_{p,q}(1)}\,,\ldots,\,x_n + \sum_{(p,q)\prec(p^*,q^*)} s_{p,q}\cdot z_{p,q}^{\beta_{p,q}(n)}\right)$$

has support-$(2\mu-(q^*-1))$ rank concentration in the $\mathbf{y}$-variables over

$$\mathbb{F}\left(s_{p,q}, z_{p,q} : (p,q)\prec(p^*,q^*)\right).$$

**Base case.** In the base case, $(p^*,q^*)=(1,1)$. Observe that we can assume that $w\geq 2$; if $w=1$, then $g$ is a product of univariates and the existence of a polynomial time hitting set follows from Observation 2.1. For any $w\geq 2$, $2\leq 2\mu$. As a product of any two consecutive matrices in $G$ has support $2\mu$ rank concentration in the $\mathbf{y}$-variables over $\mathbb{F}$, the base case is satisfied.

**Induction step.** We need to show that there exist efficiently computable and good $\{\beta_{p^*,q^*}(i) : i\in[n]\}$ such that after translating by $\mathbf{t}_{p^*,q^*}$ and substituting $t_{p^*,q^*,i}\to s_{p^*,q^*}\cdot z_{p^*,q^*}^{\beta_{p^*,q^*}(i)}$, the product of any $2^{p^*}$ consecutive matrices in

$$G\left(x_1 + \sum_{(p,q)\preccurlyeq(p^*,q^*)} s_{p,q}\cdot z_{p,q}^{\beta_{p,q}(1)}\,,\ldots,\,x_n + \sum_{(p,q)\preccurlyeq(p^*,q^*)} s_{p,q}\cdot z_{p,q}^{\beta_{p,q}(n)}\right)$$

has support-$(2\mu-q^*)$ rank concentration in the $\mathbf{y}$-variables over $\mathbb{F}\left(s_{p,q}, z_{p,q} : (p,q)\preccurlyeq(p^*,q^*)\right)$. If $q^*<\mu$, then this would mean that the induction hypothesis holds immediately before translation by $\mathbf{t}_{p^*,q^*+1}$. On the other hand, if $q^*=\mu$, then the following easy-to-verify observation implies that the induction hypothesis holds immediately before translation by $\mathbf{t}_{p^*+1,1}$.

**Observation 3.2** *Suppose that $\{\beta_{p,q}(i) : (p,q)\preccurlyeq(p^*,\mu)\}$ are such that the product of any $2^{p^*}$ consecutive matrices in*

$$G\left(x_1 + \sum_{(p,q)\preccurlyeq(p^*,\mu)} s_{p,q}\cdot z_{p,q}^{\beta_{p,q}(1)}\,,\ldots,\,x_n + \sum_{(p,q)\preccurlyeq(p^*,\mu)} s_{p,q}\cdot z_{p,q}^{\beta_{p,q}(n)}\right)$$

*has support-$\mu$ rank concentration in the $\mathbf{y}$-variables over $\mathbb{F}\left(s_{p,q}, z_{p,q} : (p,q)\preccurlyeq(p^*,\mu)\right)$. Then the product of any $2^{p^*+1}$ consecutive matrices in*

$$G\left(x_1 + \sum_{(p,q)\preccurlyeq(p^*,\mu)} s_{p,q}\cdot z_{p,q}^{\beta_{p,q}(1)}\,,\ldots,\,x_n + \sum_{(p,q)\preccurlyeq(p^*,\mu)} s_{p,q}\cdot z_{p,q}^{\beta_{p,q}(n)}\right)$$

*has support-$2\mu$ rank concentration in the* $\mathbf{y}$*-variables over* $\mathbb{F}\left(s_{p,q}, z_{p,q} : (p,q) \preccurlyeq (p^*,\mu)\right)$.

**Simplifying notations for the ease of exposition.** By focusing on the induction step, we will henceforth denote $\mathbb{F}\left(s_{p,q}, z_{p,q} : (p,q) \prec (p^*,q^*)\right)$ by $\mathbb{F}$, and for all $i \in [n]$,

$$M_i\left(y_j + \sum_{(p,q)\prec(p^*,q^*)} \ell_i\left(s_{p,q}\cdot z_{p,q}^{\beta_{p,q}(1)}, \ldots, s_{p,q}\cdot z_{p,q}^{\beta_{p,q}(n)}\right)\right)$$

by $M_i(y_i)$, $t_{p^*,q^*,i}$ by $t_i$, $r_{p^*,q^*,i}$ by $r_i$, $s_{p^*,q^*}$ by $s$, $z_{p^*,q^*}$ by $z$ and $\beta_{p^*,q^*}(i)$ by $\beta(i)$.

Without loss of generality, we shall consider the product $M_1(y_1+r_1)\cdots M_m(y_n+r_m)$ of the first $m = 2^{p^*}$ matrices. Our goal is to show that there exist efficiently computable and good $\{\beta(i) : i \in [m]\}$ such that after substituting $t_i \to s\cdot z^{\beta(i)}$, the above product has support-$(2\mu - q^*)$ rank concentration in the $\mathbf{y}$-variables over $\mathbb{F}(s,z)$ *assuming that* $M_1(y_1)\cdots M_m(y_m)$ *has support-*$(2\mu-(q^*-1))$ *rank concentration in the* $\mathbf{y}$*-variables over* $\mathbb{F}$.

### 3.3.2 Details of the induction step

**Recalling some notations.** Before we show how to achieve rank concentration, let us recall some notations defined in Section 3.2. While in Section 3.2, the individual degree is $d$, here the individual degree is 1 and so, we modify the definitions accordingly. $\mathbb{A}$ is used to denote the matrix algebra $\mathbb{F}^{w\times w}$. For $i \in [m]$, $M_i(y_i) = \sum_{e_i=0}^{1} u_{i,e_i} y_i^{e_i}$, where $u_{i,e_i} \in \mathbb{A}$ and $M_i(y_i + r_i) = \sum_{b_i=0}^{1} v_{i,b_i} y_i^{b_i}$, where $v_{i,b_i} \in \mathbb{A}[r_i] \subset \mathbb{A}[\mathbf{t}]$. For $\mathbf{b} = (b_1,\ldots,b_m)$ and $\mathbf{e} = (e_1,\ldots,e_m)$ in $\{0,1\}^m$, $\binom{\mathbf{b}}{\mathbf{e}} := \prod_{i\in[m]}\binom{b_i}{e_i}$. Also, $v_{\mathbf{b}} := \prod_{i\in[m]} v_{i,b_i}$ and $u_{\mathbf{e}} := \prod_{i\in[m]} u_{i,e_i}$. Moreover, $\mathbf{r} := (-r_1,\ldots,-r_m)$, $\mathbf{r}^{\mathbf{b}} := \prod_{i\in[m]}(-r_i)^{b_i}$ and $\mathbf{r}^{-\mathbf{e}} := \prod_{i\in[m]}(-r_i)^{-e_i}$. Let $\mathbf{t} := (t_1,\ldots,t_n)$.

The following vectors and matrices are defined by fixing an arbitrary order on the elements of $\{0,1\}^m$. $V := \left(v_{\mathbf{b}} : \mathbf{b}\in\{0,1\}^m\right)$ and $U := \left(u_{\mathbf{e}} : \mathbf{e}\in\{0,1\}^m\right)$; $V$ is a row vector in $\mathbb{A}[\mathbf{r}]^{2^m}$ whereas $U$ is a row vector in $\mathbb{A}^{2^m}$. Both $C := \mathrm{diag}(\mathbf{r}^{\mathbf{b}} : \mathbf{b}\in\{0,1\}^m)$ and $D := \mathrm{diag}(\mathbf{r}^{-\mathbf{e}} : \mathbf{e}\in\{0,1\}^m)$ are $2^m\times 2^m$ diagonal matrices. Finally, $M$ is a $2^m\times 2^m$ numeric matrix whose rows and columns were indexed by $\mathbf{b}\in\{0,1\}^m$ and $\mathbf{e}\in\{0,1\}^m$, respectively. The entry of $M$ indexed by $(\mathbf{b},\mathbf{e})$ contains $\binom{\mathbf{b}}{\mathbf{e}}$. The proof of the following transfer equation is same as the proof of Claim 3.1.

**Claim 3.8** *Let* $U$, $V$, $C$, $M$ *and* $D$ *be as defined above. Then,* $U = VCMD$.

Let $F := \left\{\mathbf{b}\in\{0,1\}^m : \mathrm{Supp}(\mathbf{b}) > 2\mu - q^*\right\}$.[1] Also, recall that the the *null space* of $U$ is

[1]There is a slight overloading of notation here: We have used $F$ before at the beginning of Section 3.3 to denote the product $M_1(x_1)M_2(x_2)\cdots M_n(x_n)$. However, since all our arguments involve only $G = F(A\mathbf{x})$ and not $F$, we would use $F$ in this section to denote the set that is mentioned here.

the set of all vectors $\left(n_{\mathbf{e}} : \mathbf{e} \in \{0,1\}^m\right) \in \mathbb{F}^{2^m}$ for which $\sum_{\mathbf{e}\in\{0,1\}^m} n_{\mathbf{e}} u_{\mathbf{e}} = 0$. We have the following lemma.

**Lemma 3.4** *There are vectors $\{\mathbf{n_b} : \mathbf{b} \in F\}$ in the null space of $U$ such that the following holds: Let $N$ be the $2^m \times |F|$ matrix whose rows are indexed by $\mathbf{e} \in \{0,1\}^m$ and whose columns are indexed by $\mathbf{b} \in F$ and whose column indexed by $\mathbf{b}$ is $\mathbf{n_b}$. Then, the square matrix $[CMDN]_F$ is invertible, where $[CMDN]_F$ is the sub-matrix of $CMDN$ consisting of only those rows of $CMDN$ that are indexed by $F$. Also, $\det\left([CMDN]_F\right) \in \mathbb{F}[\mathbf{r}] \subset \mathbb{F}[\mathbf{t}]$ can be expressed as the ratio of a polynomial in $\mathbb{F}[\mathbf{t}]$ that contains a monomial of degree at most $2w^2\mu$ in the $\mathbf{t}$-variables and a product of linear forms in $\mathbb{F}[\mathbf{t}]$.*

The proof of this lemma is mostly similar to the proof of Lemma 3.2. However we need to do a little more work since now we also want to prove that $\det\left([CMDN]_F\right)$ has the desired structure.

**Proof:** The entries of $U$, the columns of $M$, the rows and columns of $D$, and the rows of $N$ are indexed by $\mathbf{e} \in \{0,1\}^m$. Impose the degree lexicographic order, denoted by $\prec_{\text{dlex}}$, on the indices $\mathbf{e} \in \{0,1\}^m$ of $U$ and the other three matrices[1]. Pick the *minimal* basis of the space spanned by the entries of $U$ according to this order, i.e., consider the entries of $U$ in the order dictated by $\prec_{\text{dlex}}$ while forming the basis. Let $\mathcal{B} := \{\mathbf{e} \in \{0,1\}^m : u_{\mathbf{e}} \text{ is in the minimal basis of } U \text{ w.r.t. } \prec_{\text{dlex}}\}$.

**Observation 3.3** *By the induction hypothesis, for every $\mathbf{e} \in F \cap \mathcal{B}$, $\text{Supp}(\mathbf{e}) = 2\mu - (q^* - 1)$.*

**Construction of the matrix $N$.** The columns of $N$ are indexed by $\mathbf{b} \in F$. We will now specify a set of column vectors $\{\mathbf{n_b} : \mathbf{b} \in F\}$ in the null space of $U$ such that the column of $N$ indexed by $\mathbf{b} \in F$ is $\mathbf{n_b}$. There are two cases for $\mathbf{b} \in F$:

**Case 1: $\mathbf{b} \in F \setminus \mathcal{B}$.** In this case, $u_{\mathbf{b}}$ is dependent on $\{u_{\mathbf{e}} : \mathbf{e} \in \mathcal{B} \text{ and } \mathbf{e} \prec_{\text{dlex}} \mathbf{b}\}$. Pick this dependence vector as $\mathbf{n_b}$.
**Case 2: $\mathbf{b} \in F \cap \mathcal{B}$.** Let there be $p$ such $\mathbf{b}, \mathbf{b}_1, \ldots, \mathbf{b}_p$, where $p \leq |\mathcal{B}| \leq w^2$. For a set $E \subseteq [m]$ and $\mathbf{b} \in \{0,1\}^m$, let $(\mathbf{b})_E$ denote the vector obtained by projecting $\mathbf{b}$ to the coordinates in $E$. Roughly speaking, the following claim says that each of these $p$ vectors has a "small signature" that differentiates it from the other $p-1$ vectors.

**Claim 3.9** *There exist sets $E_1, \ldots, E_p \subseteq [m]$, each of size $w^2 - 1$ such that for all $k \in [p]$,*

[1] by identifying $\mathbf{e}$ with an $m$-variate monomial.

1. $\text{Supp}\left((\mathbf{b}_k)_{E_k}\right) = w^2 - 1$,

2. $(\mathbf{b}_k)_{E_k} \neq (\mathbf{b}_\ell)_{E_k}\ \forall \ell \neq k$.

**Proof:** For $k \in [p]$, let $\mathcal{S}(\mathbf{b}_k)$ be the set of coordinates where $\mathbf{b}_k$ is non-zero. Fix any $k \in [p]$. Notice that $\text{Supp}(\mathbf{b}_k) = |\mathcal{S}(\mathbf{b}_k)| = 2\mu - (q^* - 1) \geq \mu + 2 = w^2 + \lceil \log w^2 \rceil + 2$. For $\ell \neq k$, as $\text{Supp}(\mathbf{b}_k) = \text{Supp}(\mathbf{b}_\ell)$ and $\mathbf{b}_k \neq \mathbf{b}_\ell$, there must exist an $i_\ell \in \mathcal{S}(\mathbf{b}_k)$, such that the $i_\ell$-th coordinate of $\mathbf{b}_k$ and $\mathbf{b}_\ell$ are distinct. Put all such $i_\ell$ for $\ell \neq k$ in $E_k$. If $|E_k|$ is still less than $w^2 - 1$, then arbitrarily put some more elements in $E_k$ from $\mathcal{S}(\mathbf{b}_k)$ so that $|E_k| = w^2 - 1$. This can be done as $\mathcal{S}(\mathbf{b}_k)$ is sufficiently large. □

As before, we will call $E_k$ the signature of $\mathbf{b}_k$. The following claim tells us that for each vector $\mathbf{b}_k$, there is a vector that is not in $\mathcal{B}$ and has support less than $2\mu - (q^* - 1)$, but agrees with $\mathbf{b}_k$ on its signature and so in some sense can be used as a proxy for $\mathbf{b}_k$.

**Claim 3.10** *For every $k \in [p]$, there exists a vector $\mathbf{b}'_k \in \{0,1\}^m \setminus (F \cup \mathcal{B})$ such that $(\mathbf{b}'_k)_{E_k} = (\mathbf{b}_k)_{E_k}$ and also $\mathbf{b}'_k$ and $\mathbf{b}_k$ agree on all locations where $\mathbf{b}'_k$ is non-zero.*

**Proof:** Similar to the proof of Claim 3.3. □

We will now use the above two claims to construct $\mathbf{n}_{\mathbf{b}_k}$ for all $k \in [p]$. We will use $\mathbf{b}'_k$ from Claim 3.10 as a proxy for $\mathbf{b}_k$. Notice that $u_{\mathbf{b}'_k}$ is dependent on $\left\{u_{\mathbf{e}} : \mathbf{e} \in \mathcal{B} \text{ and } \mathbf{e} \prec_{\text{dlex}} \mathbf{b}'_k\right\}$. Let this dependence vector be $\mathbf{n}_{\mathbf{b}_k}$. This completes the construction of $N$. We will now show that $[CMDN]_F$ is invertible. In fact, we will show that $\det\left([CMDN]_F\right)$ is the ratio of a polynomial in $\mathbb{F}[\mathbf{t}]$ which contains a monomial of degree at most $2w^2\mu$ and a product of a bunch of non-zero linear forms in $\mathbb{F}[\mathbf{t}]$.

**$[CMDN]_F$ is invertible.** Let $[M]_F$ be the restriction of $M$ to the rows indexed by $F$, and $[C]_F$ the restriction of $C$ to the rows and columns indexed by $F$.

**Observation 3.4** *The matrix $[M]_F$ has the following structure: The rows of $[M]_F$ are indexed by $\mathbf{b} = (b_1, \ldots, b_m) \in F$ and its columns by $\mathbf{e} = (e_1, \ldots, e_m) \in \{0,1\}^m$. Its entry indexed by $(\mathbf{b}, \mathbf{e})$ is non-zero if and only if for all $i \in [m]$, $b_i = e_i$ if $e_i \neq 0$. All non-zero entries are 1.*

We order the indices $\mathbf{b} \in F$ as follows: Let $F_0 := \{\mathbf{b} \in F : \text{Supp}(\mathbf{b}) > 2\mu - (q^* - 1)\}$ and $F_1 := \{\mathbf{b} \in F : \text{Supp}(\mathbf{b}) = 2\mu - (q^* - 1)\}$. We first keep the $\mathbf{b} \in F_0$ in (descending) degree

lexicographic order[1], followed by $\mathbf{b} \in F_1 \setminus \mathcal{B}$ in (reverse) lexicographic order[2], and then $\mathbf{b}_1, \ldots, \mathbf{b}_p$. Also, let $A := (F \setminus \mathcal{B}) \uplus \left\{\mathbf{b}'_1, \ldots, \mathbf{b}'_p\right\}$. Notice that $|A| = |F|$. Also, the elements of $A$ are ordered as the elements of $F$ but with $\mathbf{b}'_k$ replacing $\mathbf{b}_k$ for $k \in [p]$. For any $S \subseteq \{0,1\}^m$ of size $|S| = |F|$, let $[M]_{F,S}$ denote the restriction of $[M]_F$ to the columns indexed by $\mathbf{e} \in S$, and $[N]_S$ denote the restriction of $N$ to the rows indexed by $\mathbf{e} \in S$. Now,

$$
\begin{aligned}
&\det([CMDN]_F) \\
&= \det([C]_F)\det([M]_F DN) \\
&= \prod_{\mathbf{b}\in F} \mathbf{r}^{\mathbf{b}} \cdot \left( \sum_{\substack{S\subseteq\{0,1\}^m \\ |S|=|F|}} \det\left([M]_{F,S}\right) \cdot \det([N]_S) \cdot \prod_{\mathbf{e}\in S} \mathbf{r}^{-\mathbf{e}} \right) \\
&= \prod_{\mathbf{b}\in F} \mathbf{r}^{\mathbf{b}} \cdot \left( \sum_{\substack{S\subseteq A\uplus\mathcal{B} \\ |S|=|F|}} \det\left([M]_{F,S}\right) \cdot \det([N]_S) \cdot \prod_{\mathbf{e}\in S} \mathbf{r}^{-\mathbf{e}} \right) \\
&= \prod_{\mathbf{b}\in F} \mathbf{r}^{\mathbf{b}} \cdot \left( \sum_{\substack{S\subseteq A\uplus\mathcal{B} \\ |S|=|F|}} \det\left([M]_{F,S}\right) \cdot \det([N]_S) \cdot \prod_{\mathbf{e}\in S\cap A} \mathbf{r}^{-\mathbf{e}} \cdot \prod_{\mathbf{e}\in S\cap\mathcal{B}} \mathbf{r}^{-\mathbf{e}} \right) \\
&= \prod_{\mathbf{b}\in F} \mathbf{r}^{\mathbf{b}} \cdot \prod_{\mathbf{e}\in A\uplus\mathcal{B}} \mathbf{r}^{-\mathbf{e}} \cdot \left( \sum_{\substack{S\subseteq A\uplus\mathcal{B} \\ |S|=|F|}} \det\left([M]_{F,S}\right) \cdot \det([N]_S) \cdot \prod_{\mathbf{e}\in A\setminus S} \mathbf{r}^{\mathbf{e}} \cdot \prod_{\mathbf{e}\in\mathcal{B}\setminus S} \mathbf{r}^{\mathbf{e}} \right),
\end{aligned}
$$

where the second equality follows from the Cauchy-Binet formula and the third equality from the fact that for any $S \not\subseteq A \uplus \mathcal{B}$, $\det([N]_S) = 0$. Now, notice that $\prod_{\mathbf{b}\in F} \mathbf{r}^{\mathbf{b}} \cdot \prod_{\mathbf{e}\in A\uplus\mathcal{B}} \mathbf{r}^{-\mathbf{e}}$ is the reciprocal of a product of non-zero linear forms in $\mathbf{t}$-variables, as $F \subseteq A \uplus \mathcal{B}$. We shall now prove that

$$
\sum_{\substack{S\subseteq A\uplus\mathcal{B} \\ |S|=|F|}} \det\left([M]_{F,S}\right) \cdot \det([N]_S) \cdot \prod_{\mathbf{e}\in A\setminus S} \mathbf{r}^{\mathbf{e}} \cdot \prod_{\mathbf{e}\in\mathcal{B}\setminus S} \mathbf{r}^{\mathbf{e}} \tag{3.3}
$$

has a $\mathbf{t}$-monomial of degree at most $w^2(2\mu - (q^* - 1))$.

**Claim 3.11** *$[N]_A$ is an identity matrix.*

[1] i.e., $\mathbf{b}$ comes before $\hat{\mathbf{b}}$ if $\mathrm{Supp}(\mathbf{b}) > \mathrm{Supp}(\hat{\mathbf{b}})$, or if $\mathrm{Supp}(\mathbf{b}) = \mathrm{Supp}(\hat{\mathbf{b}})$ and $\hat{\mathbf{b}} \prec_{lex} \mathbf{b}$.

[2] i.e., $\mathbf{b}$ comes before $\hat{\mathbf{b}}$ if $\hat{\mathbf{b}} \prec_{lex} \mathbf{b}$.

**Proof:** Same as that of Claim 3.5. □

**Claim 3.12** *The matrix $[M]_{F,A}$ is an upper triangular matrix with ones on the diagonal.*

**Proof:** Consider the column of $[M]_{F,A}$ indexed by some $\mathbf{b} \in F \setminus \mathcal{B}$. Because of the way we have ordered the elements in $F$ and $A$, it follows from Observation 3.4, the only non-zero entries in this column are in and above the row indexed by $\mathbf{b}$. Now consider a column of $[M]_{F,A}$ indexed by $\mathbf{b}'_k$ for some $k \in [p]$. From Claims 3.9 and 3.10, $(\mathbf{b}'_k)_{E_k} = (\mathbf{b}_k)_{E_k} \neq (\mathbf{b}_\ell)_{E_k}$ for all $\ell \neq k$. As every coordinate of $(\mathbf{b}_k)_{E_k}$ is non-zero, it follows from Observation 3.4 that the entry in the row indexed by $\mathbf{b}_\ell$ must be 0 for every $\ell \neq k$. Also, from Claim 3.10, as $\mathbf{b}_k$ and $\mathbf{b}'_k$ agree at all coordinates $\mathbf{b}'_k$ is non-zero. So, from Observation 3.4, the entry in the row indexed by $\mathbf{b}_k$ must be non-zero. Also, recall from Observation 3.4 that the non-zero entries of $[M]_F$ are ones. The claim then follows from the fact that the elements of $A$ are ordered as that of $F$ but with $\mathbf{b}'_k$ replacing $\mathbf{b}_k$ for $k \in [p]$. □

**Claim 3.13** $\det([M]_{F,A}) \cdot \det([N]_A) \cdot \prod_{\mathbf{e} \in \mathcal{B} \setminus A} \mathbf{r}^{\mathbf{e}} = \prod_{\mathbf{e} \in \mathcal{B}} \mathbf{r}^{\mathbf{e}} \neq 0$ *and has* $\mathbf{t}$*-degree at most* $2w^2\mu$.

**Proof:** $\det([M]_{F,A}) \cdot \det([N]_A) \cdot \prod_{\mathbf{e} \in \mathcal{B} \setminus A} \mathbf{r}^{\mathbf{e}} = \prod_{\mathbf{e} \in \mathcal{B}} \mathbf{r}^{\mathbf{e}} \neq 0$ follows from Claims 3.11 and 3.12 and the fact that $A \cap \mathcal{B}$ is empty. For every $\mathbf{e} \in \mathcal{B}$, $\deg_{\mathbf{t}}(\mathbf{r}^{\mathbf{e}}) \leq 2\mu - (q^* - 1)$. So, $\deg_{\mathbf{t}}\left(\prod_{\mathbf{e} \in \mathcal{B}} \mathbf{r}^{\mathbf{e}}\right) \leq w^2 \cdot (2\mu - (q^* - 1)) \leq 2w^2\mu$, as $|\mathcal{B}| \leq w^2$. □

**Claim 3.14** *For any $S \subseteq A \uplus \mathcal{B}$ such that $|S| = |F|$ and $\det([N]_S)$ is non-zero, there is a one-to-one correspondence between $A \setminus S$ and $S \cap \mathcal{B}$ such that if $\mathbf{e} \in A \setminus S$ corresponds to $\mathbf{e}' \in S \cap \mathcal{B}$, then $\mathbf{e}' \prec_{\text{dlex}} \mathbf{e}$.*

**Proof:** As $\det([N]_S) \neq 0$, there must be a one-to-one correspondence between the rows and columns of $[N]_S$ such that if the column indexed by $\mathbf{b} \in F$ corresponds to a row indexed by $\mathbf{e} \in S$, then the $(\mathbf{e}, \mathbf{b})$-th entry of $[N]_S$ must be non-zero. Obtain a one-to-one correspondence between $A$ and $S$ from the above correspondence by replacing $\mathbf{b}_k$ with $\mathbf{b}'_k$ for all $k \in [p]$. Notice that, if $\mathbf{e} \in A$ corresponds to $\mathbf{e}'$ in $S$, then either $\mathbf{e}' \prec_{\text{dlex}} \mathbf{e}$ or $\mathbf{e}' = \mathbf{e}$. Now, removing $A \cap S$ from $A$ yields $A \setminus S$, and removing $A \cap S$ from $S$ yields $S \cap \mathcal{B}$. So the correspondence between $A$ and $S$ yields the desired correspondence between $A \setminus S$ and $S \cap \mathcal{B}$. □

The above claim implies that for every $S \in A \uplus \mathcal{B}$ of size $|F|$, either $\det([M]_{F,S}) \cdot \det([N]_S) \cdot \prod_{\mathbf{e} \in A \setminus S} \mathbf{r}^{\mathbf{e}} \cdot \prod_{\mathbf{e} \in \mathcal{B} \setminus S} \mathbf{r}^{\mathbf{e}}$ is 0, or $\prod_{\mathbf{e} \in \mathcal{B}} \mathbf{r}^{\mathbf{e}} \prec_{\text{dlex}} \prod_{\mathbf{e} \in A \setminus S} \mathbf{r}^{\mathbf{e}} \cdot \prod_{\mathbf{e} \in \mathcal{B} \setminus S} \mathbf{r}^{\mathbf{e}}$. Hence, $\prod_{\mathbf{e} \in \mathcal{B}} \mathbf{r}^{\mathbf{e}}$ is the smallest $\mathbf{r}$-monomial in the polynomial given in (3.3) w.r.t. $\prec_{\text{dlex}}$ order, and so, the homogeneous component of this polynomial that has the same $\mathbf{r}$-degree as that of $\prod_{\mathbf{e} \in \mathcal{B}} \mathbf{r}^{\mathbf{e}}$ survives.

Now, from Claim 3.13 and the fact that $\ell_1, \ldots, \ell_n$ are linearly independent, the polynomial in (3.3) has a $\mathbf{t}$-monomial of degree $\leq 2w^2\mu$. □

We now complete the induction step using lemma that we just proved. As $\det([CMDN]_F)$ is a polynomial in $\mathbb{F}[\mathbf{r}]$ we get the following corollaries.

**Corollary 3.2** *Let $h(\mathbf{r}) := \det([CMDN]_F)$. Then, for every $\mathbf{b} \in F$,*

$$h(\mathbf{r}) \cdot v_{\mathbf{b}} \in \mathbb{F}[\mathbf{t}]\text{-span}\left\{v_{\mathbf{b}'} : \mathbf{b}' \in \{0,1\}^m \text{ and } \text{Supp}\left(\mathbf{b}'\right) \leq 2\mu - q^*\right\}. \tag{3.4}$$

**Proof:** Same as the proof of Corollary 3.1. □

**Corollary 3.3** *Suppose $\{\beta(i) : i \in [n]\}$ are such that the substitution $t_i \mapsto s \cdot z^{\beta(i)}$ keeps all non-zero polynomials in $\mathbb{F}[\mathbf{t}]$ containing a monomial of degree at most $2w^2\mu$ in the $\mathbf{t}$-variables non-zero. Then, the product $M_1(y_1 + r_1) \cdots M_m(y_m + r_m)$ has support-$(2\mu - q^*)$ rank concentration in the $\mathbf{y}$-variables over $\mathbb{F}(s, z)$ after substituting $t_i \to s \cdot z^{\beta(i)}$.*

**Proof:** Multiply both sides of (3.4) by $(h(\mathbf{r}))^{-1}$ after substituting $t_i \mapsto s \cdot z^{\beta(i)}$. □

We now prove that $\{\beta(i) : i \in [n]\}$ as in the above corollary can be computed efficiently.

**Claim 3.15** *There exist $\{\beta(i) : i \in [n]\}$ such that the substitution $t_i \mapsto s \cdot z^{\beta(i)}$ keeps all non-zero polynomials in $\mathbb{F}[\mathbf{t}]$ containing a monomial of degree at most $2w^2\mu$ in the $\mathbf{t}$-variables non-zero. Moreover, we can find all the $\beta(i)$ in time $n^{O(w^4)}$ and each $\beta(i) \leq n^{O(w^4)}$.*

**Proof:** Because of the presence of $s$, the substitution $t_i \mapsto s \cdot z^{\beta(i)}$ keeps any two homogeneous polynomials of different degrees distinct (unless it maps both of them to 0). So, we need to find $\{\beta(i) : i \in [n]\}$ such that the substitution $t_i \mapsto z^{\beta(i)}$ maps any two $\mathbf{t}$-monomials of degree at most $2w^2\mu = O(w^4)$ to distinct monomials in $z$. Now, there are at most $\binom{n+2w^2\mu}{2w^2\mu} = n^{O(w^4)}$ such monomials. Klivans and Spielman [KS01] proved that given $k$ distinct monomials in $y_1, \ldots, y_n$ of degree at most $r$, one can find $a_1, \ldots, a_n$ in time $\text{poly}(n, k, r)$ such that every pair of monomials continues to remain distinct even after substituting $y^{a_i}$ for $y_i$, where $y$ is a fresh variable. So we can find a $\{\beta(i) : i \in [n]\}$ where each $\beta(i) \leq n^{O(w^4)}$ in time $n^{O(w^4)}$. □

This completes the induction step. We now ready to prove Lemma 3.3 stated in Section 3.3.1.

**Proof of Lemma 3.3.** So far we have proved that there exist $\{\beta_{p,q}(i) : p \in [\lceil \log n \rceil], q \in [\mu], i \in [n]\}$, such that

$$G\left(x_1 + \sum_{\substack{p\in\lceil \log n\rceil,\\ q\in[\mu]}} s_{p,q}\cdot z_{p,q}^{\beta_{p,q}(1)}, \ldots, x_n + \sum_{\substack{p\in\lceil \log n\rceil,\\ q\in[\mu]}} s_{p,q}\cdot z_{p,q}^{\beta_{p,q}(n)}\right)$$

has support-$\mu$ rank concentration in the $\mathbf{y}$-variables over $\mathbb{F}\left(s_{p,q}, z_{p,q} : p \in [\lceil \log n \rceil], q \in [\mu]\right)$. Moreover, for each $(p,q)$, we can find all $\beta_{p,q}(i)$ in time $n^{O(w^4)}$ and each $\beta_{p,q}(i) \leq n^{O(w^4)}$. However, since the algorithm that follows from [KS01] is oblivious,[1] the $\beta_{p,q}(i)$ found for some fixed $(p,q)$ can be used for all values of $(p,q)$. This proves the lemma.

### 3.3.3 Proof of Theorem 1.3

Let $f = \mathbf{1}^T \cdot M_1(x_1)M_2(x_2)\cdots M_n(x_n)\cdot \mathbf{1}$ be a multilinear width-$w$ ROABP; here $M_i(x_i) \in \mathbb{F}^{w\times w}[x_i]$ for all $i \in [n]$. Also, let $F = M_1(x_1)M_2(x_2)\cdots M_n(x_n)$. For any $A \in \mathrm{GL}(n,\mathbb{F})$, let $g = f(A\mathbf{x})$ and $G = F(A\mathbf{x})$. For $i \in [n]$, suppose that $A$ maps $x_i \mapsto \ell_i(\mathbf{x})$, where $\ell_i$ is a linear form, and let $y_i = \ell_i(\mathbf{x})$ and $\mathbf{y} = \{y_1, \ldots, y_n\}$. Then, $g = \mathbf{1}^T \cdot M_1(y_1)M_2(y_2)\cdots M_n(y_n)\cdot \mathbf{1}$ and $G = M_1(y_1)M_2(y_2)\cdots M_n(y_n)$. Let $\mu = w^2 + \lceil \log w^2 \rceil$. From Lemma 3.3, there exist polynomials, say $t_1, \ldots, t_n$, in $\mathbb{F}\left[s_{p,q}, z_{p,q} : p \in [\lceil \log n \rceil], q \in [\mu]\right]$ of degree at most $n^{O(w^4)}$ such that $G(x_1 + t_1, \ldots, x_n + t_n)$ has support-$\mu$ rank concentration in the $\mathbf{y}$-variables over $\mathbb{F}\left(\{s_{p,q}, z_{p,q}\}_{p,q}\right)$. Moreover, these polynomials can be computed in time $n^{O(w^4)}$. Suppose that $g \neq 0$. Then, from Observation 2.4, $g(x_1 + t_1, \ldots, x_n + t_n)$ has a support-$\mu$, $\mathbf{y}$-monomial when viewed as a polynomial over $\mathbb{F}\left[\{s_{p,q}, z_{p,q}\}_{p,q}\right]$ in the $\mathbf{y}$-variables. Since $f$ is multilinear, as seen in the proof of Theorem 3.2, $g(x_1 + t_1, \ldots, x_n + t_n)$ has a support-$\mu$, $\mathbf{x}$-monomial. Thus, $g\left(\mathcal{G}_\mu^{SV} + (t_1, \ldots, t_n)\right) \neq 0$. Now, $g\left(\mathcal{G}_\mu^{SV} + (t_1, \ldots, t_n)\right)$ is a polynomial in $2\mu + \mu\cdot\lceil \log n \rceil$ variables over $\mathbb{F}$. Also, its degree is at most $n^{O(w^4)}$. So, if $|\mathbb{F}| > n^{O(w^4)}$, a hitting set for $g$ can be computed in time

$$n^{O(w^4\cdot\mu\cdot\log n)} = n^{O\left(w^6\cdot\log n\right)}.$$

This, along with the time required to compute $t_1, \ldots, t_n$, still gives a $n^{O\left(w^6\cdot\log n\right)}$-time hitting set for $g$.

[1] I.e. given $n, k, r \in \mathbb{N}$, the algorithm in [KS01] outputs an $H \subseteq \mathbb{N}^n$ such that for every set of $k$ distinct monomials in $y_1, \ldots, y_n$, there exists $(a_1, \ldots, a_n) \in H$ such that the monomials continue to remain distinct under the substitution $y_i \mapsto y^{a_i}$.

# Chapter 4

# Hitting sets for orbits of constant occur formulas

This section gives hitting set constructions for orbits of constant occur, constant depth formulas as well as occur once formulas. The contents of this chapter are from a joint work with Chandan Saha [ST24].

## 4.1 Introduction

In this section, we describe quasi-polynomial time hitting set constructions for two subclasses of constant occur formulas: constant depth, constant occur formulas and occur once formulas (see Definition 2.5). As a corollary to our result for the former, we also obtain a quasi-polynomial time hitting set for the orbit of depth-4 set-multilinear formulas. This model has been studied extensively in the literature [SV18, KMSV13]. Moreover, hitting sets for orbits of the sum-product polynomial and power symmetric polynomials (see Definitions 2.12, 2.13) also follow from that result. The hitting set for orbits of constant depth, constant occur formulas is obtained by combining the algebraic independence technique of [ASSS16, BMS13] with the hitting set for orbits of sparse polynomials given by [MS21] and mentioned in Theorem 3.3. The hitting set for orbits of occur-once formulas is obtained by using techniques similar to those used to obtain hitting sets for read-once formulas [SV15].

This section contains the proofs of the following theorems.

**Theorem 1.4 (Hitting sets for the orbits of constant-depth, constant-occur formulas)** *Let $\mathcal{C}$ be the set of $n$-variate, degree-$D$ polynomials that are computable by depth-$\Delta$, occur-$k$ formulas of size $s$. Let $R := (2k)^{2\Delta \cdot 2^{\Delta}}$. If $\mathrm{char}(\mathbb{F}) = 0$ or $> (2ks)^{\Delta^3 R}$, then a hitting set for $\mathrm{orb}(\mathcal{C})$ can be computed in $(nRD)^{O(R(\log R + \Delta \log k + \Delta \log s) + \Delta R)}$ time. If the leaves are labelled by $b$-variate polynomials,*

*then a hitting set for* $\mathrm{orb}(\mathcal{C})$ *can be computed in* $(nRD)^{O(Rb+\Delta R)}$ *time. In particular, if* $\Delta$ *and* $k$ *are constants, then the hitting sets can be constructed in time* $(nD)^{O(\log s)}$ *and* $(nD)^{O(b)}$*, respectively.*

**Theorem 1.5 (Hitting sets for the orbits of occur-once formulas)** *Let* $\mathcal{C}$ *be the set of* $n$*-variate, degree-*$D$ *polynomials that are computable by occur-once formulas whose leaves are labelled by* $s$*-sparse polynomials. If* $|\mathbb{F}| > nD$ *and* $\mathrm{char}(\mathbb{F}) = 0$ *or* $> D$*, then a hitting set for* $\mathrm{orb}(\mathcal{C})$ *can be computed in* $(nD)^{O(\log n+\log s)}$ *time. If the leaves are labelled by* $b$*-variate polynomials, then a hitting set for* $\mathrm{orb}(\mathcal{C})$ *can be computed in* $(nD)^{O(\log n+b)}$ *time.*

For ease of exposition, we first prove 1.4 in the special case of $\Delta = 4$ in Section 4.2; the general case is proved in Section 4.3. Section 4.4 is dedicated to the proof of Theorem 1.5.

## 4.2 Hitting sets for the orbits of depth-4, constant-occur formulas

**The strategy.** We prove Theorem 1.4 by combining the algebraic independence based technique in Agrawal, Saha, Saptharishi, and Saxena [ASSS16, BMS13] with Theorem 3.1. Let $f$ be a constant-depth, constant-occur formula. We first show that it can be assumed without loss of generality that the top-most gate of $f$ is a $+$ gate whose fan-in is upper bounded by the occur of $f$, say $k$. In [ASSS16], they were able to upper bound the top fan-in by simply translating a variable by 1 and subtracting the original formula. However, the same idea does not quite work here because we have only access to a polynomial in the *orbit* of $f$. To upper bound the top fan-in, we show that there exists a variable $x_i$ such that $\frac{\partial f}{\partial x_i}$ is a constant-depth, constant-occur formula with top fan-in bounded by $k$. Then, using the chain rule of differentiation, we show that one can construct a hitting set generator for $\mathrm{orb}(f)$ from a generator for $\mathrm{orb}\left(\frac{\partial f}{\partial x_i}\right)$; this means that we can shift our attention to $f' = \frac{\partial f}{\partial x_i}$, which we shall henceforth refer to as $f$.

Let $f = f_1 + \cdots + f_k$, $A \in \mathrm{GL}(n, \mathbb{F})$, $g = f(A\mathbf{x})$, $g = g_1 + \ldots + g_k$ where for all $i \in [k]$, $g_i = f_i(A\mathbf{x})$. It was shown in [ASSS16] that a homomorphism, which is faithful (see Definition 2.22) to $f_1, \ldots, f_k$, is a hitting set generator for $f$. In our case, this translates to 'a homomorphism that is faithful to $g_1, \ldots, g_k$ is a hitting set generator for $g$ '. [ASSS16] also showed that the problem of constructing a homomorphism $\phi$ that is faithful to $f_1, \ldots, f_k$ reduces to constructing a homomorphism $\psi$ that preserves the determinant of a certain matrix. This matrix is an appropriate sub-matrix of the Jacobian of $f_1, \ldots, f_k$. Also, it was argued that its determinant is a product of sparse polynomials and so $\psi$ was obtained from Klivans and Spielman [KS01]. We use a similar argument, along with the chain rule, to show

that the problem of constructing a homomorphism $\phi$ that is faithful to $g_1, \ldots, g_k$ reduces to constructing a homomorphism $\psi$ that preserves the determinant of a sub-matrix of the same Jacobian *evaluated at* $A\mathbf{x}$. As this determinant is a product of polynomials in the orbit of sparse polynomials, we can use Theorem 3.1 to construct such a $\psi$.

**Notations.** For some $k \in \mathbb{N}$, let $f \in \mathbb{F}[\mathbf{x}]$ be an $n$-variate, degree-$D$ polynomial computed by a $(4, k, s)$ formula, i.e., a depth-4, occur-$k$ formula of size-$s$. We will identify $f$ with the formula computing it. We first upper bound the top fan-in of $f$ in Section 4.2.1 and then use the notion of faithful homomorphisms, defined in Section 2.5.3, to construct hitting sets for $\text{orb}(f)$.

### 4.2.1 Upper bounding the top fan-in of $f$

To upper bound the fan-in of $f$, we show that for all $i \in [n]$, $\frac{\partial f}{\partial x_i}$ is a depth-4, occur-$k'$ formula with top fan-in at most $k$; here $k'$ is not too large compared to $k$ (see Claim 4.1 below). We then argue in Claim 4.2 that there exists an $i \in [n]$ such that a hitting set generator for $\text{orb}(f)$ can be constructed using a hitting set generator for $\text{orb}(\frac{\partial f}{\partial x_i})$. Thus, by overloading the notation and referring to $\frac{\partial f}{\partial x_i}$ as $f$, we can assume that the top fan-in of $f$ is at most $k$.

**Claim 4.1** *Let $f$ be a $(4, k, s)$ formula. Then, for every $i \in [n]$, $\frac{\partial f}{\partial x_i}$ is a $(4, 2k^2, 2ks)$ formula with top fan-in bounded by $k$.*

**Proof:** Let $x = x_i$. Let $f = \sum_{i\in[m]} f_i$, and $x$ be present only in $f_1, \ldots, f_r$, where $r \leq k$. Furthermore, for all $i \in [r]$, let $f_i = \prod_{j\in m_i} q_{i,j}^{e_{i,j}}$ and $x$ be present only in $q_{i,1}, \ldots q_{i,r_i}$, $\sum_{i\in[r]} r_i \leq k$. Here, $q_{i,j}$ are $s$-sparse polynomials. Now,

$$\begin{aligned}\frac{\partial f}{\partial x} &= \sum_{i\in[r]} \left(\prod_{j=r_i+1}^{m_i} q_{i,j}^{e_{i,j}}\right) \cdot \left(\sum_{j\in[r_i]} e_{i,j} \frac{\partial q_{i,j}}{\partial x} \cdot q_{i,j}^{e_{i,j}-1} \cdot \prod_{\substack{j'\in[r_i]\\ j'\neq j}} q_{i,j'}^{e_{i,j'}}\right) \\ &= \sum_{i\in[r]} \sum_{j\in[r_i]} \left(e_{i,j} \frac{\partial q_{i,j}}{\partial x} \cdot \prod_{j'\in[m_i]} q_{i,j'}^{e'_{i,j'}}\right),\end{aligned}$$

where $e'_{i,j'} = e_{i,j'}$ for $j' \neq j$ and $e'_{i,j} = e_{i,j} - 1$. First of all, notice that the top fan-in of $\frac{\partial f}{\partial x}$ is at most $\sum_{i\in[r]} r_i \leq k$, its depth is 4, and as the leaves are still $q_{i,j}$ or $\frac{\partial q_{i,j}}{\partial x}$, the sparsity of the polynomials labelling the leaves are also at most $s$. However, the size and the occur may

change.

For all $i \in [r]$, let the occur of $f_i$ be $p_i \leq k$; then the occur of $\prod_{j' \in [m_i]} q_{i,j'}^{e'_{i,j'}}$ is at most $p_i$. Also, as $\frac{\partial q_{i,j}}{\partial x}$ is an $s$-sparse polynomial, its occur is 1. Then, the occur of $\frac{\partial f}{\partial x}$ is at most

$$\sum_{i \in [r]} r_i \left(1 + p_i\right) \leq \sum_{i \in [r]} r_i + \sum_{i \in [r]} r_i k \leq k + k^2 \leq 2k^2.$$

Similarly, suppose that the size of $f_i$ is $s_i \leq s - 1$ [1]; then the size of $\prod_{j' \in [m_i]} q_{i,j'}^{e'_{i,j'}}$ is at most $s_i - 1$ (as $e'_{i,j} = e_{i,j} - 1$). Also, as the size of $q_{i,j}$ is $\leq s$, the size of $\frac{\partial q_{i,j}}{\partial x}$ is at most $s$. So, the size of $\frac{\partial f}{\partial x}$ is at most

$$\sum_{i \in [r]} r_i \left(s + s_i + 1\right) \leq \sum_{i \in [r]} r_i \left(s + s\right) \leq 2ks.$$

□

We now show that there exists an $i \in [n]$ such that a hitting set generator for $\text{orb}(f)$ can be constructed using a hitting set generator for $\text{orb}(\frac{\partial f}{\partial x_i})$.

**Claim 4.2** *Let $f \in \mathbb{F}[\mathbf{x}]$ be an $n$-variate polynomial of degree $D$, and $\text{char}(\mathbb{F}) = 0$ or $> D$. There is an $i \in [n]$ such that $\frac{\partial f}{\partial x_i} \neq 0$, and if $\mathcal{G}$ is a hitting set generator for $\text{orb}\left(\frac{\partial f}{\partial x_i}\right)$, then $\widetilde{\mathcal{G}} := \mathcal{G} + \mathcal{G}_1^{SV}$ is a hitting set generator for $\text{orb}(f)$, provided $|\mathbb{F}| > \deg(\mathcal{G}) \cdot D$.*

**Proof:** Let $A \in \text{GL}(n, \mathbb{F})$ and $g = f(A\mathbf{x})$. If $f$ is a constant, then constructing a hitting set for $\text{orb}(f)$ is trivial. Otherwise, there exists an $i \in [n]$ such that $\frac{\partial f}{\partial x_i} \neq 0$ (because $\text{char}(\mathbb{F}) = 0$ or $> D$). Suppose that a polynomial map $\mathcal{G}$ is a hitting set generator for $\text{orb}\left(\frac{\partial f}{\partial x_i}\right)$. The gradient of a polynomial $p(\mathbf{x})$, denoted by $\nabla p$, is the column vector $\left(\frac{\partial p}{\partial x_1} \ \frac{\partial p}{\partial x_2} \ \ldots \ \frac{\partial p}{\partial x_n}\right)^T$. By the chain rule of differentiation,

$$\nabla g = A^T \cdot [\nabla f](A\mathbf{x}).$$

As $A^T$ is invertible, $\frac{\partial f}{\partial x_i}(A\mathcal{G}) \neq 0 \implies [\nabla f](A\mathcal{G}) \neq 0 \implies [\nabla g](\mathcal{G}) \neq 0 \implies \exists j \in [n]$ such that $\frac{\partial g}{\partial x_j}(\mathcal{G}) \neq 0$. Since $|\mathbb{F}| > \deg(\mathcal{G}) \cdot D$, by Observation 2.2, $g(\widetilde{\mathcal{G}})$ is not a constant.

□

[1] 1 less than $s$, as $f_i$ is connected to the top-most $+$ gate by an edge.

All we need to do now is construct a hitting set generator for orb $\left(\frac{\partial f}{\partial x_i}\right)$. Overloading the notation, we refer to $\frac{\partial f}{\partial x_i}$ as $f$, which is computed by a $(4, k, s)$ formula whose top fan-in is at most $k$.

### 4.2.2 Constructing a faithful homomorphism for the orbits

Let $f = \sum_{i\in[m]} f_i$ be a $(4, k, s)$ formula. From the discussion in the previous section, we can assume without loss of generality that $m \leq k$. Let $A \in \mathrm{GL}(n, \mathbb{F})$, and $g_i = f_i(A\mathbf{x})$ for all $i \in [m]$. Recall that a homomorphism $\phi$ is said to be faithful to $\mathbf{g} = (g_1, \ldots, g_m) \in \mathbb{F}[\mathbf{x}]^m$ if $\text{tr-deg}_{\mathbb{F}}(\mathbf{g}) = \text{tr-deg}_{\mathbb{F}}(\phi(\mathbf{g}))$. Also, from Lemma 2.3, if $\phi$ is faithful to $\mathbf{g}$, then for any $m$-variate polynomial $p$, $p(\phi(\mathbf{g})) = 0$ if and only if $p(\mathbf{g}) = 0$. Thus, if we have a homomorphism $\phi$ that is faithful to $\mathbf{g}$ (irrespective of $A$), then we can use $\phi$ as a hitting set generator for orb$(f)$. The following lemma helps us construct such a homomorphism.

**Lemma 4.1** *Let $\mathbf{f} = (f_1, ..., f_m) \in \mathbb{F}[\mathbf{x}]^m$ be a tuple of $n$-variate polynomials of degree at most $\delta$, $A \in \mathrm{GL}(n, \mathbb{F})$, $g_i = f_i(A\mathbf{x})$ for all $i \in [m]$, and $\mathbf{g} = (g_1, \ldots, g_m)$. Further, suppose that $\text{tr-deg}_{\mathbb{F}}(\mathbf{f}) \leq r$, and $\mathrm{char}(\mathbb{F}) = 0$ or $> \delta^r$. Let $\psi : \mathbb{F}[\mathbf{x}] \to \mathbb{F}[\mathbf{z}]$ be a homomorphism such that $\mathrm{rank}_{\mathbb{F}(\mathbf{x})} J_{\mathbf{x}}(\mathbf{f})(A\mathbf{x}) = \mathrm{rank}_{\mathbb{F}(\mathbf{z})} \psi(J_{\mathbf{x}}(\mathbf{f})(A\mathbf{x}))$. Then, the map $\phi : \mathbb{F}[\mathbf{x}] \to \mathbb{F}[\mathbf{z}, t, y_1, ..., y_r]$ that, for all $i \in [n]$, maps*

$$x_i \mapsto \left(\sum_{j=1}^{r} y_j t^{ij}\right) + \psi(x_i)$$

*is faithful to $\mathbf{g}$.*

**Proof:** Let $J_{\mathbf{x}}(\mathbf{g})$ be the Jacobian matrix of $\mathbf{g}$, and $J_{\mathbf{x}}(\mathbf{f})(A\mathbf{x})$ the Jacobian matrix of $\mathbf{f}$ evaluated at $A\mathbf{x}$. From the chain rule of differentiation, $J_{\mathbf{x}}(\mathbf{g}) = J_{\mathbf{x}}(\mathbf{f})(A\mathbf{x}) \cdot A$. As $A$ in an invertible matrix,

$$\mathrm{rank}_{\mathbb{F}(\mathbf{x})} J_{\mathbf{x}}(\mathbf{g}) = \mathrm{rank}_{\mathbb{F}(\mathbf{x})} J_{\mathbf{x}}(\mathbf{f})(A\mathbf{x}). \tag{4.1}$$

Also, for any homomorphism $\psi : \mathbb{F}[\mathbf{x}] \to \mathbb{F}[\mathbf{z}]$, $\psi\left(J_{\mathbf{x}}(\mathbf{g})\right) = \psi\left(J_{\mathbf{x}}(\mathbf{f})(A\mathbf{x})\right) \cdot A$ and hence,

$$\mathrm{rank}_{\mathbb{F}(\mathbf{z})} \psi\left(J_{\mathbf{x}}(\mathbf{g})\right) = \mathrm{rank}_{\mathbb{F}(\mathbf{z})} \psi\left(J_{\mathbf{x}}(\mathbf{f})(A\mathbf{x})\right). \tag{4.2}$$

So, if we have a homomorphism $\psi$ satisfying $\mathrm{rank}_{\mathbb{F}(\mathbf{x})} J_{\mathbf{x}}(\mathbf{f})(A\mathbf{x}) = \mathrm{rank}_{\mathbb{F}(\mathbf{z})} \psi\left(J_{\mathbf{x}}(\mathbf{f})(A\mathbf{x})\right)$, then from (4.1) and (4.2),

$$\mathrm{rank}_{\mathbb{F}(\mathbf{x})} J_{\mathbf{x}}(\mathbf{g}) = \mathrm{rank}_{\mathbb{F}(\mathbf{z})} \psi\left(J_{\mathbf{x}}(\mathbf{g})\right).$$

Also, from Observation 2.5, tr-deg($\mathbf{g}$) = tr-deg($\mathbf{f}$) $\leq r$, and deg($g_i$) = deg($f_i$) $\leq \delta$. So, using Lemma 2.4, we can construct a homomorphism $\phi$ faithful to $\mathbf{g}$ from $\psi$, as stated in the lemma. □

Let us apply Lemma 4.1 to the $(4, k, s)$ formula $f = \sum_{i\in[m]} f_i$, where $m \leq k$. Let $\mathbf{f} = (f_1, ..., f_m)$ and tr-deg$_{\mathbb{F}}(\mathbf{f}) = r \leq m \leq k$.[1] Observe that the degree of each $f_i$ is at most $\delta \leq s^2$. Then, from Lemma 2.2, rank$_{\mathbb{F}(\mathbf{x})}$ $J_{\mathbf{x}}(\mathbf{f}) = r$, provided char($\mathbb{F}$) = 0 or $> \delta^r$. As $A$ is invertible, this means that rank$_{\mathbb{F}(\mathbf{x})}$ $J_{\mathbf{x}}(\mathbf{f})(A\mathbf{x}) = r$. Assume without loss of generality that $\{f_1, \ldots, f_r\}$ is a transcendence basis of $\mathbf{f}$. Then, again from Lemma 2.2, the sub-matrix of $J_{\mathbf{x}}(\mathbf{f})$ consisting of the rows corresponding to $f_1, \ldots, f_r$ must be full rank. Thus, we can assume without loss of generality that the minor $M$ of $J_{\mathbf{x}}(\mathbf{f})$ consisting of those rows, and columns corresponding to $x_1, \ldots, x_r$, has non-zero determinant. Notice that, as $A$ is invertible, the determinant of $M$ evaluated at $A\mathbf{x}$, i.e., det($M(A\mathbf{x})$) = [det($M$)]($A\mathbf{x}$) is also non-zero. To ensure that the rank$_{\mathbb{F}(\mathbf{z})}$ $\psi\left(J_{\mathbf{x}}(\mathbf{f})(A\mathbf{x})\right)$ is also $r$, it suffices to construct a homomorphism $\psi$ that is a hitting set generator for orb(det($M$)).

**Constructing $\psi$.** Let us look at det($M$) a little more closely. As before, let $f_i = \prod_{j\in m_i} q_{i,j}^{e_{i,j}}$, where $q_{i,j}$ are $s$-sparse polynomials of degree at most $s$. For $i \in [r]$, let the number of $q_{i,j}$ containing any of $x_1, \ldots, x_r$ be $c_i$. As $f$ is an occur-$k$ formula, $\sum_{i\in[m]} c_i \leq kr \leq k^2$. From the $i$-th row of $M$, we can factor out $q_{i,j}^{e_{i,j}}$ if $q_{i,j}$ does not contain any of $x_1, \ldots, x_r$. Moreover, even if $q_{i,j}$ contains some variable from $x_1, \ldots, x_r$, we can still factor out $q_{i,j}^{e_{i,j}-1}$. After we have taken out all these factors, let the residual matrix be $M'$. Then, each entry of the $i$-th row of $M'$ is a polynomial with sparsity at most $c_i s^{c_i}$ and degree at most $c_i s$. Thus, det($M'$) is a polynomial with sparsity at most $r! \cdot \prod_{i\in[r]} c_i s^{c_i} \leq k! \cdot k^k \cdot s^{k^2} \leq k^{2k} \cdot s^{k^2}$ and degree at most $\sum_{i\in[r]} c_i s \leq k^2 s$. So, det($M$) is a product of polynomials with sparsity at most $k^{2k} \cdot s^{k^2}$ and degree at most $k^2 s$. From Theorem 3.3, $\psi = \mathcal{G}^{SV}_{\left(\left\lceil \log\left(k^{2k}\cdot s^{k^2}\right)\right\rceil + 1\right)} = \mathcal{G}^{SV}_{O(k^2(\log k + \log s))}$ is a hitting set generator for orb(det($M$)), if $|\mathbb{F}| > nk^2 s$ and char($\mathbb{F}$) = 0 or $> k^2 s$.

If the $q_{i,j}$ are $b$-variate polynomials, then det($M'$) is a polynomial in $\sum_{i\in[r]} c_i b \leq k^2 b$ variables. From Observation 2.1, $\psi = \mathcal{G}^{SV}_{k^2 b}$ is a hitting set generator for orb(det($M$)).

**Constructing $\phi$.** Using $\psi$ and Lemma 4.1, we get a homomorphism $\phi$ that is faithful to $\mathbf{g}$. Observe that $\phi$ is a polynomial map in at most $O\left(k^2\left(\log k + \log s\right)\right) + k + 1 = O\left(k^2\left(\log k + \log s\right)\right)$ variables and of degree at most $nk + 1$ (as the degree of the polynomial map $\psi$ is at most $n$

[1] Recall that, by definition, tr-deg$_{\mathbb{F}}(\mathbf{f})$ is just the transcendence degree of the set $\{f_1, \ldots, f_m\}$.

and, in Lemma 4.1, $\deg\left(\sum_{j=1}^{r} y_j t^{ij}\right) \leq nk+1$).

If the $q_{i,j}$ are $b$-variate polynomials, then $\phi$ is a polynomial map in at most $O(k^2 b)+k+1 = O(k^2 b)$ variables and of degree at most $nk+1$.

### 4.2.3 Proof of Theorem 1.4: the depth-4 case

For $\Delta = 4$, the value of $R$ in the statement of theorem is $(2k)^{128}$. However, in this special case, one can work with a much smaller value for $R$. We choose $R = k^4$ so that $\mathrm{char}(\mathbb{F}) = 0$ or $> (2ks)^{6k^2}$. This ensures that the constraints on $\mathrm{char}(\mathbb{F})$ and $|\mathbb{F}|$, coming from Claim 4.2, Lemma 4.1 and the application of Theorem 3.3 in the construction of $\psi$, are satisfied.

Let $f$ be a $(4,k,s)$ formula. If $f$ is a constant, then so is every polynomial in $\mathrm{orb}(f)$. In this case, the set containing any non-zero point in $\mathbb{F}^n$ is a hitting set for $\mathrm{orb}(f)$; so suppose that $f$ is not a constant. There exists an $i \in [n]$ such that $\frac{\partial f}{\partial x_i} \neq 0$ (as $\mathrm{char}(\mathbb{F}) = 0$ or $> s^2 \geq D$). From Claim 4.1, $\frac{\partial f}{\partial x_i} \neq 0$ can be computed by a $(4, 2k^2, 2ks)$ formula with top fan-in at most $k$. Moreover, from the proof of Claim 4.2, if $\mathcal{G}$ is a hitting set generator for $\mathrm{orb}\left(\frac{\partial f}{\partial x_i}\right)$, then $\widetilde{\mathcal{G}} = \mathcal{G} + \mathcal{G}_1^{SV}$ is a hitting set generator for $\mathrm{orb}(f)$, provided $\mathrm{char}(\mathbb{F}) = 0$ or $> D$ and $|\mathbb{F}| > \deg(\mathcal{G}) \cdot D$. From Section 4.2.2, there exists a $\mathcal{G}$ that has at most $O\left(\left(2k^2\right)^2 \left(\log 2k^2 + \log 2ks\right)\right) = O\left(k^4\left(\log k + \log s\right)\right)$ many variables and has degree at most $2nk^2+1$. As $\mathcal{G}_1^{SV}$ has 2 variables and has degree $n$, $\widetilde{\mathcal{G}}$ has $O\left(k^4\left(\log k + \log s\right)\right)$ variables and has degree at most $2nk^2+1$. Thus, for any $g \in \mathrm{orb}(f)$, $g(\widetilde{\mathcal{G}})$ has $O\left(k^4\left(\log k + \log s\right)\right)$ variables and has degree at most $(2nk^2+1)D$. So, a hitting set for $\mathrm{orb}(f)$ can be computed in time $(nk^2D)^{O\left(k^4(\log k+\log s)\right)} = (nRD)^{O(R(\log k+\log s))}$.

The proof for the case where the leaves are labelled by $b$-variate polynomials is similar. Now, $\mathcal{G}$ has $O(k^4 b)$ variables and has degree at most $2nk^2+1$. Thus, $g(\widetilde{\mathcal{G}})$ has $O(k^4 b)$ variables and is of degree at most $(2nk^2+1)D$, and so, a hitting set for $\mathrm{orb}(f)$ can be computed in $(nk^2D)^{O(k^4 b)}$ time.

## 4.3 Hitting sets for the orbits of constant-depth, constant-occur formulas

Let $f \in \mathbb{F}[\mathbf{x}]$ be a $n$-variate, degree-$D$ polynomial computed by a $(\Delta,k,s)$ formula i.e., a depth-$\Delta$, occur-$k$ formula of size-$s$. Let us identify $f$ with a $(\Delta,k,s)$ formula computing it. In this section, the level of a gate in $f$ will be one plus its distance from the output gate of $f$. Just like we did in Section 4.2, we first upper bound the top fan-in of $f$ in Section 4.3.1 and then use the notion of faithful homomorphisms to construct hitting sets for $\mathrm{orb}(f)$ in Section

4.3.2.

### 4.3.1 Upper bounding the top fan-in of $f$

We begin by showing that $f$ can be written in a "canonical" form.

**Claim 4.3** *If $f$ is a $(\Delta, k, s)$ formula, then it can also be computed by a $(\Delta, k, (2s)^{\Delta})$ formula in a canonical form with the following properties:*

1. *All gates connected to the leaves of $f$ are $\times$人 gates.*
2. *$f$ has alternating levels of $+$ and $\times$人 gates.*

**Proof:** While $f$ contains a $+$ gate connected to the leaves, we merge all the leaves connected to it into a single leaf node computing their sum. Now, if this $+$ gate is not connected to any gate other than this leaf, it can simply be replaced by the leaf after multiplying the sparse polynomial computed by the leaf by the label of the edge between it and the $+$ gate. This does not increase the depth, size or occur of $f$. Otherwise, we add a $\times$人 gate between the $+$ gate and the leaf. While this can increase the size of $f$ by a factor of 2, the occur remains the same. The depth does not increase, because the $+$ gate is also connected to some non-leaf node. Now $f$ has property 1.

If $f$ has a $+$ gate $q$ which is fed another $+$ gate $h$ as input and the edge connecting them is labelled by $\alpha$, then we can simply remove $h$, connect all its inputs directly to $q$ and multiply the labels of edges connecting all these inputs to $q$ by $\alpha$. This modification to $f$ clearly does not increase its depth, size or occur. Also, now each sum gate in $f$ is connected solely to $\times$人 gates.

Consider any *maximal* sub-tree of $f$ made up, solely, of $\times$人 gates. Let its root be $q$ and its inputs $h_1, \ldots, h_m$. Then, $q = h_1^{e_1} \cdots h_m^{e_m}$, where $e_i$ is the product of the weights of all edges on the path from $h_i$ to $q$. As the sub-tree is maximal, none of $h_1, \ldots, h_m$ are $\times$人 gates and $q$ is also not an input to a $\times$人 gate. Thus, if we replace each such sub-tree with a single $\times$人 gate computing the same polynomial, $f$ will also satisfy 2. Notice that, doing this does not increase the depth or occur; size on the other hand, may increase. Suppose that the depth of the sub-tree is $\Delta'$. Let the sum of weights of edges connecting gates at level $\ell + 1$ (from $q$) to gates at level $\ell$ be $r_\ell \leq 2s$, for all $\ell \in [\Delta' - 1]$. Also, let the sum of weights of edges connecting the leaves be $r_{\Delta'}$. As, all edge weights are non-negative, $\sum_{i \in [m]} e_i \leq \prod_{\ell \in [\Delta']} r_\ell \leq (2s)^{\Delta'} \leq (2s)^{\Delta - 2}$. Since, there can be no more than $(2s)$ such sub-trees, the size of $f$ can increase by at most $(2s)^{\Delta - 1}$. Thus, size of $f$ is at most $2s + (2s)^{\Delta - 1} \leq (2s)^{\Delta}$. □

We can also assume that the output gate of $f$ is not a $\times_{\mathcal{K}}$ gate, for otherwise, we only need to construct a hitting set generator for orbits of all of its factors which themselves are $(\Delta-1, k, (2s)^{\Delta})$ formulas, with $+$ gates at the top or are sparse polynomials. Thus, we can assume without loss of generality that $\Delta$ is an even number: if $\Delta \neq 2$, then the top most gate is a $+$ gate, $f$ has alternating levels of $+$ and $\times_{\mathcal{K}}$ gates and gates connected to the leaves are $\times_{\mathcal{K}}$ gates. We now make the following claim which will allow us to assume that the top fan-in of $f$ is at most $k$.

**Claim 4.4** *Let $f$ be a $(\Delta, k, s)$ formula in the canonical form of Claim 4.3, with either a $+$ gate at the top or $\Delta = 2$. Then, for any $i \in [n]$, $\frac{\partial f}{\partial x_i}$ is a $(\Delta, (2k)^{\Delta/2}, (2k)^{\Delta/2}s)$ formula in the canonical form with the top fan-in bounded by $k$.*

**Proof:** When $\Delta = 2$, $f$ is a polynomial of sparsity $s$ and $k = 1$. So, the sparsity of $\frac{\partial f}{\partial x_i}$ is at most $s$ and the depth and occur do not increase, making the claim true. Assume, by the way of induction, that the claim is true for all formulas of depth $\Delta - 2$. Let $x = x_i$, $f = \sum_{i\in[m]} f_i$ and $x$ be present only in $f_1, \ldots, f_r$, $r \leq k$. Furthermore, for all $i \in [r]$, let $f_i = \prod_{j\in m_i} q_{i,j}^{e_{i,j}}$ and $x$ be present only in $q_{i,1}, \ldots q_{i,r_i}$, $\sum_{i\in[r]} r_i \leq k$. Then,

$$\begin{aligned}\frac{\partial f}{\partial x} &= \sum_{i\in[r]} \left( \prod_{j=r_i+1}^{m_i} q_{i,j}^{e_{i,j}} \right) \cdot \left( \sum_{j\in[r_i]} e_{i,j} \frac{\partial q_{i,j}}{\partial x} \cdot q_{i,j}^{e_{i,j}-1} \cdot \prod_{\substack{j'\in[r_i] \\ j'\neq j}} q_{i,j'}^{e_{i,j'}} \right) \\ &= \sum_{i\in[r]} \sum_{j\in[r_i]} \left( \frac{\partial q_{i,j}}{\partial x} \cdot \prod_{j'\in[m_i]} q_{i,j'}^{e'_{i,j'}} \right),\end{aligned}$$

where $e'_{i,j'}$ is either $e_{i,j'}$ or $e_{i,j'} - 1$. First of all, notice that, the top fan-in of $\frac{\partial f}{\partial x}$ is at most $\sum_{i\in[r]} r_i \leq k$. As all $q_{i,j}$ are formulas of depth $\Delta - 2$, from the induction hypothesis, $\frac{\partial q_{i,j}}{\partial x}$ is also a depth $\Delta - 2$ formula. Thus, the depth of $\frac{\partial f}{\partial x}$ is at most $\Delta$. However, the size and occur may change.

For all $i \in [r]$, let the occur of $f_i$ be $p_i \leq k$; then the occur of $\prod_{j'\in[m_i]} q_{i,j'}^{e'_{i,j'}}$ is at most $p_i$. Also, from the induction hypothesis, $\frac{\partial q_{i,j}}{\partial x}$ has occur $(2k)^{(\Delta-2)/2}$. So, the occur of $\frac{\partial f}{\partial x}$ is at most $\sum_{i\in[r]} r_i \left( (2k)^{(\Delta-2)/2} + p_i \right)$, which can be bounded from above by $(2k)^{\Delta/2}$. Similarly, suppose that the size of $f_i$ is $s_i \leq s - 1$; then the size of $\prod_{j'\in[m_i]} q_{i,j'}^{e'_{i,j'}}$ is at most $s_i$.

Also, from the induction hypothesis, $\frac{\partial q_{i,j}}{\partial x}$ has size $(2k)^{\frac{\Delta-2}{2}}s$. So, the size of $\frac{\partial f}{\partial x}$ is at most $\sum_{i\in[r]} r_i \left((2k)^{\frac{\Delta-2}{2}}s + s_i + 1\right) \leq (2k)^{\Delta/2}s$. □

We now upper bound the top fan-in of $f$ using this claim. Let $A \in \mathrm{GL}(n,\mathbb{F})$ and $g(\mathbf{x}) = f(A\mathbf{x})$. If $f$ is a constant, then constructing a hitting set for $\mathrm{orb}(f)$ is trivial. Otherwise, there exists an $i \in [n]$ such that $\frac{\partial f}{\partial x_i} \neq 0$ (because $\mathrm{char}(\mathbb{F}) > (2ks)^{\Delta^3 R} \geq D$). Suppose that a polynomial map, $\mathcal{G} : \mathbb{F}^t \to \mathbb{F}^n$ of degree at most $nR+1$ is a hitting set generator for orb $\left(\frac{\partial f}{\partial x_i}\right)$. The gradient of a polynomial $p(\mathbf{x})$, denoted by $\nabla p$, is the column vector $\left(\frac{\partial p}{\partial x_1}\ \frac{\partial p}{\partial x_2}\ \cdots\ \frac{\partial p}{\partial x_n}\right)^T$. By the chain rule of differentiation,

$$\nabla g = A^T \cdot [\nabla f](A\mathbf{x}).$$

As $A^T$ is invertible, $\frac{\partial f}{\partial x_i}(A\mathcal{G}) \neq 0 \implies \nabla f(A\mathcal{G}) \neq 0 \implies \nabla g(\mathcal{G}) \neq 0 \implies \exists \in [n]$, such that $\frac{\partial g}{\partial x_j}(\mathcal{G}) \neq 0$. Then, from Observation 2.2, for $\widetilde{\mathcal{G}} := \mathcal{G} + \mathcal{G}_1^{SV}$, $g(\widetilde{\mathcal{G}}) \neq 0$, i.e., $\widetilde{\mathcal{G}}$ is a hitting set generator for $\mathrm{orb}(f)$. So, all we need to do now is construct a hitting set generator for orb $\left(\frac{\partial f}{\partial x_j}\right)$ and from Claim 4.4, $\frac{\partial f}{\partial x_j}$ has top fan-in at most $k$. Overloading the notation, we refer to $\frac{\partial f}{\partial x_j}$ as $f$, which is computed by a $(\Delta,k,s)$ formula in the canonical form and with a $+$ gate at the top whose fan-in is at most $k$.

### 4.3.2 Constructing a faithful homomorphism

Let $f = f_1 + \cdots + f_k$ and $A \in \mathrm{GL}(n,\mathbb{F})$. Let $g_i = f_i(A\mathbf{x})$ for all $i \in [k]$, $\mathbf{f} = (f_1,\ldots,f_k)$ and $\mathbf{g} = (g_1,\ldots,g_k)$. We now show how to create a homomorphism $\phi$ that is faithful to $\mathbf{g}$; from Lemma 2.3, this homomorphism will be a hitting set generator for $\mathrm{orb}(f)$. $\phi$ will be constructed recursively as follows: each level of recursion corresponds to a level in $f$, with the recursion starting at level 2 and ending at level $\Delta-2$. At level $\ell$, our goal will be to construct a homomorphism $\phi_\ell$ which is faithful to every tuple in a certain set $C_\ell$ of tuples. Each tuple in $C_\ell$ consists of at most $r_\ell$ derivatives, of order at most $a_\ell$, of disjoint groups of gates at level $\ell$ of $f$, evaluated at $A\mathbf{x}$. Note that, as the derivatives are of disjoint groups of gates in $f$, $|C_\ell| \leq s$.

For $\ell = 2$, $C_2$ contains only one tuple, namely $\mathbf{g}$, $r_2 = k$ and $a_2 = 0$. For any $\ell \geq 2$, let $\mathbf{q} \in C_\ell, \mathbf{q} = (q_1,\ldots,q_{r_\ell})$, where $q_i = h_i(A\mathbf{x})$ for all $i \in [r_\ell]$ and let $\mathbf{h} = (h_1,\ldots,h_{r_\ell})$. If $\phi_{\ell+1}$ is such that $\mathrm{rank}_{\mathbb{F}(\mathbf{x})}\, J_{\mathbf{x}}(\mathbf{h})(A\mathbf{x}) = \mathrm{rank}_{\mathbb{F}(\mathbf{z})}\, \phi_{\ell+1}\left(J_{\mathbf{x}}(\mathbf{h})(A\mathbf{x})\right)$, then using Lemma 4.1, we can construct a $\phi_\ell$ faithful to $\mathbf{q}$. The following lemma which was proved in [ASSS16], helps us reduce the problem from level $\ell$ to level $\ell+1$.

**Lemma 4.2 (Lemma 4.4 of [ASSS16])** *Let* $\mathbf{h}$ *be a tuple of* $r_\ell$ *derivatives, of order at most* $a_\ell$*, of*

*gates $G$ at level $\ell$ of $f$, $\text{tr-deg}_{\mathbb{F}}(\mathbf{h}) = r'_\ell$ and $\mathbf{h}'$ be a transcendence basis of $\mathbf{h}$. Any $r'_\ell \times r'_\ell$ minor of $J_{\mathbf{x}}(\mathbf{h}')$ is of the form $\prod_i p_i^{e_i}$, where $p_i$s are polynomials in at most $r_{\ell+1} := (a_\ell + 1) \cdot 2^{a_\ell+1} k \cdot r_\ell^2$ many derivatives of order at most $a_{\ell+1} := a_\ell + 1$ of disjoint groups of children of $G$.*

For each $\mathbf{h}$, we will use the above lemma for a non-zero $r'_\ell \times r'_\ell$ minor of $J_{\mathbf{x}}(\mathbf{h}')$. Then, the lemma gives a bunch of tuples $\mathbf{h}_1, \ldots, \mathbf{h}_u$, one for each $p_i$. Suppose that $p_i$ is a polynomial in $p_{i,1}, \ldots, p_{i,m}$, which are derivatives of gates at level $\ell+1$ of $f$. Then, $\mathbf{h}_i = (p_{i,1}(A\mathbf{x}), \ldots, p_{i,m}(A\mathbf{x}))$ and $C_{\ell+1}$ is a set of all $\mathbf{h}_i$, for all $\mathbf{h}$. If $\phi_{\ell+1}$ is faithful to each tuple in $C_{\ell+1}$, then from Lemma 2.3, $\phi_{\ell+1}\left(p_i^{e_i}(A\mathbf{x})\right) \neq 0$ and hence it preserves the rank of $J_{\mathbf{x}}(\mathbf{h})(A\mathbf{x})$.

The base case of the recursion is when $\ell = \Delta - 2$. Our goal is to create a homomorphism $\phi_{\Delta-2}$ which is faithful to every tuple in the set $C_{\Delta-2}$, $|C_{\Delta-2}| \leq s$ of at most $r_{\Delta-2}$ many sparse polynomials (because any derivative of a sparse polynomial is a sparse polynomial) evaluated at $A\mathbf{x}$. $r_{\Delta-2}$ can be bounded from above by $R := (2k)^{2\Delta \cdot 2^\Delta}$. For all $\mathbf{q} = \mathbf{h}(A\mathbf{x}) = (h_1(A\mathbf{x}), \ldots, h_R(A\mathbf{x})) \in C_{\Delta-2}$, we will create a $\phi_{\Delta-1}$ such that

$$\text{rank}_{\mathbb{F}(\mathbf{x})}\, J_{\mathbf{x}}(\mathbf{h})(A\mathbf{x}) = \text{rank}_{\mathbb{F}(\mathbf{z})}\, \phi_{\Delta-1}\left(J_{\mathbf{x}}(\mathbf{h})(A\mathbf{x})\right).$$

Let $h_1, \ldots, h_{R'}$ be a transcendence basis of $\mathbf{h}$. As the size of $f$ is $s$, every entry of any $|R'| \times |R'|$ sub-matrix of $J_{\mathbf{x}}(\mathbf{h})$ is a polynomial with sparsity and degree at most $s$. So, the determinant of any such sub-matrix is a polynomial with sparsity at most $R'! \cdot s^{R'} \leq R! \cdot s^R$ and degree at most $sR$. Hence, from Theorem 3.3, $\mathcal{G}^{SV}_{\left(\lceil \log(R! \cdot s^R) \rceil + 1\right)} = \mathcal{G}^{SV}_{(O(R(\log R + \log s)))}$ is a hitting set generator for orbits of these determinants. Thus, we can put $\phi_{\Delta-1} = \mathcal{G}^{SV}_{(O(R(\log R+\log s)))}$. We then repeatedly use Lemma 4.1 to construct $\phi_2$. At level $\ell$ of the recursion, we add at most $r_\ell + 1 \leq R + 1$ many new variables for a total of at most $(\Delta - 2)(R+1)$ new variables. Also, notice that at level $\ell$, the polynomial that we add to $\phi_{\ell+1}$ to create $\phi_\ell$ has degree at most $nr_\ell + 1 \leq nR + 1$. Thus, there exists a homomorphism $\psi$ in at most $(\Delta - 2)(R+1)$ variables and of degree at most $nR + 1$, such that $\mathcal{G}^{SV}_{(O(R(\log R+\log s)))} + \psi$ is a hitting set generator for $\text{orb}(f)$. We are now ready to prove Theorem 1.4.

### 4.3.3 Proof of Theorem 1.4

A non-zero polynomial $f \in \mathcal{C}$ is computed by a $(\Delta, k, s)$ formula. Then, $f$ is also computed by a $(\Delta, k, (2s)^\Delta)$ formula in the canonical form of Claim 4.3. There are two cases:

**Case 1:** The top most gate of the formula is a $+$ gate. If $f$ is constant, then so is every polynomial in $\text{orb}(f)$. In this case, the set containing any point in $\mathbb{F}^n$ is a hitting set for

orb$(f)$; so we will assume that $f$ is not constant. Then, there exists a $x_i$ such that $\frac{\partial f}{\partial x_i} \neq 0$ (as $\text{char}(\mathbb{F}) > (2ks)^{\Delta^3 R} \geq D$) and as argued in Section 4.3.1, $\frac{\partial f}{\partial x_i}$ can be computed by a $(\Delta, (2k)^{\Delta/2}, (2k)^{\Delta/2}(2s)^{\Delta})$ formula with $+$ gate at the top and top fan-in bounded by $k$. Moreover, if $\mathcal{G}$ is a hitting set generator for orb $\left(\frac{\partial f}{\partial x_i}\right)$, then since $\text{char}(\mathbb{F}) > (2ks)^{\Delta^3 R} \geq (nR+1)D$, $\widetilde{\mathcal{G}} = \mathcal{G} + \mathcal{G}_1^{SV}$ is a hitting set generator for orb$(f)$. As $\text{char}(\mathbb{F}) = 0$ or $> (2ks)^{\Delta^3 R}$, Lemma 4.1 works, since the degree of polynomials computed by gates in $f$ can be at most $\left((2k)^{\Delta/2}(2s)^{\Delta}\right)^{\Delta} \leq (2ks)^{\Delta^3}$. Thus, as shown in Section 4.3.2, there exists a $\mathcal{G}$ that has at most

$$O\left(R\left(\log R + \log\left((2k)^{\Delta/2}(2s)^{\Delta}\right)\right)\right) + (\Delta-2)(R+1) = O\left(R\left(\log R + \Delta\log k + \Delta\log s\right) + \Delta R\right)$$

many variables and of degree $nR+1$. As $G_1^{SV}$ has 2 variables and is of degree $n$, the number of variables in $\widetilde{\mathcal{G}}$ is $O\left(R\left(\log R + \Delta\log k + \Delta\log s\right) + \Delta R\right)$ and its degree is $nR+1$. Thus, for any $A \in \text{GL}(n, \mathbb{F})$, and $g(\mathbf{x}) := f(A\mathbf{x})$, $g(\widetilde{\mathcal{G}})$ is a polynomial in

$$O\left(R\left(\log R + \Delta\log k + \log s\right) + \Delta R\right)$$

variables and of degree at most $(nR+1)D$. So, a hitting set for $g$ can be constructed in time $(nRD)^{O(R(\log R + \Delta\log k + \Delta\log s) + \Delta R)}$.

**Case 2:** The top most gate of the formula is a $\times$人 gate. Then, all inputs to this gate are computed by $(\Delta-1, k, (2s)^{\Delta})$ formulas in the canonical form of Claim 4.3 and with a $+$ gates at the top. Hence, all inputs of $f$ are in Case 1.

The proof for the case where the leaves are labelled by $b$-variate polynomials is similar; all we need to do is observe that $\mathcal{G}_{Rb}^{SV}$ is a hitting set generator for $b$-variate polynomials. So, we can use $\mathcal{G} = \mathcal{G}_{Rb}^{SV} + \psi$.

## 4.4 Hitting sets for the orbits of occur-once formulas

**The strategy.** We prove Theorem 1.5 by building upon the arguments in Shpilka and Volkovich [SV15] and linking it with Theorem 3.1. At first, we show two structural results (Lemma 4.3 and 4.4) for occur-once formulas. These lemmas are generalizations of similar structural results for read-once formulas shown in [SV15]. Much like in [SV15], the structural results help us show that for a "typical" occur-once formula $f$ with a $+$ gate as the root node, there exists a variable $x_i$ such that $\frac{\partial f}{\partial x_i}$ is a product of occur-once formulas, each of which has at most half as many non-constant leaves as $f$. We then use this fact to show that a hitting-set generator for orb$(f)$ can be constructed from a generator for orb $\left(\frac{\partial f}{\partial x_i}\right)$. [SV15] uses the

derivatives of $f$ in a similar way to show that a generator for $f$ can be constructed from that for $\frac{\partial f}{\partial x_i}$ using the SV generator (see Definition 2.20). However, in our case, we want a generator for $\text{orb}(f)$ and not just for $f$. For this reason, we first use the chain rule for derivatives to relate the gradient of a $g \in \text{orb}(f)$ with that of $f$, and then argue that there exists a $x_j$ such that a generator for $\text{orb}\left(\frac{\partial f}{\partial x_i}\right)$ is also a generator for $\frac{\partial g}{\partial x_j}$. Finally, we use this generator for $\frac{\partial g}{\partial x_j}$ to construct a generator for $g$. The argument then proceeds by induction on the number of non-constant leaves. In the base case, we need a hitting set generator for orbits of sparse polynomials which we get from Theorem 3.1.

**Notations.** Assume, without loss of generality, that none of the edge labels of an occur-once formula is zero. We will identify an occur-once formula with the polynomial $f$ it computes and define the width of $f$ - denoted by $\text{width}(f)$ - to be the number of non-constant sparse polynomials at the leaves of the formula. Observe that if $\text{width}(f) \geq 1$, then $f$ is not a constant. As mentioned above, we reduce the problem of finding a hitting set generator for $\text{orb}(f)$ to that of finding a generator for $\text{orb}(\frac{\partial f}{\partial x_i})$, where $x_i$ is such that $\frac{\partial f}{\partial x_i}$ is a product of occur-once formulas of widths at most $\frac{\text{width}(f)}{2}$; this is done in Theorem 4.1.

We start by proving two structural results.

### 4.4.1 Structural results

We will call an occur-once formula an *s-sparse occur-once formula* if the leaves of the formula are labelled by $s$-sparse polynomials. Without loss of generality, assume that an $s$-sparse occur-once formula is layered with all the leaves appearing in layer 0. If a gate appears in layer $k$, then the depth of the occur-once formula rooted at the gate is $k+2$. We will also identify a gate with the occur-once formula rooted at the gate.

**Lemma 4.3** *Let $f$ be an $s$-sparse occur-once formula having $\text{width}(f) \geq 2$. Then, $f$ can be expressed in one of the following three forms:*

1. $f = \alpha(f_1 + f_2) + \beta$,
2. $f = \alpha(f_1 \cdot f_2) + \beta$,
3. $f = \alpha f_1^e + \beta$,

*where $\alpha, \beta \in \mathbb{F}$, $\alpha \neq 0$ and $f_1, f_2$ are non-constant, variable disjoint, $s$-sparse occur-once formulas. Further, $\text{width}(f_1) + \text{width}(f_2) = \text{width}(f)$ in the first two forms, and $\text{width}(f_1) = \text{width}(f)$ and $\text{depth}(f_1) < \text{depth}(f)$ in the third form.*

**Proof:** Let the depth of $f$ be $\Delta$, which equals the number of layers in $f$ plus 1. Let $h$ be any gate in $f$ in layer 1 (i.e., the layer just above the leaves) and $\text{width}(h) \geq 2$. If $h$ is a $+$ gate, then it can be expressed in form 1. If $h$ is a $\times$人 gate, then it can be written in form 2.

Assume, by the way of induction, that the lemma is true for all gates $h'$ in $f$ of $\text{width}(h') \geq 2$ and at layers less than $k$ for some $1 < k \leq \Delta - 2$. Let $h$ be a gate in the $k$-th layer with $\text{width}(h) \geq 2$. There are two cases:

**Case 1:** $h$ is a $+$ gate, say $h = \alpha_1 h_1 + \cdots + \alpha_m h_m$. Clearly, if at least two of its children are non-constants, then $h$ is in form 1. On the other hand, if only one child, say $h_1$, is a non-constant, then $\text{width}(h_1) = \text{width}(h) \geq 2$. As $h_1$ is in layer $k-1$, from the induction hypothesis, it can be written in one of the three forms with the corresponding constants $\alpha$ and $\beta$. Then, by adding $\alpha_2 h_2 + \cdots + \alpha_m h_m$ (which is a constant) to $\alpha_1 \beta$ and multiplying $\alpha_1$ by $\alpha$, $h$ can also be written in the same form.

**Case 2**: $h$ is a $\times$人 gate, say $h = {h_1}^{e_1} \cdots {h_m}^{e_m}$. Clearly, if at least two of its children are non-constants, then $h$ is in form 2. On the other hand, if only one child, say $h_1$, is a non-constant, then $\text{width}(h_1) = \text{width}(h) \geq 2$. In this case, by taking $\alpha = {h_2}^{e_2} \cdots {h_m}^{e_m}$ (which is a constant), and observing that $\text{depth}(h_1) = k - 1 + 2 < k + 2 = \text{depth}(h)$, we see that $h$ is in form 3. □

**Lemma 4.4** *Let $f$ be an $s$-sparse occur-once formula. Then for any $i \in [n]$, $\frac{\partial f}{\partial x_i}$ is a product of $s$-sparse occur-once formulas of widths at most* $\text{width}(f)$.

**Proof:** Let the depth of $f$ be $\Delta$. Notice that the lemma is true for all the leaves (i.e., at layer 0) of $f$ as any derivative of an $s$-sparse polynomial is also an $s$-sparse polynomial. Assume, by the way of induction, that the lemma is true for all gates at layers less than $k$, for some $1 \leq k \leq \Delta - 2$ and let $h$ be any gate in the $k$-th layer of $f$. There are two cases:

**Case 1:** $h$ is a $+$ gate, say $h = \alpha_1 h_1 + \cdots + \alpha_m h_m$. As $f$, and hence $h$, is an $s$-sparse occur-once formula, we can assume without loss of generality that $x_i$ appears only in $h_1$, if it appears at all. Then, $\frac{\partial h}{\partial x_i} = \alpha_1 \frac{\partial h_1}{\partial x_i}$. From the induction hypothesis, $\frac{\partial h_1}{\partial x_i}$ is a product of $s$-sparse occur-once formulas of widths at most $\text{width}(h_1) \leq \text{width}(h)$, and so, the lemma is true for $h$.

**Case 2:** $h$ is a $\times$人 gate, say $h = {h_1}^{e_1} \cdots {h_m}^{e_m}$. As, in the previous case, assume that $x_i$ appears only in $h_1$. Then,

$$\frac{\partial h}{\partial x_i} = e_1 \cdot {h_1}^{e_1-1} \cdot {h_2}^{e_2} \cdot \ \cdots \ \cdot {h_m}^{e_m} \cdot \frac{\partial h_1}{\partial x_i}.$$

From the induction hypothesis, $\frac{\partial h_1}{\partial x_i}$ is a product of $s$-sparse occur-once formulas of widths at most $\text{width}(h_1) \leq \text{width}(h)$. Moreover, ${h_1}^{e_1-1}, {h_2}^{e_2}, ..., {h_m}^{e_m}$ are also $s$-sparse occur-once formulas of widths at most $\text{width}(h)$. Thus, the lemma is true for $h$. □

### 4.4.2 Proof of Theorem 1.5

We now show the existence of an efficient hitting set generator for orbits of occur-once formulas.

**Theorem 4.1** *Let $f$ be an $n$-variate, degree-$D$ polynomial that is computable by an $s$-sparse occur-once formula, and $g \in \text{orb}(f)$. Also, let $|\mathbb{F}| > nD$ and $\text{char}(\mathbb{F}) = 0$ or $> D$. Then for any $t \geq \log(\text{width}(f))$, $g \neq 0$ implies $g\left(\mathcal{G}^{SV}_{(\lceil \log s \rceil + 1 + t)}\right) \neq 0$. In fact, if $g$ is not a constant, then neither is $g\left(\mathcal{G}^{SV}_{(\lceil \log s \rceil + 1 + t)}\right)$.*

**Proof:** Notice that if $g$ is a non-zero constant, then $g\left(\mathcal{G}^{SV}_{(\lceil \log s \rceil + 1 + t)}\right) \neq 0$ for all $t$. So, to prove the theorem, we need to show that if $g$ is not a constant, then neither is $g\left(\mathcal{G}^{SV}_{(\lceil \log s \rceil + 1 + t)}\right)$.

Let $h$ be an $s$-sparse occur-once formula satisfying $\text{width}(h) = 1$. Then, $h$ must be of the form

$$\alpha_m \left( \cdots \left( \alpha_2 \left( \alpha_1 p(\mathbf{x})^{e_1} + \beta_1 \right)^{e_2} + \beta_2 \right) \cdots \right)^{e_m} + \beta_m,$$

where $p(\mathbf{x})$ is an $s$-sparse polynomial, $e_1, ..., e_m \in \mathbb{N}$, $\alpha_1, ..., \alpha_m \in \mathbb{F} \setminus \{0\}$ and $\beta_1, ..., \beta_m \in \mathbb{F}$. Let $A \in \text{GL}(n, \mathbb{F})$. If $h(A\mathbf{x})$ is not a constant, then neither is $p(A\mathbf{x})$. Thus, from Theorem 3.3 and the fact that $\text{Img}(\mathcal{G}^{SV}_k) \subseteq \text{Img}(\mathcal{G}^{SV}_{k+1})$ for any $k \geq 0$, we have that $p\left(A\mathcal{G}^{SV}_{(\lceil \log s \rceil + 1 + t)}\right)$ is not a constant for any $t \geq 0$. Hence, $h\left(A\mathcal{G}^{SV}_{(\lceil \log s \rceil + 1 + t)}\right)$ is also not a constant for any $t \geq 0$.

Assume, by the way of induction, that the theorem is true for all $g'$ such that $g' \in \text{orb}(f')$ for some $n$-variate, degree-$D$, $s$-sparse occur-once formula $f'$ with $1 \leq \text{width}(f') < \ell \leq \text{width}(f)$. Let $h$ be an $n$-variate, degree-$D$, $s$-sparse occur-once formula having $\text{width}(h) = \ell \geq 2$, and $A \in \text{GL}(n, \mathbb{F})$. From Lemma 4.3, there are three cases,

**Case 1:** $h = \alpha(h_1 + h_2) + \beta$. Then, we can assume without loss of generality that $\text{width}(h_1) \leq \frac{\text{width}(h)}{2} = \frac{\ell}{2}$, as $\text{width}(h_1) + \text{width}(h_2) = \text{width}(h)$. Since $h_1$ is not a constant, there exists an $i \in [n]$ such that $\frac{\partial h_1}{\partial x_i} \neq 0$ (because $\text{char}(\mathbb{F})$ is 0 or $> D$). As $\frac{\partial h}{\partial x_i} = \alpha \cdot \frac{\partial h_1}{\partial x_i}$ ($h_1$ and $h_2$ being variable disjoint) and $\alpha \neq 0$, $\frac{\partial h}{\partial x_i} \neq 0$. Now, from Lemma 4.4, $\frac{\partial h_1}{\partial x_i}$ is a product of $s$-sparse occur-once formulas of width at most $\frac{\ell}{2}$. Then, from the induction hypothesis,

$\frac{\partial h}{\partial x_i}\left(A\mathcal{G}^{SV}_{(\lceil \log s\rceil+1+t)}\right) \neq 0$ for any $t \geq \log \ell - 1$. Let $q = h(A\mathbf{x})$. The gradient of a polynomial $p(\mathbf{x})$, denoted by $\nabla p$, is the column vector $\left(\frac{\partial p}{\partial x_1}\ \frac{\partial p}{\partial x_2}\ \ldots\ \frac{\partial p}{\partial x_n}\right)^T$. By the chain rule of differentiation,

$$\nabla q = A^T \cdot [\nabla h](A\mathbf{x}).$$

As $A^T$ is invertible, there exists a $j \in [n]$ such that $\frac{\partial q}{\partial x_j}\left(\mathcal{G}^{SV}_{(\lceil \log s\rceil+1+t)}\right) \neq 0$ for any $t \geq \log \ell - 1$. This means, by Observation 2.2, $q(\mathcal{G}^{SV}_{(\lceil \log s\rceil+1+t)})$ is not a constant for any $t \geq \log \ell$ (as $\deg(q) \leq D$ and $|\mathbb{F}| > nD$). In other words, $h(A\mathcal{G}^{SV}_{(\lceil \log s\rceil+1+t)})$ is not a constant for any $t \geq \log \ell$.

**Case 2:** $h = \alpha(h_1 \cdot h_2) + \beta$. As $\text{width}(h_1), \text{width}(h_2) < \text{width}(h)$, from the induction hypothesis, we have that for any $t \geq \log \ell$, $h_1\left(A\mathcal{G}^{SV}_{(\lceil \log s\rceil+1+t)}\right), h_2\left(A\mathcal{G}^{SV}_{(\lceil \log s\rceil+1+t)}\right)$ are not constants and so neither is $h\left(A\mathcal{G}^{SV}_{(\lceil \log s\rceil+1+t)}\right)$.

**Case 3:** $h = \alpha h_1^e + \beta$. In this case, $\text{width}(h_1) = \text{width}(h) = \ell \geq 2$, but $\text{depth}(h_1) < \text{depth}(h)$. As $h\left(A\mathcal{G}^{SV}_{(\lceil \log s\rceil+1+t)}\right)$ is not a constant if and only if $h_1\left(A\mathcal{G}^{SV}_{(\lceil \log s\rceil+1+t)}\right)$ is not a constant, the problem reduces to showing that for any $g_1 \in \text{orb}(h_1)$, $g_1\left(\mathcal{G}^{SV}_{(\lceil \log s\rceil+1+t)}\right)$ is not a constant for any $t \geq \log \ell$. We now run the argument from the beginning with $h$ replaced by $h_1$, which has a smaller depth. Eventually, we will land up in Case 1 or 2, as a depth-3 occur-once formula having width $\geq 2$ is either in form 1 or 2 (see proof of Lemma 4.3). □

A non-zero polynomial $f \in \mathcal{C}$ is computable by an $s$-sparse occur-once formula. Observe that $\text{width}(f) \leq n$. Let $g \in \text{orb}(f)$. From Theorem 4.1, we have that $g\left(\mathcal{G}^{SV}_{(\lceil \log s\rceil+1+\lceil \log n\rceil)}\right)$ is a non-zero polynomial in $2(\lceil \log s\rceil + 1 + \lceil \log n\rceil)$ variables of degree at most $nD$. As $|\mathbb{F}| > nD$, a hitting set for $\text{orb}(\mathcal{C})$ can be computed in time $(nD+1)^{2(\lceil \log s\rceil+1+\lceil \log n\rceil)} = (nD)^{O(\log n + \log s)}$.

The proof is similar if the leaves of the occur-once formulas in $\mathcal{C}$ are labelled by $b$-variate polynomials. We just need to apply Observation 2.1 instead of Theorem 3.3 in the base case.

# Chapter 5

# Equivalence test for read-once arithmetic formulas

This chapter gives an equivalence test for read-once arithmetic formulas. The contents of this chapter are from a joint work with Nikhil Gupta and Chandan Saha [GST23]. A read-once arithmetic formula is said to be *regular* if no $+$ gate in it has a variable directly connected to it. An equivalence test for the special case of regular read-once arithmetic formulas can be found in [Gup22].

## 5.1 Introduction

We say that $f, g \in \mathbb{F}[\mathbf{x}]$ are equivalent polynomials, denoted $f \sim g$, if there exists an $A \in \mathrm{GL}(|\mathbf{x}|, \mathbb{F})$ and $\mathbf{b} \in \mathbb{F}^{|\mathbf{x}|}$ such that $g(\mathbf{x}) = f(A\mathbf{x} + \mathbf{b})$ (see Definition 2.31). The equivalence test problem for a circuit class $\mathcal{C}$ is as follows: given a $g \in \mathbb{F}[\mathbf{x}]$ check if there exists an $f \in \mathbb{F}[\mathbf{x}]$ computed by a circuit in $\mathcal{C}$ such that $g \sim f$. If yes, find $A \in \mathrm{GL}(|\mathbf{x}|, \mathbb{F})$ and $\mathbf{b} \in \mathbb{F}^{|\mathbf{x}|}$ such that $g(\mathbf{x}) = f(A\mathbf{x} + \mathbf{b})$. This section is devoted to designing and analyzing a randomized polynomial time algorithm for equivalence test for ROFs (see Definition 2.4). In particular, we prove the following theorem.

**Theorem 1.6 (ET for ROFs)** *Let $n \in \mathbb{N}$, $\mathrm{char}(\mathbb{F}) = 0$ or $\geq n^2$, and $|\mathbb{F}| \geq n^{13}$. There is a $\mathrm{poly}(n)$ time randomized algorithm (with oracle access to QFE over $\mathbb{F}$) that takes input black-box access to an $n$-variate polynomial $f \in \mathbb{F}[\mathbf{x}]$, which is in the orbit of an unknown canonical ROF $\mathtt{C}$, and outputs (with high probability) an $A \in \mathrm{GL}(n, \mathbb{F})$ such that $f(A\mathbf{x}) = \mathtt{C}(PS\mathbf{x} + \mathbf{b})$, where $P \in M(n, \mathbb{F})$ and $S \in M(n, \mathbb{F})$ are permutation and scaling (i.e., diagonal) matrices respectively, and $\mathbf{b} \in \mathbb{F}^n$.*

Quadratic Form Equivalence can be solved efficiently over $\mathbb{C}, \mathbb{R}, \mathbb{F}_q$ and also over $\mathbb{Q}$ with oracle access to integer factoring (see Fact 5.3). Hence, ET for ROFs can also be solved efficiently over these fields.

Note that as $\mathtt{C}(PS\mathbf{x} + \mathbf{b})$ is an ROF, we can apply any of the known polynomial-time ROF reconstruction algorithms [HH91, BHH95, SV14, MV18] to first get an ROF for $\mathtt{C}(PS\mathbf{x} + \mathbf{b})$, and then obtain a formula for $f$ by applying $A^{-1}$ on the variables of the reconstructed ROF. We present a randomized polynomial-time ROF reconstruction algorithm in Section 5.6 as we need to use some of its properties for designing an algorithm for the polynomial equivalence problem for orbits of ROFs in Chapter 6.

We first give a high-level overview of our proof of Theorem 1.6 in Section 5.2. Then we prove some preliminary results in Section 5.3 and analyse the Hessian determinant of an ROF in Section 5.4; the results of both of these sections are crucially used to give an equivalence test for ROFs in Section 5.5. Finally we give an algorithm for reconstructing ROFs in Section 5.6.

## 5.2 Proof techniques

**First, an example.** The algorithm in Theorem 1.6 is based on a few crucial properties of the Hessian determinant of an ROF (see Definition 2.27). The effectiveness of the Hessian, in this context, is best demonstrated by an equivalence test for the sum-product polynomial $\mathsf{SP} := \sum_{i\in[s]} \prod_{j\in[d]} x_{i,j}$, which is an ROF of product-depth 1. Assume that $d \geq 3$. The algorithm takes input an $f = \mathsf{SP}(B\mathbf{x})$, where $B \in \mathrm{GL}(sd, \mathbb{F})$ is unknown. It computes the Hessian determinant of $f$, which is denoted as $\det(H_f)$. By Fact 2.4, $\det(H_f)$ is a non-zero $\mathbb{F}$-multiple of $\det(H_{\mathsf{SP}})(B\mathbf{x})$ – the Hessian determinant of $\mathsf{SP}$ evaluated at $B\mathbf{x}$. Now, it can be shown that $\det(H_{\mathsf{SP}})$ factorizes as follows:

$$\det(H_{\mathsf{SP}}) = (-1)^{s(d-1)} \cdot (d-1)^s \cdot \prod_{i\in[s], j\in[d]} x_{i,j}^{d-2}.$$

So, the algorithm factorizes $\det(H_f)$ into irreducible factors and figures out[1] $B$ from the factors. The test can be implemented in the black-box setting by observing that black-box access to the second-order partials of $f$ can be computed efficiently (see Fact 5.1) and by invoking a black-box polynomial factorization algorithm (see Fact 5.2). The running time is polynomial in $s$ and $d$.

[1]The algorithm finds an $A = PSB$, where $P$ is a permutation matrix and $S$ is a diagonal matrix, from the factors of $\det(H_f)$. It then interpolates $f(A^{-1}\mathbf{x})$ (using the sparse polynomial interpolation algorithm in [KS01]) to learn $P$ and $S$ (up to the symmetries of the polynomial $\mathsf{SP}$).

### 5.2.1 A basic approach

Can the Hessian determinant be exploited to devise an equivalence test for ROFs of *arbitrary* product-depth and fan-in? A rudimentary approach is outlined in [Kay11]: Let $g = g_1(x_1, \ldots, x_i) + g_2(x_{i+1}, \ldots, x_n)$, where $g_1$ and $g_2$ are variable disjoint polynomials. Given black-box access to $f = g(B\mathbf{x})$, where $g$ and $B \in \mathrm{GL}(n, \mathbb{F})$ are unknown, can we find an $A \in \mathrm{GL}(n, \mathbb{F})$ such that $f(A\mathbf{x})$ can be expressed as a sum of two variable disjoint polynomials?[1] [Kay11] gave an algorithm that finds such an $A$ *provided the number of essential variables*[2] *of* $\det(H_g)$ *is exactly* $n$.

The algorithm uses the fact that $\det(H_g) = \det(H_{g_1})(x_1, \ldots, x_i) \cdot \det(H_{g_2})(x_{i+1}, \ldots, x_n)$, and so, $\det(H_f)$ is a non-zero $\mathbb{F}$-multiple of $\det(H_{g_1})(B\mathbf{x}) \cdot \det(H_{g_2})(B\mathbf{x})$. It turns out that an $A \in \mathrm{GL}(n, \mathbb{F})$ can be found efficiently from black-box access to $\det(H_f)$ such that $\det(H_{g_1})(BA\mathbf{x})$ and $\det(H_{g_2})(BA\mathbf{x})$ are variable disjoint; this step involves black-box factorization of $\det(H_f)$ [KT90] and elimination of redundant variables from the irreducible factors of $\det(H_f)$ in a careful way (see Claim 5.3). Now, it can also be shown that if the number of essential variables of $\det(H_g)$ is exactly $n$, then $g_1(BA\mathbf{x})$ and $g_2(BA\mathbf{x})$ are variable disjoint (see Observation 2.8).

The correctness of the algorithm depends critically on the condition that the number of essential variables of $\det(H_g)$ is *exactly* $n$. If this condition does not hold, then the algorithm fails completely. The approach can be viewed as a generalization of the algorithm given in the above example for the SP polynomial. Indeed, the number of essential variables of $\det(H_{\mathsf{SP}})$ is $n = sd$.

**Can the basic approach be used to learn orbits of ROFs?** At a high level, the basic approach is encouraging as a $+$-rooted ROF is a sum of variable disjoint polynomials. Let $\mathtt{C} = T_1 + \ldots + T_s$ be a $+$-rooted canonical ROF, where $T_1, \ldots, T_s$ are the *terms* of $\mathtt{C}$, i.e., the polynomials computed by the second (from the top) layer of gates in $\mathtt{C}$. Given black-box access to $f = \mathtt{C}(B\mathbf{x}) = T_1(B\mathbf{x}) + \ldots + T_s(B\mathbf{x})$, where $\mathtt{C}$ and $B \in \mathrm{GL}(n, \mathbb{F})$ are unknown, we hope to apply the approach in [Kay11] to find an $A \in \mathrm{GL}(n, \mathbb{F})$ such that $T_1(BA\mathbf{x}), \ldots, T_s(BA\mathbf{x})$ are *variable disjoint*. If we succeed in finding $A$, then we wish to obtain *efficient black-box access* to $T_1(BA\mathbf{x}), \ldots, T_s(BA\mathbf{x})$ by exploiting their variable disjointness. From black-box access to $T_i(BA\mathbf{x})$, we get black-box access to $Q_{i,1}(BA\mathbf{x}), \ldots, Q_{i,m_i}(BA\mathbf{x})$, where $Q_{i,1}, \ldots, Q_{i,m_i}$ are the

[1] This problem was referred to as the *polynomial decomposition* problem in [Kay11]. It should not be confused with the functional decomposition of polynomials which is also known as the polynomial decomposition problem.

[2] See Definition 2.28.

irreducible factors of $T_i$. Claim 5.2 then lets us find a $C \in \mathrm{GL}(n, \mathbb{F})$ such that $Q_{i,1}(BAC\mathbf{x}), \ldots, Q_{i,m_i}(BAC\mathbf{x})$, for all $i \in [s]$, are variable disjoint. At this point, we plan to recurse on $Q_{i,1}(BAC\mathbf{x}), \ldots, Q_{i,m_i}(BAC\mathbf{x})$ that are in the orbits of variable disjoint $+$-rooted ROFs of smaller size and depth.

Although the method looks promising, there are a few significant hurdles that render the basic approach almost useless. First, we shall see (in the next section) that the number of essential variables of the Hessian determinant of a canonical ROF can be dramatically smaller than $n$, although the ROF itself has no redundant variable. This is indeed a serious problem for the approach as the step of making $T_1(BA\mathbf{x}), \ldots, T_s(BA\mathbf{x})$ variable disjoint may break down completely. Second, even if we manage to make $T_1(BA\mathbf{x}), \ldots, T_s(BA\mathbf{x})$ variable disjoint, the complexity of the recursive algorithm may grow *exponentially* with the product depth of the ROF unless we generate super-efficient black-box access to $T_1(BA\mathbf{x}), \ldots, T_s(BA\mathbf{x})$. In the next section, we elaborate on these (and more) hurdles and explain how we overcome them and salvage the basic approach.

### 5.2.2 Outline of the ROF equivalence test: Salvaging the basic approach

Without loss of generality, assume that the root node of an ROF $\mathtt{C}$ is a $+$-gate; if not, use black-box polynomial factorization to reduce to the $+$-rooted case. We need to answer two questions:

1. How do we efficiently find a transformation that makes the terms variable disjoint?
2. How do we get efficient black-box access to the terms once they are variable disjoint?

We now elaborate on the technical hurdles that we encounter and deal with while answering these.

#### A. Making the terms variable disjoint

We know that the terms can be made variable disjoint if the number of essential variables in $\det(H_{\mathtt{C}})$ is exactly $n$. But this need not be the case. In fact, we face an even more basic hurdle.

- Hurdle 1: The Hessian determinant of a non-zero canonical ROF can be identically zero.

For instance, the Hessian determinant of $(x_1x_2 + x_3x_4)(x_5x_6 + x_7x_8) + (y_1y_2 + y_3y_4)(y_5y_6 + y_7y_8)$ is identically zero over $\mathbb{F}_3$; the Hessian determinant of $x_1x_2x_3 + x_4$ is zero over any $\mathbb{F}$.

None of these ROFs have redundant variables (see Observation 5.2), and yet their Hessian determinants are zero.

*When is the Hessian determinant non-zero?* We show in Lemma 5.1 that the Hessian determinant of a non-zero $n$-variate canonical ROF $\mathtt{C}$ is non-zero provided $\mathrm{char}(\mathbb{F}) = 0$ or $\geq n$ and none of the children of the top $+$-gate of $\mathtt{C}$ is a variable[1]. Henceforth, we assume that $\mathrm{char}(\mathbb{F}) = 0$ or $\geq n$. We call a variable that is directly connected to a $+$-gate a *dangling variable*, and a variable that is directly connected to the top $+$-gate the *top dangling variable*. Since $\mathtt{C}$ is in canonical form (see Definition 2.23), it can have at most one top dangling variable.

*Few words on the proof of Lemma 5.1:* We show that the coefficient of a certain high degree monomial in $\det(H_{\mathtt{C}})$ is a product of "small", non-zero numbers. Depending on the structure of $\mathtt{C}$, we first carefully pick a variable $x$ in it and treat $\det(H_{\mathtt{C}})$ as a univariate polynomial over $\mathbb{F}[\mathbf{x} \setminus \{x\}]$. We then show that the coefficient of the highest degree term in $x$ is a product of the Hessian determinants of "smaller", $\times$-rooted ROFs. We repeat this process inductively on these smaller ROFs to show that their Hessian determinants are non-zero. The inductive process constructs a high degree monomial implicitly. The base case of the induction deals with Hessians of monomials of degree at least 2. The Hessian determinant of a degree $d$ monomial is itself a monomial with a non-zero coefficient. Thus, the coefficient of the special monomial constructed by the inductive process is a product of the coefficients of the Hessian determinants of monomials.

The presence of a top dangling variable makes $\det(H_{\mathtt{C}})$ zero. We will see shortly how to prevent $\det(H_{\mathtt{C}})$ from vanishing. At first, let us assume that $\mathtt{C}$ has *no* top dangling variable. Now, even if the Hessian determinant of $\mathtt{C}$ is non-zero, there is no guarantee that the number of essential variables of $\det(H_{\mathtt{C}})$ is the maximum possible. This poses the second and the main hurdle.

- Hurdle 2: The number of essential variables of $\det(H_{\mathtt{C}}) \neq 0$ can be much smaller than $n$.

For example, the Hessian determinant of $x_1(x_2x_3 + x_4) + y_1(y_2y_3 + y_4)$ has merely two essential variables; the Hessian determinant of $x_1x_2x_3 + x_4x_5$ has only three essential variables. For ease of explanation, we split the above hurdle into two questions. The first one is,

[1]Observe that if the top $+$-gate has a variable child, then the Hessian determinant is identically zero over any $\mathbb{F}$.

- Hurdle 2a: Which variables of a canonical ROF $\mathtt{C}$ are essential for its Hessian determinant?

The notion of "skewed paths" turns out to be quite useful in answering this question.

*Skewed paths, truly essential variables, and good and bad terms:* A *skewed path* in $\mathtt{C}$ is a special structure that can be identified with a unique "marker" monomial (see Definition 5.1). In Claim 5.8, we show that every variable other than the dangling variables along skewed paths, the variables in quadratic forms along skewed paths, and the variables in the (top) quadratic form of $\mathtt{C}$, are *truly essential* for $\det(H_{\mathtt{C}})$ (see Definition 2.29)[1]. This knowledge enables us to categorize the terms of $\mathtt{C}$ into three types – *good*, *bad*, and the quadratic form of $\mathtt{C}$. A bad term looks like $x \cdot Q$, where $x \in \mathbf{x}$ and $Q$ is a $+$-rooted ROF. In the example, both $x_1(x_2x_3 + x_4)$ and $y_1(y_2y_3 + y_4)$ are bad terms. In $x_1(x_2x_3 + x_4)$, the "marker" monomial $x_1$ (which is a variable in this simple case) defines a skewed path, $x_2$ and $x_3$ are the variables of the quadratic form along this skewed path, and $x_4$ is the dangling variable along this skewed path. Terms that are not bad and have degree $\geq 3$ are good.

**Making good terms variable disjoint.** If $T$ is a good (similarly, bad) term of $\mathtt{C}$, then we say $T(B\mathbf{x})$ is a good (respectively, bad) term of the input $f = \mathtt{C}(B\mathbf{x})$. It follows from Definition 5.1 that the skewed paths in $\mathtt{C}$ occur only in the bad terms of $\mathtt{C}$, and so, from Claim 5.8, all the variables of the good terms of $\mathtt{C}$ are truly essential for $\det(H_{\mathtt{C}})$. This fact along with Claim 5.3 and Observation 2.8 help us infer that a slight variant of the basic strategy given in Section 5.2.1 succeeds in finding an $A_0 \in \mathrm{GL}(n, \mathbb{F})$ such that the good terms of $f$ become variable disjoint under the action of $A_0$. See Step 1 in Section 5.5.1 and Section 5.7 for a more detailed and pictorial overview of this step.

Making the good terms of $f$ variable disjoint is the first step towards overcoming Hurdle 2. But it is far from sufficient even if $\mathtt{C}$ is devoid of bad terms, the top quadratic form, and the top dangling variable. This is because the algorithm may encounter bad terms, quadratic forms and dangling variables at deeper levels of the recursion, whence the basic strategy will fail. Therefore, we must answer the following (second) question to tackle the acute loss of essential variables in the Hessian determinant due to the presence of skewed paths in bad terms.

- Hurdle 2b: How do we handle the bad terms and the quadratic form of $\mathtt{C}$?

[1] See the paragraph before Claim 5.8 for relevant terminologies. Partitioning a set of essential variables into truly essential variables and ordinary essential variables helps us crucially in the arguments.

We call the dangling variables along skewed paths, the variables in quadratic forms along skewed paths, and the variables of the top quadratic form of $\mathsf{C}$ the *bad variables* of $\mathsf{C}$. The remaining variables are the *good variables*. Note that a good term of $\mathsf{C}$ has only good variables, whereas a bad term has both good and bad variables. For example, $x_1$ is a good variable of the bad term $x_1(x_2x_3 + x_4)$, and $x_2, x_3, x_4$ are its bad variables. By Claim 5.8, the good variables are truly essential for $\det(H_{\mathsf{C}})$, but they need not be the only essential variables. Some bad variables can be truly or ordinarily essential for $\det(H_{\mathsf{C}})$ or totally absent from $\det(H_{\mathsf{C}})$; this complicates the matter a bit.

**Making the bad terms and the top quadratic form variable disjoint.** It turns out that Claim 5.8, Claim 5.3 and Observation 2.8 together imply that the transformation $A_0$ is such that $BA_0$ maps every good variable of a (good or bad) term $T_k$ to a linear form in $\mathbf{z}_k \subseteq \mathbf{x}$, where the variable sets $\mathbf{z}_k$ (as $T_k$ runs over all good and bad terms) are disjoint. Let $\mathbf{z}$ be the disjoint union of these sets $\mathbf{z}_k$, and $\mathbf{y} := \mathbf{x} \setminus \mathbf{z}$. Let $\ell_x := BA_0 \circ x$ for $x \in \mathbf{x}$. Observe that the $\mathbf{y}$-variables appear only in the linear forms $\ell_x$ where $x$ is a bad variable. Let $[\ell_x]_{\mathbf{y}}$ be $\ell_x$ restricted to the $\mathbf{y}$-variables. Loosely speaking, we make the bad terms and the top quadratic form of $f$ variable disjoint in three (implicit) steps: "access" the linear forms $[\ell_x]_{\mathbf{y}}$, map them to distinct $\mathbf{y}$-variables, and then remove "external variables" from each of the terms. Let us elaborate on these steps by focusing on the bad terms.

*Mapping "garbled" skewed paths back to monomials to access $[\ell_x]_{\mathbf{y}}$:* How do we access $[\ell_x]_{\mathbf{y}}$, where $x$ is a variable in a quadratic form along a skewed path or a dangling variable along a skewed path? The answer lies in the fact that a skewed path is identified with a unique "marker" monomial $\mu$. This monomial can potentially help us access $[\ell_x]_{\mathbf{y}}$, where $x$ a quadratic form or a dangling variable along the skewed path $\mu$. But the problem is that the transformation $BA_0$ may have "garbled" the variables of $\mu$. If for every variable $z$ of $\mu$, we find $\ell_z \in \mathbb{F}[\mathbf{z}]$, then we can map $\ell_z$ to a distinct $\mathbf{z}$-variable and get back a marker monomial – this works as $z$ is a good variable. By Claim 5.7, such an $\ell_z$ is a factor of $\det(H_f)(A_0\mathbf{x})$. We can factorize $\det(H_f)(A_0\mathbf{x})$ and try to find $\ell_z$, but there is a problem: $\det(H_f)(A_0\mathbf{x})$ might have other spurious linear factors that are not $\ell_z$ for any $z \in \mathbf{x}$. Fortunately, we can distinguish $\ell_z$ from spurious linear factors of $\det(H_f)(A_0\mathbf{x})$ by examining the number of essential variables of $f(A_0\mathbf{x})$ modulo affine forms; this crucial result is proved in Claim 5.1. So, we can safely assume without any loss of generality that $\ell_z = z$ for every variable $z$ in $\mu$.

*Processing quadratic forms along skewed paths and the top quadratic form:* We focus on a quadratic form $q = y_1y_2 + \ldots + y_{l-1}y_l$ along a skewed path $\mu$, and let $\widetilde{q} = [\ell_{y_1}]_{\mathbf{y}}[\ell_{y_2}]_{\mathbf{y}} + \ldots + [\ell_{y_{l-1}}]_{\mathbf{y}}[\ell_{y_l}]_{\mathbf{y}}$. We can access $\widetilde{q}$ as follows: Treat $f(BA_0\mathbf{x})$ as a polynomial in $\mathbf{y}$ over $\mathbb{F}[\mathbf{z}]$ and extract out

black-box access to the homogeneous degree-2 component in $\mathbf{y}$; call it $\widehat{q}$. As the degree-2 monomials in $\mathbf{y}$ are contributed only by the quadratic forms on skewed paths and the quadratic form of $\mathsf{C}$, and there are at most $n$ different skewed paths, $\widehat{q}$ is $n^3$-sparse as a polynomial in $\mathbb{F}[\mathbf{y}, \mathbf{z}]$. We find the dense representation of $\widehat{q}$ using the sparse polynomial interpolation algorithm of [KS01]. Observe that the coefficient of $\mu$ in $\widehat{q}$ (as a polynomial in $\mathbf{z}$ over $\mathbb{F}[\mathbf{y}]$) is $\widetilde{q}$. Once we collect *all* the $\widetilde{q}$ for quadratic forms along skewed paths, we map them simultaneously to quadratic SP polynomials in distinct $\mathbf{y}$-variables using Claim 5.2 and the QFE oracle. The existence of such a map $A_1$ is ensured by Claim 5.9 which shows that the variables of a quadratic form are either all truly essential for $\det(H_{\mathsf{C}})$ or they are absent from $\det(H_{\mathsf{C}})$. We then argue (in Claim 5.15) that $q(BA_0A_1\mathbf{x})$ can be expressed as $(y_1 + h_1)(y_2 + h_2) + \cdots + (y_{l-1} + h_{l-1})(y_l + h_l)$ for some (hitherto unknown) linear forms $h_1, \ldots, h_l \in \mathbb{F}[\mathbf{z}]$. A similar process for $\mu = 1$ takes care of the top quadratic form of $\mathsf{C}$. See Step 2.1 in Section 5.5.1 and Section 5.7 for a more detailed and pictorial overview of this step.

*Handling dangling variables on skewed paths:* Now let $\ell_x := BA_0A_1 \circ x$ and $\mathbf{u}$ be the $\mathbf{y}$-variables that have not been 'used up' by the QFE oracle in the previous step. Consider a dangling variable $x$ along a skewed path $\mu$. We can access $[\ell_x]_{\mathbf{u}}$ using $\mu$, just as we accessed $\widetilde{q}$ before, by treating $f(BA_0A_1\mathbf{x})$ as a polynomial in $\mathbf{u}$ over $\mathbb{F}[\mathbf{x} \setminus \mathbf{u}]$ and extracting out the homogeneous degree-1 component in $\mathbf{u}$. However, unlike the variables of a quadratic form along a skewed path, a dangling variable $x$ might not enjoy the property that it is either truly essential for $\det(H_{\mathsf{C}})$ or it is absent from $\det(H_{\mathsf{C}})$. So it may not be possible to map all the linear forms $[\ell_x]_{\mathbf{u}}$ for dangling variables along skewed paths to distinct $\mathbf{u}$-variables. This makes the argument here a bit subtle: We show, using Observation 5.17, Claim 5.17 and Observation 5.19, that it is sufficient to work with *any* basis $\mathcal{B}$ of the vector space spanned by the linear forms $[\ell_x]_{\mathbf{u}}$ as $x$ varies over dangling variables along skewed paths. Mapping the elements of $\mathcal{B}$ to distinct $\mathbf{u}$-variables, using a transformation $A_2$, automatically 'takes care of' the linear forms outside $\mathcal{B}$. At a high level, this strategy works because the elements of $\mathcal{B}$ essentially corresponds to a set of redundant variables of $\det(H_{\mathsf{C}})$. See Step 2.2 in Section 5.5.1 and Section 5.7 for a more detailed and pictorial overview of this step.

*Removing external variables from the terms:* Let $\ell_x := BA_0A_1A_2 \circ x$ for $x \in \mathbf{x}$. For a bad term $T_k$, let $\mathbf{y}_k$ be the union of the $\mathbf{y}$-variables appearing in all $\ell_x$, where $x$ is a variable of a quadratic form along a skewed path in $T_k$, and the $\mathbf{u}$-variables present in all $\ell_x$, where $x$ is a dangling variable along a skewed path in $T_k$. The variables not in $\mathbf{z}_k \uplus \mathbf{y}_k$ are the *external variables* of $T_k$. Observe that the external variables appear in $\ell_x$ only if $x$ is a dangling variable along a skewed path in $T_k$ or a variable of a quadratic form along a skewed path in $T_k$. In

this step, we intend to remove these external variables and complete the process of making the bad terms and the top quadratic form of $f$ variable disjoint. At a high level, this is done by examining some carefully chosen first-order partials of $f(BA_0A_1A_2\mathbf{x})$ and engaging the skewed paths again to access the external variables. The proof of correctness of this step involves a few "disambiguation arguments" (see Observations 5.20, 5.21 and 5.22) which ensure that relevant monomials are generated "uniquely". See Step 2.3 in Section 5.5.1 and Section 5.7 for a more detailed and pictorial overview of this step.

**Handling the top dangling variable.** If $\det(H_{\mathtt{C}}) = 0$ (which, by Lemma 5.1, happens if and only if $\mathtt{C}$ has a top dangling variable) then we can reduce to the non-zero Hessian determinant case as follows: Apply a random transformation on the variable set $\mathbf{x} = \{x_1, \ldots, x_n\}$ and consider the Hessian of the resulting $f$ with respect to only $x_1, \ldots, x_{n-1}$. Intuitively, the random transformation lets us assume two facts – one, the top dangling variable of $\mathtt{C}$ is $x_n$, i.e., $\mathtt{C} = \mathtt{C}_1(x_1, \ldots, x_{n-1}) + x_n$, where $\mathtt{C}_1$ is a canonical ROF with no top dangling variable; two, $f = \mathtt{C}_1(B\mathbf{x}) + \ell(\mathbf{x})$ for some $B \in \mathrm{GL}(n, \mathbb{F})$ such that $B \circ x_n = x_n$ and $\ell$ is an affine form. Now observe that the determinant of the Hessian of $f$ with respect to $x_1, \ldots, x_{n-1}$ is an $\mathbb{F}$-multiple of $\det(H_{\mathtt{C}_1})(B\mathbf{x})$, which is non-zero as $\mathtt{C}_1$ has no top dangling variable. We can then remove the redundant variable $x_n$ from $\det(H_{\mathtt{C}_1})(B\mathbf{x})$ and hope to find a $D \in \mathrm{GL}(n, \mathbb{F})$ such that $T_1(BD\mathbf{x}), \ldots, T_{s-1}(BD\mathbf{x})$ are variable disjoint, where $T_1, \ldots, T_{s-1}$ are the terms of $\mathtt{C}_1$. Once $D$ is obtained, we are left with finding $\ell(D\mathbf{x})$ from black-box access to $f(D\mathbf{x}) = \mathtt{C}_1(BD\mathbf{x}) + \ell(D\mathbf{x})$. Indeed, the knowledge of $D$ and $\ell(D\mathbf{x})$ is sufficient to construct an $A \in \mathrm{GL}(n, \mathbb{F})$ such that $T_1(BA\mathbf{x}), \ldots, T_s(BA\mathbf{x})$ are variable disjoint (here, $T_s = x_n$). See Step 3 in Section 5.5.1 for a more detailed overview on how to find $\ell(D\mathbf{x})$ by exploiting the Hessian determinant again! A special case of this problem when $\ell$ is a constant also arises in the resolution of the final hurdle (stated below). We give the proof idea for this special case next.

### B. Obtaining efficient black-box access to the terms

The above process finds an $A \in \mathrm{GL}(n, \mathbb{F})$ such that the terms $T_1(BA\mathbf{x}), \ldots, T_s(BA\mathbf{x})$ are variable disjoint. Let $r_i(\mathbf{x}_i) := T_i(BA\mathbf{x})$.

- Hurdle 3: How do we get *efficient* black-box access to $r_1(\mathbf{x}_1), \ldots, r_s(\mathbf{x}_s)$?

In other words, how do we simulate a black-box query to $r_i(\mathbf{x}_i)$ using *only one* query to the black-box for the input polynomial $f$. It is important to use only one query to $f$, as otherwise

the time complexity of the recursive algorithm will become exponential in the product-depth of the ROF. The product depth of an $n$-variate ROF can be as high as $\Omega(n)$. We address this issue as follows.

At first, we examine the second-order derivatives of $f$ to learn the variable sets $\mathbf{x}_1, \ldots, \mathbf{x}_s$ (see Claim 5.23). Then, we set the variables in $\mathbf{x}_1, \ldots, \mathbf{x}_{i-1}, \mathbf{x}_{i+1}, \ldots, \mathbf{x}_s$ to arbitrary field constants to reduce the problem to securing black-box access to $r_i(\mathbf{x}_i)$ from black-box access to $g_i := r_i(\mathbf{x}_i) + c$, where $c \in \mathbb{F}$ is unknown. If $r_i$ is quadratic or linear, then we simply interpolate $g_i$ and know $r_i$. Otherwise, we can still hope to learn $c$ as it is the *unique* constant such that $g_i - c$ is reducible [1]. The uniqueness of $c$ follows from the irreducibility of a $+$-rooted ROF (see Fact 2.1). But how do we learn $c$ efficiently? The Hessian determinant comes in handy again.

**Finding $c$.** Suppose $r_i(\mathbf{x}_i) = r_{i,1}(\mathbf{x}_i) \cdot \ldots \cdot r_{i,m_i}(\mathbf{x}_i)$, where $r_{i,1}, \ldots, r_{i,m_i}$ are the irreducible factors of $r_i$, and $\deg(r_i) \geq 3$. It follows from Corollary 5.1 that $\det(H_{r_i})$, which equals $\det(H_{g_i})$, has as one of its irreducible factors an $\mathbb{F}$-multiple of $r_{i,j}$ for some $j \in [m_i]$. The efficient black-box polynomial factorization algorithm [KT90] gives us black-box access to all the irreducible factors of $\det(H_{g_i})$. Now suppose we pick the irreducible factor $\alpha \cdot r_{i,j}$, where $\alpha \in \mathbb{F}^\times$, from among the irreducible factors of $\det(H_{g_i})$. Define a random substitution map $\pi$ on the variables of $\mathbf{x}_i$ as follows: $\pi(x) := c_x t$, where $c_x \in_r \mathbb{F}$ and $t$ is a fresh variable, for every $x \in \mathbf{x}_i$. Interpolate the univariate polynomials $\pi(g_i)(t)$ and $\pi(\alpha \cdot r_{i,j})(t)$ that are non-constant with high probability, if $|\mathbb{F}|$ is sufficiently large. The degrees of $\pi(g_i)$ and $\pi(\alpha \cdot r_{i,j})$ are upper bounded by $n$. To find $c$, we set up and solve a linear system via the equation $\pi(g_i) = (a_{n-1}t^{n-1} + \ldots + a_0) \cdot \pi(\alpha \cdot r_{i,j}) + c_0$, [2] by pretending that $a_{n-1}, \ldots, a_0$ and $c_0$ are variables. The system has a solution that is obtained by choosing $a_{n-1}t^{n-1} + \ldots + a_0 = \pi(\alpha^{-1} \cdot \prod_{l \in [m_i] \setminus \{j\}} r_{i,l})$ and $c_0 = c$. This solution is unique. To see this, suppose $a_{n-1,1}, \ldots, a_{0,1}, c_{0,1}$ and $a_{n-1,2}, \ldots, a_{0,2}, c_{0,2}$ are two different solutions. Then,

$$((a_{n-1,1} - a_{n-1,2})t^{n-1} + \ldots + (a_{0,1} - a_{0,2})) \cdot \pi(\alpha \cdot r_{i,j}) + (c_{0,1} - c_{0,2}) = 0,$$

indicating that $\pi(\alpha \cdot r_{i,j})$ divides $(c_{0,1} - c_{0,2})$. But this is not possible as $\pi(\alpha \cdot r_{i,j})$ is not a constant. So, we solve the above system and declare the solution for $c_0$ as $c$. This procedure works if we pick an irreducible factor of $\det(H_{g_i})$ that is an $\mathbb{F}$-multiple of $r_{i,j}$ for some

[1] More generally, this is true if $r_i$ is a multilinear polynomial having at least two non-trivial factors. But, $c$ need not be unique if $r_i$ is not multilinear. For example, if $r_i = x^2$, then $g_i - c$ is reducible for both $c = 0$ and $c = 1$.

[2] As $\deg(\alpha \cdot r_{i,j}) \geq 1$, the degree of $\alpha^{-1} \cdot \prod_{l \in [m_i] \setminus \{j\}} r_{i,l}$ is at most $n - 1$.

$j \in [m_i]$. But what if we pick a "wrong" factor? Indeed, the Hessian determinant can have other "spurious" factors. The point is that irrespective of what factor we choose, we can run the above procedure and find some $c_0$. If no $c_0$ is found, then we know immediately that a wrong factor is chosen. Otherwise, we check if $g_i - c_0$ is reducible, and if so, then take $c_0$ as $c$. The uniqueness of $c$ implies that we always find the right $c$. Once we know $c$, we can simulate a black-box query to $r_i$ using only one query to $f$.

**Preparing for recursion.** From efficient black-box access to $r_i$, we need to gain efficient black-box access to the irreducible factors of $r_i$ as the algorithm essentially recurses on these factors. This is done as follows: Use the efficient black-box polynomial factorization algorithm [KT90] to get (not necessarily efficient) black-box access to $\alpha_j \cdot r_{i,j}$ for every $j \in [m_i]$, where $\alpha_j \in \mathbb{F}^\times$ and $\alpha_1 \cdot \alpha_2 \cdot \ldots \cdot \alpha_{m_i} = 1$. Claim 5.2 then allows us to find a $C_i \in \mathrm{GL}(|\mathbf{x}_i|, \mathbb{F})$ such that $\alpha_1 \cdot r_{i,1}(C_i\mathbf{x}_i), \ldots, \alpha_{m_i} \cdot r_{i,m_i}(C_i\mathbf{x}_i)$ are variable disjoint. Notice that we can easily get efficient black-box access to $r_i(C_i\mathbf{x}_i)$ from the efficient black-box for $r_i$. It is now sufficient to create an efficient black-box for $\alpha_j \cdot r_{i,j}(C_i\mathbf{x}_i)$ from the black-box for $r_i(C_i\mathbf{x}_i)$. Substitute the variables in $\alpha_l \cdot r_{i,l}(C_i\mathbf{x}_i)$ by random field constants for every $l \in [m_i]\backslash\{j\}$; denote this substitution map by $\rho$. Let $\beta_l = \rho(\alpha_l \cdot r_{i,l}(C_i\mathbf{x}_i))$. Observe that we know $\beta_l$ from the already acquired (possibly inefficient) black-box for $\alpha_l \cdot r_{i,l}(C_i\mathbf{x}_i)$. Also, $\beta_l \neq 0$ with high probability. Then, the relation $\alpha_j \cdot r_{i,j}(C_i\mathbf{x}_i) = \rho(r_i(C_i\mathbf{x}_i)) \cdot \prod_{l\in[m_i]\backslash\{j\}} \beta_l^{-1}$ produces an efficient black-box for $\alpha_j \cdot r_{i,j}(C_i\mathbf{x}_i)$. The algorithm recurses on $\alpha_j \cdot r_{i,j}(C_i\mathbf{x}_i)$ with this black-box.

To summarize, irrespective of the level of the recursion, a required black-box can be obtained as an expression $\alpha f(C\mathbf{x} + \mathbf{c}) + \beta$, where $C \in M(n, \mathbb{F})$, $\mathbf{c} \in \mathbb{F}^n$, and $\alpha, \beta \in \mathbb{F}$ are known. Thus, the black-box query time is independent of the recursion depth. Moreover, the time taken to prepare a black-box for a subsequent level of the recursion (i.e., to make ready the knowledge of a relevant affine projection $C, \mathbf{c}$ and appropriate constants $\alpha, \beta$) is independent of the recursion depth.

## 5.3 Preliminaries

In this section, we give a detailed list of notations used in this and the next chapter and mention a few structural and algorithmic results crucially used in the equivalence test. These preliminary results are part of a joint work [GST23]. Some of the preliminary results mentioned here are also required for equivalence test for regular ROFs which can be found in [Gup22]. As such, they also appear in [Gup22].

Recall the definition of canonical ROF from Section 2.23. The following tables lists the notations used in this and the next chapter.

| Notations | Usage |
|---|---|
| $\mathsf{C}$, $\Delta$ | An ROF and its product depth, respectively |
| $T, Q$ (with or without subscripts) | $\times$-rooted and +-rooted sub-ROFs, respectively |
| $A, B, C, P, R, S$ | Matrices over $\mathbb{F}$ |
| $U, W$ | Spaces spanned by the first order partials of polynomials |
| $E, F, I, J, N, V$ | Sets |
| $f, g, h, p, q, \ell, r$ | Polynomials |
| $t, u, x, y, z$ | Variables |
| $\mathbf{x}, \mathbf{y}, \mathbf{z}, \mathbf{u}$ | Sets of variables |
| $\alpha, \beta, \gamma, c$ | Elements of $\mathbb{F}$ |
| $d, e, i, j, k, l, m, n, s$ | Natural numbers |
| $\mathbf{a}, \mathbf{b}, \mathbf{d}, \boldsymbol{\alpha}$ | Vectors over $\mathbb{F}$ |

Table 5.1: Notations

### 5.3.1 Structural preliminaries

**Observation 5.1 (Orbit of a canonical ROF)** *Let $\mathsf{C}$ be an ROF over $\mathbb{F}$. Then, there is a canonical ROF $\mathsf{C}'$ over $\mathbb{F}$ such that $\mathsf{C}' \in \mathrm{orb}(\mathsf{C})$. If $\mathsf{C}$ is additive-constant-free, then so is $\mathsf{C}'$.*

**Proof:** From the definition of a formula, the first three properties of Definition 2.23 are satisfied by $\mathsf{C}$. We now "push" the labels on the edges of $\mathsf{C}$ down to the leaves so that the variables labelling the leaves are scaled. Then, we apply an invertible diagonal transformation $S$ to $\mathbf{x}$ to rescale the variables appropriately. This ensures that property 4 is satisfied. To satisfy property 5, observe that if a $+$ gate has variable children $x_{i_1}, \ldots, x_{i_m}$ and constant children $\gamma_1, \ldots, \gamma_k$, then we can replace all the constants by $\gamma = \gamma_1 + \ldots + \gamma_k$, and apply an invertible affine transformation that maps $x_{i_1}$ to $x_{i_1} - (x_{i_2} + \cdots + x_{i_m} + \gamma)$ and every other variable to itself.

Suppose $u$ is a $+$ gate that has among its children a variable $x$ and a $\times$ gate $v$ such that $v$ has two children – a variable $y$ and a $+$ gate $v'$. Suppose $v'$ has a constant child $\gamma$. The polynomial computed at $v$ is of the form $x + y(T + \gamma) +$ other terms $= (x + \gamma y) + yT +$ other terms, where $T$ is $x$ and $y$ free. Now, if we apply an invertible linear transformation that maps $x$ to $x - \gamma y$ and every other variable to itself, then property 6 is satisfied with respect to nodes $u$ and $v$. Finally, it is easy to see that this canonization process does not introduce any extra additive-constant. □

**Observation 5.2 (Essential variables of a canonical ROF)** *The set of variables labelling the non-constant leaves of a canonical ROF $\mathsf{C}$ is the set of essential variables of $\mathsf{C}$, i.e., $\mathsf{C}$ has no redundant*

*variable.*

**Proof:** Let $\text{var}(\mathtt{C}) = \mathbf{x}$. Over any field, $N_{ess}(\mathtt{C}) \geq \dim \left\langle \frac{\partial \mathtt{C}}{\partial x} : x \in \mathbf{x} \right\rangle$. So it is sufficient to show that $\dim \left\langle \frac{\partial \mathtt{C}}{\partial x} : x \in \mathbf{x} \right\rangle = |\mathbf{x}|$. We will prove this by induction on the product depth $\Delta$ of $\mathtt{C}$. In the base case, $\Delta = 0$, and $\mathtt{C}$ computes a polynomial $x + \gamma$, for $\gamma \in \mathbb{F}$; so, $\dim \left\langle \frac{\partial \mathtt{C}}{\partial x} : x \in \mathbf{x} \right\rangle = |\mathbf{x}| = 1$. Suppose that the induction hypothesis holds for canonical ROFs of product depth $\Delta - 1$ or less.

Let $\mathtt{C} = T_1 + \ldots + T_s + \gamma$ be a canonical ROF of product depth $\Delta$, where each $T_i$ is a $\times$-rooted ROF having at least two non-constant, variable disjoint factors. Consider an $\mathbb{F}$-linear dependence $\sum_{x \in \mathbf{x}} \alpha_x \frac{\partial \mathtt{C}}{\partial x} = 0$, where $\alpha_x \in \mathbb{F}$. Then, $\sum_{x \in \text{var}(T_i)} \alpha_x \frac{\partial T_i}{\partial x} \in \mathbb{F}$ for every $i \in [s]$. This is because $T_i$ and $T_j$ are variable disjoint for $i \neq j$. But $\sum_{x \in \text{var}(T_i)} \alpha_x \frac{\partial T_i}{\partial x} \in \mathbb{F}$ implies $\sum_{x \in \text{var}(T_i)} \alpha_x \frac{\partial T_i}{\partial x} = 0$, as $T_i$ is a product of at least two non-constant factors and a common root of these variable disjoint factors is also a root of $\sum_{x \in \text{var}(T_i)} \alpha_x \frac{\partial T_i}{\partial x}$. Now suppose $\alpha_x \neq 0$ for some $x \in \text{var}(T_i)$ and $i \in [s]$. Let $T_i = Q_1 \cdots Q_m$, where $Q_1, \ldots, Q_m$ are variable disjoint $+$-rooted canonical ROFs of product depth at most $\Delta - 1$. Suppose that the $x$ mentioned above is in $\text{var}(Q_l)$. By the induction hypothesis, $\sum_{y \in \text{var}(Q_l)} \alpha_y \frac{\partial Q_l}{\partial y} \neq 0$ unless every $\alpha_y = 0$. So the dependence $\sum_{x \in \text{var}(T_i)} \alpha_x \frac{\partial T_i}{\partial x} = 0$ implies $Q_l$ divides $\sum_{y \in \text{var}(Q_l)} \alpha_y \frac{\partial Q_l}{\partial y} \neq 0$, which is not possible as the latter has a smaller degree. Therefore, $\alpha_x = 0$ for every $x \in \mathbf{x}$, and so, $\dim \left\langle \frac{\partial \mathtt{C}}{\partial x} : x \in \mathbf{x} \right\rangle = |\mathbf{x}|$. □

If $\mathtt{C}$ is not canonical, then all the variables of $\mathtt{C}$ need not be essential. For e.g., $x_1 + \cdots + x_n$ is an ROF with only one essential variable. The above observations imply the following:

**Observation 5.3** *Let $\mathtt{C}$ be an ROF, $\mathtt{C}'$ a canonical ROF, and $\mathtt{C}' \in \text{orb}(\mathtt{C})$. Then, $N_{ess}(\mathtt{C}) = |\text{var}(\mathtt{C}')|$.*

We now state an important property of a canonical ROF which will be used in the equivalence test.

**Claim 5.1 (Canonical ROF modulo an affine form)** *Let $n \in \mathbb{N}$, $\text{char}(\mathbb{F}) \neq 2$, $|\mathbb{F}| > n$, $\mathtt{C}$ be a $+$-rooted $n$-variate canonical ROF, and $\ell$ an affine form which is not a constant multiple of some variable. Then, $N_{ess}(\mathtt{C}_\ell) \geq n - 2$.*

**Proof:** Let $\mathbf{x} = \{x_1, \ldots, x_n\}$ be $\text{var}(\mathtt{C})$, where $\mathtt{C} = T_1 + \cdots + T_s + \gamma$ is a canonical ROF. Let $\ell = \sum_{x \in \mathbf{x}} \alpha_x x + \alpha$, where either $|\text{var}(\ell)| \geq 2$, $\alpha \in \mathbb{F}$ and for every $x \in \mathbf{x}$, $\alpha_x \in \mathbb{F}$ or $|\text{var}(\ell)| = 1$ and $\alpha \in \mathbb{F}^\times$. Let $\mathbf{x}' = \mathbf{x} \setminus \{y_1\}$, $\ell_1 = \sum_{x \in \mathbf{x}'} -\alpha'_x x - \alpha'$, where for every $x \in \mathbf{x}', \alpha'_x = \alpha_x \alpha_{y_1}^{-1}$ and $\alpha' = \alpha \alpha_{y_1}^{-1}$. Notice that $\ell_1 \neq 0$. Then, we know that $\mathtt{C}_\ell = \mathtt{C}(y_1 = \ell_1, \mathbf{x}')$. As $\mathtt{C}$ is

canonical, there exists at most one $l \in [s]$, such that $T_l$ is a variable. If such an $l$ exists and $\text{var}(T_l) \cap \text{var}(\ell) \neq \emptyset$ then we assume without loss of generality that $l = 1$. We also assume that $y_1 \in \text{var}(T_1)$. Then, note that $\mathtt{C}_\ell = T_1' + T_2 + \cdots + T_s + \gamma$, where $T_1' := T_1(y_1 = \ell_1, \text{var}(T_1) \setminus \{y_1\})$. For $l \in [2, s]$[1], let $\mathbf{x}_l' = \text{var}(T_l)$ and $\mathbf{x}_1' = \text{var}(T_1) \setminus \{y_1\}$. We first prove the following two useful observations.

**Observation 5.4** $U := \left\langle \frac{\partial \mathtt{C}}{\partial x} : x \in \text{var}(T_l), l \in [s], |\text{var}(T_l)| \geq 2 \right\rangle$ *does not contain a non-zero constant.*

**Proof:** Suppose there exists an $\alpha \in U \cap \mathbb{F} \setminus \{0\}$. Let $\mathtt{C}' = \sum_{l \in [s], |\text{var}(T_l)| \geq 2} T_l + \alpha y$, where $y$ is a fresh variable. Then, $\mathtt{C}'$ is in the orbit of the canonical ROF $\sum_{l \in [s], |\text{var}(T_l)| \geq 2} T_l + y$ and it follows from Fact 2.8 and Observation 5.2 that $N_{ess}(\mathtt{C}') = |\text{var}(\mathtt{C}')|$. Thus, $W := \left\{ \frac{\partial \mathtt{C}'}{\partial x} : x \in \text{var}(\mathtt{C}') \right\}$ is $\mathbb{F}$-linearly independent. Note that $U = \langle W \rangle$ but $\dim U < |W|$; a contradiction. So $U \cap \mathbb{F} \setminus \{0\} = \emptyset$. □

**Observation 5.5** *If* $|\text{var}(T_1)| \leq 2$, *then* $N_{ess}(\mathtt{C}_\ell) \geq n - 2$.

**Proof:** There are two cases, $T_1 = y_1$ and $T_1 = y_1 y$ for some $y \in \mathbf{x}'$. For both cases, it follows from Observation 5.2 that $\left\{ \frac{\partial T_l}{\partial x} : l \in [2, s],\ x \in \text{var}(T_l) \right\} \uplus \left\{ \frac{\partial T_1}{\partial y_1} \right\}$ is $\mathbb{F}$-linearly independent. Now, for any $l \in [2, s]$ and $x \in \text{var}(T_l)$, $\frac{\partial \mathtt{C}_\ell}{\partial x} = \frac{\partial T_l}{\partial x} - \alpha_x' \frac{\partial T_1}{\partial y_1}$. Thus, $\left\{ \frac{\partial \mathtt{C}_\ell}{\partial x} : x \in \text{var}(T_l), l \in [2, s] \right\}$ is $\mathbb{F}$-linearly independent. Hence, from Fact 2.5, when $T_1 = y_1$, $N_{ess}(\mathtt{C}_\ell) \geq n - 1$, and when $T_1 = y_1 y$, $N_{ess}(\mathtt{C}_\ell) \geq n - 2$. □ As $\mathtt{C}$ is multilinear, for every $x \in \mathbf{x}$, the individual degree of $x$ in $\mathtt{C}$ is at most 2. Since $\text{char}(\mathbb{F}) \neq 2$, for every $l \in [s], x \in \mathbf{x}_l'$, $\frac{\partial \mathtt{C}_\ell}{\partial x} \neq 0$. For $l \in [s], x \in \mathbf{x}_l'$, let $\beta_{l,x} \in \mathbb{F}$, such that $\sum_{l \in [s]} \sum_{x \in \mathbf{x}_l'} \beta_{l,x} \frac{\partial \mathtt{C}_\ell}{\partial x} = 0$, which implies

$$\sum_{l \in [2,s]} \sum_{x \in \mathbf{x}_l'} \beta_{l,x} \left( \frac{\partial T_l}{\partial x} + \frac{\partial T_1'}{\partial x} \right) + \sum_{x \in \mathbf{x}_1'} \beta_{1,x} \frac{\partial T_1'}{\partial x} = 0. \tag{5.1}$$

Let $I = \{l \in [s] : |\text{var}(T_l)| = 1\}$ and $J = [2, s] \setminus I$. As $\mathtt{C}$ is canonical, $|I| \leq 1$. If $l \in I$, we call $T_l$ as $z$. Now, we prove the claim by induction on the product-depth $\Delta$ of $\mathtt{C}$.

**Base case:** $\Delta = 1$. Then, $\mathtt{C}_\ell = \ell_1 T_1'' + T_2 + \cdots + T_s + \gamma$, where for every $l \in [2, s]$, $T_l$ is a multilinear monomial and $T_1'' = \prod_{x \in \mathbf{x}_1'} x$. Because of Observation 5.5, we can assume that $z \notin \text{var}(\ell_1)$. Thus Equation (5.1) becomes

$$\sum_{l \in J} \sum_{x \in \mathbf{x}_l'} \beta_{l,x} \left( -\alpha_x' T_1'' + \frac{T_l}{x} \right) + \sum_{l \in I} \beta_{l,z} + \sum_{x \in \mathbf{x}_1'} \beta_{1,x} \left( -\alpha_x' T_1'' + \ell_1 \frac{T_1''}{x} \right) = 0. \tag{5.2}$$

[1] For $m < n \in \mathbb{N}$, $[m, n] := \{m, \ldots, n\}$

If $|\mathbf{x}'_1| \leq 1$, we immediately have from Observation 5.5 that $N_{ess}(C_\ell) \geq n-2$. So suppose that $|\mathbf{x}'_1| \geq 2$. Then for every $l \in J, x \in \mathbf{x}'_l$, the coefficient of $\frac{T_l}{x}$ in the above equation is $\beta_{l,x}$, which implies $\beta_{l,x} = 0$. Also, as $T''_1$ and $\frac{T''_1}{x}$ are non-constant monomials for every $x \in \mathbf{x}'_1$ and as $|I| \leq 1, \beta_{l,z} = 0$. If $\text{var}(\ell_1) \cap \text{var}(T'_1) = \emptyset$, then Equation (5.2) becomes $\sum_{x \in \mathbf{x}'_1} \beta_{1,x} \cdot \ell_1 \frac{T''_1}{x} = 0$ which implies $\sum_{x \in \mathbf{x}'_1} \beta_{1,x} \cdot \frac{T''_1}{x} = 0$. Then from Observation 5.2, $\beta_{1,x} = 0$ for all $x \in \mathbf{x}'_1$ and $N_{ess}(\mathtt{C}_\ell) = n-1$. Otherwise pick any $y \in \text{var}(\ell_1) \cap \text{var}(T''_1)$ arbitrarily. Observe that the polynomial multiplied by $y^2$ in Equation (5.2) is $-\alpha'_y \sum_{x \in \mathbf{x}'_1 \setminus \{y\}} \beta_{1,x} \frac{T''_1}{y \cdot x}$. As $\frac{T''_1}{y}$ is a canonical ROF, it follows from Observation 5.2 that $\beta_{1,x} = 0$ for all $x \in \mathbf{x} \setminus y$. Then, from Equation (5.2) we have $\beta_{1,y} \left( -\alpha'_y T''_1 + \ell_1 \frac{T''_1}{y} \right) = 0$. As the coefficient of $T''$ in this polynomial is $-2\alpha'_y \beta_{1,y}$, and $\text{char}(\mathbb{F}) \neq 2$, $\beta_{1,y} = 0$. Hence, again $N_{ess}(\mathtt{C}_\ell) = n-1$. This proves the base case.

**Induction step:** Suppose $\Delta > 1$ and the claim holds for all canonical ROFs of product-depth at most $\Delta - 1$. Let $T_1 = Q_1 \cdots Q_m$, where for every $i \in [m], Q_i$ is either a variable or a +-rooted ROF. As in the base case, if $|\mathbf{x}'_1| \leq 1$ or $z \in \text{var}(\ell_1)$, then there is nothing to prove. So, suppose that $|\mathbf{x}'_1| \geq 2$ and $z \notin \text{var}(\ell_1)$. We assume without loss of generality that $y_1 \in \text{var}(Q_1)$. It follows from the definition of $\mathtt{C}_\ell$ that $T'_1 = Q'_1 Q_2 \cdots Q_m$, where $Q'_1 = Q_1(y_1 = \ell_1, \text{var}(Q_1) \setminus \{y_1\})$. For $i \in [2, m]$, let $\widetilde{Q}_i = Q'_1 \prod_{j \in [2,m] \setminus \{i\}} Q_j$ and $\widetilde{Q}_1 = Q_2 \cdots Q_m$. Let $\mathbf{x}'_{1,1} = \text{var}(Q_1) \setminus \{y_1\}$ and for $i \in [2, m], \mathbf{x}'_{1,i} = \text{var}(Q_i)$. For $i \in [m], x \in \mathbf{x}'_{1,i}$, rename the coefficient of $\frac{\partial T'_1}{\partial x}$ in Equation (5.1) as $c_{i,x}$. Then, Equation (5.1) becomes

$$\sum_{l \in J, x \in \mathbf{x}'_l} \beta_{l,x} \frac{\partial T_l}{\partial x} + \sum_{l \in I} \beta_{l,z} + \widetilde{Q}_1 \left( \sum_{l \in J, x \in \mathbf{x}'_l} \beta_{l,x} \frac{\partial Q'_1}{\partial x} + \sum_{i \in [m], x \in \mathbf{x}'_{1,i}} c_{i,x} \frac{\partial Q'_1}{\partial x} \right) + \sum_{i \in [2,m]} \widetilde{Q}_i \left( \sum_{x \in \mathbf{x}'_{1,i}} c_{i,x} \frac{\partial Q_i}{\partial x} \right) = 0. \tag{5.3}$$

**Observation 5.6** *If $T_1 \neq y_1 Q_2$, then for every $l \in J, x \in \mathbf{x}'_l, \beta_{l,x} = 0$.*

**Proof:** If $m \geq 3$ then we substitute roots of $Q_2$ and $Q_3$ in Equation (5.3). As $|\mathbb{F}| > n$ and $Q_2$ and $Q_3$ are variable disjoint multilinear polynomials, roots of $Q_2$ and $Q_3$ exist over $\mathbb{F}$. Then, Observation 5.2 implies for $l \in J, x \in \mathbf{x}'_l, \beta_{l,x} = 0$ and as $|I| \leq 1, \beta_{l,z} = 0$.

Now, suppose $m = 2$. Let the polynomial multiplied with $\widetilde{Q}_2 = Q'_1$ in Equation (5.3) be $q_1$. We plug in a root $\mathbf{a}$ of $Q_2$ in Equation (5.3). Let $h' = \sum_{l \in J, x \in \mathbf{x}'_l} \beta_{l,x} \frac{\partial T_l}{\partial x}$ and $h = h' + \sum_{l \in I} \beta_{l,z}$.

Then $h = h(\mathbf{x}'_{1,2} = \mathbf{a}, \mathbf{x}' \setminus \mathbf{x}'_{1,2})$, and Equation (5.3) implies that $h = -q_1(\mathbf{a})Q'_1(\mathbf{x}'_{1,2} = \mathbf{a}, \mathbf{x}' \setminus \mathbf{x}'_{1,2})$. Note that $q_1(\mathbf{a}) \in \mathbb{F}$. If either $q_1(\mathbf{a}) = 0$ or $\mathrm{var}(\ell_1) \cap \left(\uplus_{l \in J} \mathbf{x}'_l\right) = \emptyset$, then Observation 5.4 implies that $h' = 0$. Otherwise, $\deg(Q'_1(\mathbf{x}'_{1,2} = \mathbf{a}, \mathbf{x}' \setminus \mathbf{x}'_{1,2})) = \deg(Q'_1)$. Also, in this case, $\deg(Q'_1) = \deg(Q_1)$. As $Q_1 \neq y_1$, $\deg(Q_1) \geq 2$. Hence, $\deg(Q'_1(\mathbf{x}'_{1,2} = \mathbf{a}, \mathbf{x}' \setminus \mathbf{x}'_{1,2})) \geq 2$. Then there exists a monomial $p$ in $Q'_1(\mathbf{x}'_{1,2} = \mathbf{a}, \mathbf{x}' \setminus \mathbf{x}'_{1,2})$, such that $\deg(p) \geq 2$ and $\mathrm{var}(p) \cap \mathrm{var}(Q_1) \neq \emptyset$. Clearly, $p$ is not in $h$, and as $h = -q_1(\mathbf{a})Q'_1(\mathbf{x}'_{1,2} = \mathbf{a}, \mathbf{x}' \setminus \mathbf{x}'_{1,2})$, we get $h = 0$. This along with Observation 5.4 implies $h' = 0$. Thus from Observation 5.2, $\beta_{l,x} = 0$ for every $l \in J, x \in \mathbf{x}'_l$. □

It follows from the above observation that when $T_1 \neq y_1 Q_2$, Equation (5.3) becomes

$$\sum_{l \in I} \beta_{l,z} + \widetilde{Q}_1 \left( \sum_{i \in [m], x \in \mathbf{x}'_{1,i}} c_{i,x} \frac{\partial Q'_1}{\partial x} \right) + \sum_{i \in [2,m]} \widetilde{Q}_i \left( \sum_{x \in \mathbf{x}'_{1,i}} c_{i,x} \frac{\partial Q_i}{\partial x} \right) = 0. \tag{5.4}$$

Now, we consider all the possible cases of $T'_1$. Recall $|\mathbf{x}'_1| \geq 2$, which implies that if $m = 2$ and $Q'_1$ is a linear polynomial then $\deg(Q_2) \geq 2$.

**Case 1**: $m = 2, \deg(Q_1) \geq 2$, and $Q_2 = y$ for some $y \in \mathbf{x}'$. Then, Equation (5.4) looks like

$$\sum_{l \in I} \beta_{l,z} + y \left( \sum_{x \in \mathbf{x}'_{1,1}} c_{1,x} \frac{\partial Q'_1}{\partial x} + c_{2,y} \frac{\partial Q'_1}{\partial y} \right) + Q'_1 c_{2,y} = 0. \tag{5.5}$$

If $y \notin \mathrm{var}(\ell_1)$, we put $y = 0$ in Equation (5.5). As $Q'_1(y = 0, \mathbf{x}' \setminus \{y\}) = Q'_1$, $c_{2,y} = 0$. As $|I| \leq 1$, $\beta_{l,z} = 0$. Thus we are left with $\sum_{x \in \mathbf{x}'_{1,1}} c_{1,x} \frac{\partial Q'_1}{\partial x} = 0$. Let $\mathbf{a} \in \mathbb{F}^{|\mathbf{x}' \setminus \mathbf{x}'_{1,1}|}$ be a point such that $\ell_1(\mathbf{x}' \setminus \mathbf{x}'_{1,1} = \mathbf{a}, \mathbf{x}'_{1,1}) \neq 0$; such a point exists. Notice that $N_{ess}(Q'_1(\mathbf{x}' \setminus \mathbf{x}'_{1,1} = \mathbf{a}, \mathbf{x}'_{1,1})) \leq N_{ess}(Q'_1)$. Because $Q_1$ is a product-depth $\Delta - 1$ ROF and $\ell_1(\mathbf{x}' \setminus \mathbf{x}'_{1,1} = \mathbf{a}, \mathbf{x}'_{1,1}) \neq 0$, it follows from the induction hypothesis that $N_{ess}(Q'_1(\mathbf{x}' \setminus \mathbf{x}'_{1,1} = \mathbf{a}, \mathbf{x}'_{1,1})) \geq |\mathrm{var}(Q_1)| - 2$. This means that at least $|\mathrm{var}(Q_1)| - 2$ many elements in $\left\{ \frac{\partial Q'_1}{\partial x} : x \in \mathbf{x}'_{1,1} \right\}$ are $\mathbb{F}$-linearly independent. Hence, at least $n - 2$ many elements in $\left\{ \frac{\partial \mathbb{C}_\ell}{\partial x} : x \in \mathbf{x}' \right\}$ are $\mathbb{F}$-linearly independent, and $N_{ess}(\mathbb{C}_\ell) \geq n - 2$.

If $y \in \mathrm{var}(\ell_1)$, $Q'_1 = yQ + q$ for some $Q \in \mathbb{F}[\mathbf{x}'_{1,1}]$ and $q \in \mathbb{F}[\mathbf{x}' \setminus \{y\}]$. If $q$ is not a constant, just as before, we set $y = 0$ in Equation (5.5). This gives us $c_{2,y} = \beta_{l,z} = 0$. If $q \in \mathbb{F}$, note that $\mathbf{x}'_{1,1} \subseteq \mathrm{var}(Q)$, and hence $y^2$ divides $y \cdot \frac{\partial Q'_1}{\partial x}$ for all $x \in \mathbf{x}'_{1,1}$. This means that $\sum_{l \in I} \beta_{l,z} + c_{2,y} \left( y \frac{\partial Q'_1}{\partial y} + Q'_1 \right) = 0$. Now $y \frac{\partial Q'_1}{\partial y} = yQ$. Thus, $y \frac{\partial Q'_1}{\partial y} + Q'_1 = 2c_{2,y} yQ + q$. As

$\mathrm{char}(\mathbb{F}) \geq 2$, $c_{2,y} = 0$, and thus $\beta_{l,z} = 0$. Then, using the induction hypothesis as before, we get $N_{ess}(\mathtt{C}_\ell) \geq n - 2$.

**Case 2**: $m = 2, Q_1 = y_1$, and $\deg(Q_2) \geq 2$. If $y_2 \in \mathrm{var}(\ell_1) \cap \mathbf{x}'_{1,2}$ then we redo this entire analysis by considering $\mathtt{C}_\ell = \mathtt{C}(y_2 = \ell_2, \mathbf{x} \setminus \{y_2\})$, where $\ell_2 := -\alpha_{y_2}^{-1}(\ell - \alpha_{y_2} y_2)$. Definition 2.30 and Observation 2.11 ensure that $N_{ess}(\mathtt{C}_\ell)$ is not affected by making this change to the definition of $\mathtt{C}_\ell$. Then, this case is same as Case 1 and we get the desired result. Otherwise, Equation (5.3) looks like

$$\sum_{l \in J, x \in \mathbf{x}'_l} \beta_{l,x} \frac{\partial T_l}{\partial x} + \sum_{l \in I} \beta_{l,z} + Q_2 \left( \sum_{l \in J, x \in \mathbf{x}'_l} -\beta_{l,x} \alpha'_x \right) + \ell_1 \left( \sum_{x \in \mathbf{x}'_{1,2}} c_{2,x} \frac{\partial Q_2}{\partial x} \right) = 0.$$

If $Q_2$ has a dangling variable connected to its top $+$ gate, let it be $y_2$. We shall consider the above equation without $c_{2,y} \frac{\partial Q_2}{\partial y_2}$. Observe that any monomial of the highest degree in $Q_2$ is not present in any other summand in the above equation. Hence $\sum_{l \in J, x \in \mathbf{x}'_l} -\beta_{l,x} \alpha'_x = 0$. Also, for every $x \in \mathbf{x}'_{1,2} \setminus \{y_2\}$, $\frac{\partial Q_2}{\partial x} \in \mathbb{F}[\mathbf{x}'_{1,2}]$. Hence, $\sum_{x \in \mathbf{x}'_{1,2} \setminus \{y_2\}} c_{2,x} \frac{\partial Q_2}{\partial x} = c$ for a $c \in \mathbb{F}$. It follows from Observation 5.4 that $c = 0$, and hence from Observation 5.2 that $c_{2,x} = 0$ for all $x \in \mathbf{x}'_{1,2} \setminus \{y_2\}$. Observation 5.4 and the fact that $|I| \leq 1$ imply that $\beta_{l,z} = 0$. Then, from Observation 5.2 $\beta_{l,x} = 0$ for all $l \in J$ and $x \in \mathbf{x}'_l$. Hence, $\left\{ \frac{\partial \mathtt{C}_\ell}{\partial x} : x \in \mathbf{x}' \setminus \{y_2\} \right\}$ is $\mathbb{F}$-linearly independent and $N_{ess}(\mathtt{C}_\ell) \geq n - 2$.

**Case 3**: $m = 2$, $\deg(Q_1) \geq 2$, and $\deg(Q_2) \geq 2$. In this case, Equation (5.4) becomes

$$\sum_{l \in I} \beta_{l,z} + Q_2 \left( \sum_{i \in [2], x \in \mathbf{x}'_{1,i}} c_{i,x} \frac{\partial Q'_1}{\partial x} \right) + Q'_1 \left( \sum_{x \in \mathbf{x}'_{1,2}} c_{2,x} \frac{\partial Q_2}{\partial x} \right) = 0.$$

Let the polynomials multiplied by $Q'_1$ and $Q_2$ in the above equation be $q_1$ and $q_2$, respectively. Let $v$ be the parent of $y_1$ in $\mathtt{C}$ and $\mathrm{path}(v)$ be the path from the root of $\mathtt{C}$ to $v$. If $v$ is the top-most + gate then substitute a root $\mathbf{a}$ of $Q_2$ in the above equation; $q_1(\mathbf{a}) \in \mathbb{F}$. As $\deg(Q'_1) \geq 2$ and $|I| \leq 1$, we get $\beta_{l,z} = 0$. Otherwise, there exists a $\times$ gate $v'$ on $\mathrm{path}(v)$, such that $Q_{v',1}$ and $Q_{v',2}$ are children of $v'$, where $Q_{v',1}$ lies on $\mathrm{path}(v)$ and $Q_{v',2}$ does not. Clearly, $\ell_1$ is present in $Q_{v',1}$, and $Q_{v',2}$ is a +-rooted sub-ROF or a variable of $Q_1$. We first substitute a root $\mathbf{a}$ of $Q_{v',2}$ in the above equation and then plug in a root of $Q'_1(\mathbf{x}'_{1,2} = \mathbf{a}, \mathbf{x}' \setminus \mathbf{x}'_{1,2})$. In this process, note that $\mathbf{x}'_{1,2} \cup \mathrm{var}(\ell_1) \setminus \mathbf{x}'_{1,1}$ is untouched. As $|I| \leq 1$, $\beta_{l,z} = 0$. Further, since $Q_2$ is irreducible (Fact 2.1), we get that $Q_2$ either divides $Q'_1$ or $q_1$. As $\deg(Q_2) \geq 2$, $Q_2$ contains a

monomial not present in $Q_1'$ and $Q_2$ does not divide $Q_1'$. As $\deg(Q_2) > \deg(q_1)$, $Q_2$ dividing $q_1$ implies that $q_1 = 0$. Thus, using Observation 5.2 we get that $c_{2,x} = 0$ for all $x \in \mathbf{x}'_{2,x}$. Then, using the induction hypothesis like in Case 1, we get $N_{ess}(\mathsf{C}_\ell) \geq n - 2$.

**Case 4**: $m \geq 3$. By putting the roots of $Q_2$ and $Q_3$ in Equation (5.4), we get $\beta_{l,z} = 0$. For $i \in [2, m]$, let $q_i$ be the polynomial multiplied with $\widetilde{Q}_i$ in Equation (5.4). Then for every $i \in [2, m]$, $Q_i$ divides $\widetilde{Q}_i q_i$. As $Q_i$ is irreducible (Fact 2.1), $Q_i$ must divide $q_i$ or $\widetilde{Q}_i$. Suppose there exists an $i$ such that $Q_i$ divides $\widetilde{Q}_i$. This happens if and only if $Q_i = x$ and $Q_1' = \ell_1 = -\alpha_x' x$, where $x \in \mathbf{x}'$. Note that such an $i$ is unique, say $i = 2$. Now, for every $j \in [3, m]$, $Q_j$ must divide $q_j$. As $\deg(Q_j) > \deg(q_j)$, $q_j = 0$. Then, Equation (5.4) becomes $c_{2,x}(-\alpha_x \widetilde{Q}_1 + \widetilde{Q}_2) = -2c_{2,x}\alpha_x \cdot \prod_{j\in[3,m]} Q_j = 0$. As $\mathrm{char}(\mathbb{F}) \neq 2$ and $\alpha_x' \neq 0$, $c_{i,x} = 0$. If such an $i$ does not exist, then $q_j = 0$ for all $j \in [2, m]$. In either case, using Observation 5.2 we get, $c_{j,x} = 0$ for every $j \in [2, m], x \in \mathbf{x}'_{1,j}$. Then, Equation (5.4) becomes $\sum_{x\in\mathbf{x}'_{1,1}} c_{1,x} \frac{\partial Q_1'}{\partial x} = 0$. Using the induction hypothesis like in Case 1, we get $N_{ess}(\mathsf{C}_\ell) \geq n - 2$. □

### 5.3.2 Algorithmic preliminaries

**Fact 5.1 (Black-box for partials)** *Let $d \in \mathbb{N}$, $\mathrm{char}(\mathbb{F}) = 0$ or $\mathrm{char}(\mathbb{F}) > d$, and $g \in \mathbb{F}[\mathbf{x}]$ be a degree $d$ polynomial given as a black-box. Then, for $x \in \mathbf{x}$, a black-box for $\frac{\partial g}{\partial x}$ can be computed in* $\mathrm{poly}(|\mathbf{x}|, d)$ *time.*

The above fact is well-known; a proof of it can be found in Section 2.2 of [KNST17].

**Fact 5.2 (Black-box polynomial factorization [KT90])** *Let $d \in \mathbb{N}$, $\mathrm{char}(\mathbb{F}) = 0$ or $\mathrm{char}(\mathbb{F}) > d$, and $|\mathbb{F}| \geq d^6$. There is a randomized algorithm, with oracle access to univariate polynomial factorization over $\mathbb{F}$, that takes input black-box access to a polynomial $g \in \mathbb{F}[\mathbf{x}]$ of degree $d$ and outputs black-boxes for the irreducible factors of $g$ in* $\mathrm{poly}(|\mathbf{x}|, d)$ *time.*

*Remark.* Since our model of computation allows univariate polynomial factorization, we will assume that black-box polynomial factorization can be done in randomized polynomial-time. This assumption is justified particularly for finite fields and $\mathbb{Q}$ [Ber70, LLL82].

**Quadratic Form Equivalence.** Known algorithms for quadratic form equivalence (QFE) over $\mathbb{C}, \mathbb{R}, \mathbb{Q}$, and finite fields are based on well-known classification of quadratic forms. Refer to [Ser73, Lam04, Ara11] for a comprehensive discussion on this. We record the complexity of QFE over these fields in the fact below. Over $\mathbb{R}$ and $\mathbb{C}$, the model of computation

is an arithmetic circuit with oracle access to a square root finding algorithm; every operation in the circuit takes a unit time. Whereas, over $\mathbb{Q}$ and finite fields, the model of computation is a Turing machine, i.e., the running time is measured as bit operations.

**Fact 5.3 (Complexity of QFE)** *Let $n$ be the number of variables in each of the two input quadratic forms.*

1. (Over $\mathbb{C}$ and $\mathbb{R}$). *There is a deterministic* $\mathrm{poly}(n)$ *time QFE algorithm.*
2. (Over finite fields). *Let* $\mathrm{char}(\mathbb{F}) \neq 2$. *There is a randomized* $\mathrm{poly}(n, \log|\mathbb{F}|)$ *time QFE algorithm.*
3. (Over $\mathbb{Q}$) [Wal13]. *There is a deterministic* $\mathrm{poly}(n, b)$ *time QFE algorithm with oracle access to integer factoring, where $b$ is the bit length of the coefficients of the input quadratic forms.*

Let $g_1, \ldots, g_m \in \mathbb{F}[\mathbf{x}]$ be pairwise variable disjoint. Recall that we use $N_{ess}(g)$ to denote the number of essential variables in a polynomial $g$ (see Definition 2.28). It can be shown that $N_{ess}(g_1 \cdots g_m) = N_{ess}(g_1) + \ldots + N_{ess}(g_m)$ over any field. The following claim proves the converse and, more importantly, provides an algorithm to find a transformation that makes $g_1, \ldots, g_m$ pairwise variable disjoint.

**Claim 5.2 (Making polynomials variable disjoint)** *Let $d \in \mathbb{N}$, $\mathrm{char}(\mathbb{F}) = 0$ or $> d$, and $|\mathbb{F}| \geq 2|\mathbf{x}|d$. There is a randomized* $\mathrm{poly}(|\mathbf{x}|, d)$ *time algorithm that takes input black-box access to $g_1, \ldots, g_m \in \mathbb{F}[\mathbf{x}]$, where $g_1 \cdots g_m \in \mathbb{F}[\mathbf{x}]_{\leq d}$ and $N_{ess}(g_1 \cdots g_m) = \sum_{i \in [m]} N_{ess}(g_i)$, and outputs an $A \in \mathrm{GL}(|\mathbf{x}|, \mathbb{F})$ such that $g_1(A\mathbf{x}), \ldots, g_m(A\mathbf{x})$ are pairwise variable disjoint and individually free of redundant variables.*

**Proof:**

**Algorithm 1** Make-Polys-Var-Disjoint($g_1, \ldots, g_m$)

**Input:** Black-box access to $g_1, \ldots g_m \in \mathbb{F}[\mathbf{x}]$ such that $N_{ess}(g_1 \cdots g_m) = N_{ess}(g_1) + \cdots + N_{ess}(g_m)$.
**Output:** An $A \in \mathrm{GL}(|\mathbf{x}|, \mathbb{F})$ such that $g_1(A\mathbf{x}), \ldots, g_m(A\mathbf{x})$ are pairwise variable disjoint and individually free of redundant variables.

1: $A \leftarrow I_{|\mathbf{x}| \times |\mathbf{x}|}$, $\mathbf{y} \leftarrow \emptyset$.
2: **for** $i = 1, \ldots, m$ **do**
3: $\quad A_i \leftarrow$ Remove-Redundant-Vars$(g_i(A\mathbf{x}), \mathbf{y})$ (see Claim 2.1); $\mathbf{y}_i \leftarrow \mathrm{var}(g_i(AA_i\mathbf{x}))$.
4: $\quad A \leftarrow AA_i, \mathbf{y} \leftarrow \mathbf{y} \cup \mathbf{y}_i$.
5: **end for**
6: Return $A$.

The correctness of the algorithm follows from the observations below.

**Observation 5.7** *For every $i \in [m]$, $N_{ess}(g_1 \cdots g_i) = N_{ess}(g_1) + \cdots + N_{ess}(g_i)$.*

**Proof:** Follows from the fact that over any $\mathbb{F}$, $N_{ess}(h_1 h_2) \leq N_{ess}(h_1) + N_{ess}(h_2)$, for $h_1, h_2 \in \mathbb{F}[\mathbf{x}]$. □

**Observation 5.8** *Suppose $\mathbf{x} = \mathbf{y} \uplus \mathbf{z}$ and $h_1(\mathbf{y}), h_2(\mathbf{z}, \mathbf{y}) \in \mathbb{F}[\mathbf{x}]$ such that $N_{ess}(h_1) = |\mathbf{y}|$ and $N_{ess}(h_1 h_2) = N_{ess}(h_1) + N_{ess}(h_2)$. Then, $\mathbf{z}$ contains a set of essential variables of $h_2$.*

**Proof:** Observe that $\dim \left\langle \frac{\partial h_1 h_2}{\partial z} : z \in \mathbf{z} \right\rangle = \dim \left\langle \frac{\partial h_2}{\partial z} : z \in \mathbf{z} \right\rangle$; let this dimension be $l$. Then, by Fact 2.5, $N_{ess}(h_1 h_2) \leq l + |\mathbf{y}|$, which implies $N_{ess}(h_2) \leq l$ (as $N_{ess}(h_1) = |\mathbf{y}|$ and $N_{ess}(h_1 h_2) = N_{ess}(h_1) + N_{ess}(h_2)$). On the other hand, $\dim \left\langle \frac{\partial h_2}{\partial z} : z \in \mathbf{z} \right\rangle = l$ implies $N_{ess}(h_2) \geq l$. Hence, $N_{ess}(h_2) = l$, and so by Fact 2.5, $\mathbf{z}$ contains a set of essential variables of $h_2$. □

This finishes the proof of Claim 5.2. □

The next claim generalizes the above claim and is used crucially in the equivalence test presented in Section 5.5.

**Claim 5.3 (Making factors variable disjoint)** *Let $d \in \mathbb{N}$, $\mathrm{char}(\mathbb{F}) = 0$ or $> d$, and $|\mathbb{F}| \geq \max\{2|\mathbf{x}|d, d^6\}$. There is a randomized $\mathrm{poly}(|\mathbf{x}|, d)$ time algorithm that takes input black-box access to an $f = g(B\mathbf{x} + \mathbf{d})$, where $B \in \mathrm{GL}(|\mathbf{x}|, \mathbb{F})$, $\mathbf{d} \in \mathbb{F}^{|\mathbf{x}|}$, and $g \in \mathbb{F}[\mathbf{x}]_{\leq d}$ such that $g = g_1 \cdots g_m$ for pairwise variable disjoint $g_1, \ldots, g_m \in \mathbb{F}[\mathbf{x}]_{\leq d}$, and does the following: (Here, $B, \mathbf{d}, g$, and $g_1, \ldots, g_m$ are unknown to the algorithm.)*

1. *It computes an $A \in \mathrm{GL}(|\mathbf{x}|, \mathbb{F})$ such that $g_1(BA\mathbf{x} + \mathbf{d}), \ldots, g_m(BA\mathbf{x} + \mathbf{d})$ are pairwise variable disjoint and individually free of redundant variables. ($g_1, \ldots, g_m$ need not be irreducible.)*
2. *It computes a set $V$ of pairwise disjoint subsets of $\mathbf{x}$ such that for every $i \in [m]$, there exist $h_{i,1}, \ldots, h_{i,m_i} \in \mathbb{F}[\mathbf{x}]$ satisfying $\prod_{l \in [m_i]} h_{i,l} = g_i(B\mathbf{x} + \mathbf{d})$, and $V = \{\mathrm{var}(h_{i,l}(A\mathbf{x})) : i \in [m], l \in [m_i]\}$.*

**Proof:**
The correctness of the algorithm follows from the following observation. Note that the number of essential variables of a polynomial can be computed efficiently using Claim 2.1. As we are merging factors in Step 4, it is clear that the running time of the algorithm is $\mathrm{poly}(|\mathbf{x}|, d)$.

**Observation 5.9** *At Step 4, $h_k$ and $h_l$ are factors of $g_i(B\mathbf{x} + \mathbf{d})$ for some $i \in [m]$.*

**Algorithm 2** Make-Factors-Var-Disjoint($g(B\mathbf{x}+\mathbf{d})$)

**Input:** Black-box access to a $g(B\mathbf{x}+\mathbf{d}) \in \mathbb{F}[\mathbf{x}]_{\leq d}$, where $g = g_1 \cdots g_m$ for pairwise variable disjoint $g_1, \ldots, g_m$. ($B \in \mathrm{GL}(|\mathbf{x}|, \mathbb{F})$, $\mathbf{d} \in \mathbb{F}^{|\mathbf{x}|}$, and $g, g_1, \ldots, g_m$ are unknown to the algorithm.)

**Output:** An $A \in \mathrm{GL}(n, \mathbb{F})$ and a set $V$ as stated in Claim 5.3.

1: Factorize $g(B\mathbf{x}+\mathbf{d})$ using Fact 5.2. Let $F \leftarrow \{h_1, \ldots, h_e\}$ be the set of (black-boxes for the) irreducible factors of $g(B\mathbf{x}+\mathbf{d})$.
2: **while** $N_{ess}(\prod_{h\in F} h) \neq \sum_{h\in F} N_{ess}(h)$ **do**
3: For the *first* $l \in [|F|]$ s.t. $N_{ess}(h_1 \cdots h_l) \neq \sum_{j\in[l]} N_{ess}(h_j)$, find a $k \in [l-1]$ s.t. $N_{ess}(h_1 \cdots h_{k-1} \cdot h_l) = \sum_{j\in[k-1]} N_{ess}(h_j) + N_{ess}(h_l)$ but $N_{ess}(h_1 \cdots h_k \cdot h_l) \neq \sum_{j\in[k]} N_{ess}(h_j) + N_{ess}(h_l)$.
4: $F \leftarrow F \cup \{h_k \cdot h_l\}, F \leftarrow F \setminus \{h_k, h_l\}$. Rename the elements of $F$ as $\{h_1, \ldots, h_s\}$.
5: **end while**
6: Let $F = \{h_1, \cdots, h_s\}$. $A \leftarrow$ Make-Polys-Var-Disjoint$(h_1, \ldots, h_s)$ (see Algorithm 1).
7: $V \leftarrow \{\mathrm{var}(h_1(A\mathbf{x})), \ldots, \mathrm{var}(h_s(A\mathbf{x}))\}$.
8: Return $A$ and $V$.

**Proof:** For contradiction, suppose $h_k$ is a factor of $g_i(B\mathbf{x}+\mathbf{d})$ and $h_l$ is a factor of $g_j(B\mathbf{x}+\mathbf{d})$ for $i \neq j$. Let $p$ be the product of all $h \in \{h_1, \ldots, h_{k-1}\}$ such that $h$ is a factor of $g_i(B\mathbf{x}+\mathbf{d})$, $q$ the product of all $h \in \{h_1, \ldots, h_{k-1}\}$ such that $h$ is a factor of $g_j(B\mathbf{x}+\mathbf{d})$, and $r$ the product of all $h \in \{h_1, \ldots, h_{k-1}\}$ such that $h$ is neither a factor of $g_i(B\mathbf{x}+\mathbf{d})$ nor a factor of $g_j(B\mathbf{x}+\mathbf{d})$. Then, $h_1 \cdots h_{k-1} = pqr$. Observe that $N_{ess}(pqrh_l) = N_{ess}(p) + N_{ess}(r) + N_{ess}(qh_l)$, as $g_1, \ldots, g_m$ are pairwise variable disjoint. On the other hand, from the condition $N_{ess}(h_1 \cdots h_{k-1} \cdot h_l) = N_{ess}(h_1) + \ldots N_{ess}(h_{k-1}) + N_{ess}(h_l)$ in Step 3, $N_{ess}(pqrh_l) = N_{ess}(p) + N_{ess}(q) + N_{ess}(r) + N_{ess}(h_l)$. Hence, $N_{ess}(qh_l) = N_{ess}(q) + N_{ess}(h_l)$. For a similar reason, $N_{ess}(ph_k) = N_{ess}(p) + N_{ess}(h_k)$. Now, $N_{ess}(pqrh_kh_l) = N_{ess}(ph_k) + N_{ess}(qh_l) + N_{ess}(r)$, as $g_1, \ldots, g_m$ are variable disjoint. This implies, $N_{ess}(pqrh_kh_l) = N_{ess}(p) + N_{ess}(h_k) + N_{ess}(q) + N_{ess}(h_l) + N_{ess}(r) = \sum_{j\in[k]} N_{ess}(h_j) + N_{ess}(h_l)$, which contradicts the condition $N_{ess}(h_1 \cdots h_k \cdot h_l) \neq N_{ess}(h_1) + \ldots N_{ess}(h_k) + N_{ess}(h_l)$ in Step 3. □

This finishes the proof of Claim 5.3. □

## 5.4 The Hessian of an ROF

In this section, we state some important properties of the Hessian determinant of a canonical ROF. These properties play a crucial role in the equivalence test given in Section 5.5 and allow us to use the Hessian determinant to learn valuable information about the matrix mapping the input polynomial to a canonical ROF. We shall denote the Hessian of a polynomial

$g$ by $H_g$.

**Theorem 5.1 (**$\det(H_{\mathsf{C}}) \neq 0$**)** *Let $n \in \mathbb{N}$ and $\mathrm{char}(\mathbb{F}) = 0$ or $\geq n$. Let $\mathsf{C} = T_1 + \cdots + T_s + \gamma$ be a canonical ROF over $\mathbb{F}$, where for every $k \in [s]$, $T_k$ is a $\times$-rooted canonical ROF, $|\mathrm{var}(T_k)| \leq n$ and $\gamma \in \mathbb{F}$. If for every $k \in [s]$, $T_k$ computes a polynomial of degree at least 2, then $\det(H_{\mathsf{C}}) \neq 0$.*

**Proof:** Notice that $\frac{\partial^2 \mathsf{C}}{\partial x \partial y} = 0$ if $x \in \mathrm{var}(T_k)$ and $y \in \mathrm{var}(T_{k'})$ for $k \neq k' \in [s]$. Thus $H_{\mathsf{C}}$ is a block diagonal matrix with the diagonal blocks being $H_{T_1}, \ldots, H_{T_s}$. Hence, $\det(H_{\mathsf{C}}) = \prod_{k \in [s]} \det\left(H_{T_k}\right)$. So to prove that $\det(H_{\mathsf{C}}) \neq 0$, it suffices to show that $\det\left(H_{T_k}\right) \neq 0$ for all $k \in [s]$.

**Lemma 5.1** *Let $n \in \mathbb{N}$, $\mathbb{F}$ be a field with $\mathrm{char}(\mathbb{F}) = 0$ or $\geq n$, and $\mathbf{x}$ be a variable set with $|\mathbf{x}| \leq n$. If $T$ is a $\times$-rooted canonical ROF computing a polynomial in $\mathbb{F}[\mathbf{x}]$ of degree at least 2, then $\det(H_T) \neq 0$.*

**Proof:** We begin by developing an understanding of the entries of $H_T$. To do this, we first understand the derivatives of $T$. Let $\mathrm{path}(x)$ denote the path from the root of $T$ to the leaf labelled by $x$. For an $x \in \mathbf{x}$, we define the *product-depth of $x$*, denoted by $\Delta_x$, to be the number of $\times$ gates on $\mathrm{path}(x)$. We say that $x$ is a *dangling variable* if $x$ is directly connected to a $+$ gate. For an $x \in \mathbf{x}$, we *expand $T$ along* $\mathrm{path}(x)$ as follows: let $T = Q_{x,1,1} \cdots Q_{x,1,m_1}$, and $x \in \mathrm{var}\left(Q_{x,1,1}\right)$. Let $Q_{x,1,1} = T_{x,1,1} + \cdots + T_{x,1,s_1} + \gamma_1$, and $x \in \mathrm{var}(T_{x,1,1})$. Let $l$ be any number less than $\Delta_x - 1$. After inductively defining $Q_{x,i,j}$ and $T_{x,i,j'}$ for all $i \in [l]$, $j \in [m_i]$, and $j' \in [s_i]$, let $T_{x,l,1} = Q_{x,l+1,1} \cdots Q_{x,l+1,m_{l+1}}$, with $x \in \mathrm{var}(Q_{x,l+1,1})$, and $Q_{x,l+1,1} = T_{x,l+1,1} + \cdots + T_{x,l+1,s_{l+1}} + \gamma_{l+1}$, with $x \in \mathrm{var}(T_{x,l+1,1})$. If $x$ is not a dangling variable, let $T_{x,\Delta_x-1,1} = xQ_{x,\Delta_x,2} \cdots Q_{x,\Delta_x,m_{\Delta_x}}$ (here $Q_{x,\Delta_x,1} = x$). If $x$ is a dangling variable, let $T_{x,\Delta_x-1,1} = Q_{x,\Delta_x,1} \cdots Q_{x,\Delta_x,m_{\Delta_x}}$ and $Q_{x,\Delta_x,1} = x + T_{x,\Delta_x,2} + \cdots + T_{x,\Delta_x,s_{\Delta_x}} + \gamma_{\Delta_x}$ (here $T_{x,\Delta_x,1} = x$). Then, $\frac{\partial T}{\partial x} = \prod_{i \in [\Delta_x]} \prod_{2 \leq j \leq m_i} Q_{x,i,j}$.

**The entries of $H_T$.** For $x, y \in \mathbf{x}$, let $[H_T]_{x,y}$ denote the $(x,y)$-th enrty of $H_T$. Because $T$ is multilinear, $[H_T]_{x,x} = 0$ for all $x \in \mathbf{x}$. For $x \neq y \in \mathbf{x}$, we define the *first common ancestor of $x$ and $y$*, denoted by $\mathrm{fca}(x,y)$, to be the first gate that appears on both the path from the leaf labelled by $x$ to the root of $T$ as well as on the path from the leaf labelled by $y$ to the root of $T$. There are two cases: $\mathrm{fca}(x,y)$ is a $+$ gate and $\mathrm{fca}(x,y)$ is a $\times$ gate. We now describe $\frac{\partial^2 T}{\partial x \partial y}$ in both these cases.

**Observation 5.10** *For all $x \neq y \in \mathbf{x}$ such that $\mathrm{fca}(x,y)$ is a $+$ gate, $[H_T]_{x,y} = [H_T]_{y,x} = 0$.*

**Proof:** Suppose that $\text{fca}(x,y) = Q_{x,l,1}$ for some $1 \leq l \leq \min\{\Delta_x, \Delta_y\}$, and $y \in \text{var}(T_{x,l,2})$. $\frac{\partial^2 T}{\partial x \partial y} = \frac{\partial^2 Q_{x,l,1}}{\partial x \partial y} \cdot \prod_{i \in [l]} \prod_{2 \leq j \leq m_i} Q_{x,i,j} = \left(\frac{\partial^2 T_{x,l,1}}{\partial x \partial y} + \frac{\partial^2 T_{x,l,2}}{\partial x \partial y}\right) \cdot \prod_{i \in [l]} \prod_{2 \leq j \leq m_i} Q_{x,i,j} = 0$. So, $[H_T]_{x,y} = [H_T]_{y,x} = 0$. □

The second case is when $\text{fca}(x,y)$ is a $\times$ gate. Suppose that $\text{fca}(x,y) = T$ or $\text{fca}(x,y) = T_{x,l,1}$ for some $1 \leq l < \min\{\Delta_x, \Delta_y\}$. As we expanded $T$ along path$(x)$, we also expand it along path$(y)$ by defining $Q_{y,i,1}, \ldots, Q_{y,i,m'_i}$ for all $i \in [\Delta_y]$ and $T_{y,i,1}, \ldots, T_{y,i,s'_i}$ for all $i \in [\Delta_y]$ if $y$ is a dangling variable, and for all $i \in [\Delta_y - 1]$ otherwise. Notice that for all $i \in [l]$, every $Q_{x,i,j} = Q_{y,i,j}$ and every $T_{x,i,j'} = T_{y,i,j'}$. Also, we can assume without loss of generality that $Q_{y,l+1,1} = Q_{x,l+1,2}$, $Q_{x,l+1,1} = Q_{y,l+1,2}$, and $Q_{x,l+1,j} = Q_{y,l+1,j}$ for all $3 \leq j \leq m_{l+1} = m'_{l+1}$. Let $\widetilde{Q}_{x,y} = \prod_{i \in [l]} \prod_{2 \leq j \leq m_i} Q_{x,i,j} \cdot \prod_{3 \leq j \leq m_{l+1}} Q_{x,l+1,j}$. Notice that $\widetilde{Q}_{x,y} = \prod_{i \in [l]} \prod_{2 \leq j \leq m'_i} Q_{y,i,j} \cdot \prod_{3 \leq j \leq m'_{l+1}} Q_{y,l+1,j}$. Then,

**Observation 5.11** *For all $x \neq y \in \mathbf{x}$ such that $\text{fca}(x,y)$ is a $\times$ gate,*

$$[H_T]_{x,y} = [H_T]_{y,x} = \widetilde{Q}_{x,y} \prod_{l+1 < i \leq \Delta_x} \prod_{2 \leq j \leq m_i} Q_{x,i,j} \cdot \prod_{l+1 < i \leq \Delta_y} \prod_{2 \leq j \leq m'_i} Q_{y,i,j}.$$

**Proof:**

$$\begin{aligned}
\frac{\partial^2 T}{\partial x \partial y} &= \prod_{i \in [l]} \prod_{2 \leq j \leq m_i} Q_{x,i,j} \cdot \frac{\partial^2 Q_{x,l,1}}{\partial x \partial y} \\
&= \prod_{i \in [l]} \prod_{2 \leq j \leq m_i} Q_{x,i,j} \cdot \frac{\partial^2 T_{x,l,1}}{\partial x \partial y} \\
&= \prod_{i \in [l]} \prod_{2 \leq j \leq m_i} Q_{x,i,j} \cdot \prod_{3 \leq j \leq m_{l+1}} Q_{x,l+1,j} \cdot \frac{\partial Q_{x,l+1,1}}{\partial x} \cdot \frac{\partial Q_{y,l+1,1}}{\partial y} \\
&= \widetilde{Q}_{x,y} \prod_{l+1 < i \leq \Delta_x} \prod_{2 \leq j \leq m_i} Q_{x,i,j} \cdot \prod_{l+1 < i \leq \Delta_y} \prod_{2 \leq j \leq m'_i} Q_{y,i,j}.
\end{aligned}$$

□

Having gained an understanding of the entries of $H_T$, we proceed with the proof of the lemma. We shall call a $\times$-rooted canonical ROF a $(\Delta, m)$ ROF if it has product-depth $\Delta$ and has exactly $m$ many product-depth $\Delta - 1$ ROFs connected to the top-most $\times$ gate. Let $H'_T$ be the matrix obtained from $H_T$ by taking $x^{-1}$ common from the $x$-th row and the $x$-th column

of $H_T$. Observe that for all $x, y \in \mathbf{x}$, $[H'_T]_{x,y} = xy \cdot [H_T]_{x,y}$. Also, notice that it suffices to show that $\det(H'_T) \neq 0$. We show this by induction on tuples of the form $(\Delta, m)$.

**Base case.** $T$ is a $(1, m)$ ROF, where $2 \leq m \leq |\mathbf{x}|$. Then, $T$ is a multilinear monomial, say $x_1 \cdots x_m$, and $\det(H'_T) = (-1)^{m-1}(m-1)\prod_{i\in[m]} x_i^m \neq 0$, as $\mathrm{char}(\mathbb{F}) = 0$ or $\geq n \geq |\mathbf{x}|$.

**Induction step.** $T$ is a $(\Delta, m)$ ROF, for some $\Delta \geq 2$. Assume, by the way of induction, that $\det(H'_{T'}) \neq 0$ for all $(\Delta', m')$ ROFs $T'$, where:

1. $\Delta' = 1$ and $m' \in \{2, \ldots, |\mathbf{x}|\}$, or
2. $1 < \Delta' < \Delta$ and $m' \in [|\mathbf{x}|]$, or
3. $\Delta = \Delta'$ and $m' < m$.

Pick a variable $x \in \mathrm{var}(T)$ as follows: arbitrarily pick a factor of $T$ with product-depth exactly $\Delta - 1$. If there is no dangling variable inside this factor, then let $x$ be any variable in it. Otherwise let $x$ be a dangling variable with the smallest product-depth in it. As before, we expand $T$ along path$(x)$ by defining $Q_{x,i,j}$ for all $i \in [\Delta_x]$ and $T_{x,i,j'}$ for all $i \in [\Delta_x]$ if $x$ is a dangling variable, and for all $i \in [\Delta_x - 1]$ otherwise. Also, we assume without loss of generality that $Q_{x,1,1}, \ldots, Q_{x,1,m}$ are the only sub-ROFs of $T$ with product-depth $\Delta - 1$. If $x$ is not a dangling variable, let $\chi = \Delta_x - 1$; otherwise let $\chi = \Delta_x$. Let

$$U = \{y \in \mathrm{var}(T) : \mathrm{fca}(x, y) \text{ is a } \times \text{ gate}\} \uplus \{x\}$$

and

$$\overline{U} = \mathrm{var}(T) \setminus U = \{y \in \mathrm{var}(T) : \mathrm{fca}(x, y) \text{ is a } + \text{ gate}\}.$$

The following, easy to see observation gives a characterisation of $U$ and $\overline{U}$.

**Observation 5.12** $U = \biguplus_{i\in[\Delta_x]} \biguplus_{2\leq j\leq m_i} \mathrm{var}(Q_{x,i,j}) \uplus \{x\}$ *and* $\overline{U} = \biguplus_{i\in[\chi]} \biguplus_{2\leq j\leq s_i} \mathrm{var}(T_{x,i,j})$.

We now upper bound the degree of $x$ in $\det(H'_T)$, denoted by $\deg_x(\det(H'_T))$, in terms of $|U|$.

**Observation 5.13** $\deg_x(\det(H'_T)) \leq |U|$.

**Proof:** $\det(H'_T) = \sum_{\sigma\in S_{\mathbf{x}}} (-1)^{sgn(\sigma)} \prod_{y\in\mathbf{x}} [H'_T]_{y,\sigma(y)}$, where $S_{\mathbf{x}}$ is the group of permutations of $\mathbf{x}$. It follows from Observations 5.10 and 5.11 that the only rows of $H'_T$ containing $x$ are

the rows labelled by variables in $U$. Thus, for any $\sigma \in S_{\mathbf{x}}$, $[H'_T]_{y,\sigma(y)}$ contains $x$ only if $y \in U$. Hence, at most $|U|$ many entries in $\left\{[H'_T]_{y,\sigma(y)} : y \in \mathbf{x}\right\}$ contain $x$. Also, the degree of $x$ in each of those entries is at most 1. The observation follows. □

Let $N \subset S_{\mathbf{x}}$ be the set of all $\sigma \in S_{\mathbf{x}}$ such that the image of $U$ under $\sigma$ is $U$, and let $\overline{N} = S_{\mathbf{x}} \setminus N$.

**Observation 5.14** *For any $\sigma \in \overline{N}$, $\deg_x\left(\prod_{y\in\mathbf{x}}[H'_T]_{y,\sigma(y)}\right) < |U|$.*

**Proof:** As $\sigma \in \overline{N}$, there exists a $y' \in U$ such that $\sigma(y') \in \overline{U}$. It follows from Observations 5.10 and 5.11 that the only columns of $H'_T$ containing $x$ are the columns labelled by variables in $U$. Hence, $x \notin \text{var}\left([H'_T]_{y',\sigma(y')}\right)$. Then, even if all entries in $\left\{[H'_T]_{y,\sigma(y)} : y \neq y' \in U\right\}$ contain $x$, $\deg_x\left(\prod_{y\in\mathbf{x}}[H'_T]_{y,\sigma(y)}\right) < |U|$. □

Now,

$$\det(H'_T) = \sum_{\sigma\in N}(-1)^{sgn(\sigma)}\prod_{y\in\mathbf{x}}[H'_T]_{y,\sigma(y)} \;+\; \sum_{\sigma\in\overline{N}}(-1)^{sgn(\sigma)}\prod_{y\in\mathbf{x}}[H'_T]_{y,\sigma(y)}.$$

Let the first summand in the above expression be $h$. It follows from Observations 5.13 and 5.14 that to prove $\det(H'_T) \neq 0$, it suffices to show that $\deg_x(h) = |U|$.

**Claim 5.4** $h = \left(\sum_{\sigma_1\in S_U}(-1)^{sgn(\sigma_1)}\prod_{y_1\in U}[H'_T]_{y_1,\sigma_1(y_1)}\right)\cdot\left(\sum_{\sigma_2\in S_{\overline{U}}}(-1)^{sgn(\sigma_2)}\prod_{y_2\in\overline{U}}[H'_T]_{y_2,\sigma_2(y_2)}\right)$.

**Proof:** For any $\sigma \in S_{\mathbf{x}}$, let $\sigma_1$ be $\sigma$ restricted to $U$ and $\sigma_2$ be $\sigma$ restricted to $\overline{U}$. For any $\sigma \in N$, notice that $\sigma_1 \in S_U$ and $\sigma_2 \in S_{\overline{U}}$. Thus,

$$\begin{aligned}
h &= \sum_{\sigma\in N}(-1)^{sgn(\sigma)}\prod_{y\in\mathbf{x}}[H'_T]_{y,\sigma(y)} \\
&= \sum_{\substack{\sigma_1\in S_U,\\ \sigma_2\in S_{\overline{U}}}}(-1)^{sgn(\sigma_1)+sgn(\sigma_2)}\prod_{y_1\in U}[H'_T]_{y_1,\sigma_1(y_1)}\cdot\prod_{y_2\in\overline{U}}[H'_T]_{y_2,\sigma_2(y_2)} \\
&= \sum_{\sigma_1\in S_U}(-1)^{sgn(\sigma_1)}\prod_{y_1\in U}[H'_T]_{y_1,\sigma_1(y_1)}\left(\sum_{\sigma_2\in S_{\overline{U}}}(-1)^{sgn(\sigma_2)}\prod_{y_2\in\overline{U}}[H'_T]_{y_2,\sigma_2(y_2)}\right) \\
&= \left(\sum_{\sigma_1\in S_U}(-1)^{sgn(\sigma_1)}\prod_{y_1\in U}[H'_T]_{y_1,\sigma_1(y_1)}\right)\cdot\left(\sum_{\sigma_2\in S_{\overline{U}}}(-1)^{sgn(\sigma_2)}\prod_{y_2\in\overline{U}}[H'_T]_{y_2,\sigma_2(y_2)}\right).
\end{aligned}$$

□

**Observation 5.15** $\deg_x \left( \sum_{\sigma_2 \in S_{\overline{U}}} (-1)^{sgn(\sigma_2)} \prod_{y_2 \in \overline{U}} [H'_T]_{y_2,\sigma_2(y_2)} \right) = 0.$

**Proof:** It follows from from Observations 5.10 and 5.11 that for no $y_2 \in \overline{U}$, does the row of $H'_T$ labelled by $y_2$ contain $x$. □

**Claim 5.5** $\sum_{\sigma_2 \in S_{\overline{U}}} (-1)^{sgn(\sigma_2)} \prod_{y_2 \in \overline{U}} [H'_T]_{y_2,\sigma_2(y_2)} \neq 0.$

**Proof:** Notice that the given polynomial is the determinant of $[H'_T]_{\overline{U},\overline{U}}$, the sub-matrix of $[H'_T]$ whose rows and columns are labelled by variables in $\overline{U}$. We show that $[H'_T]_{\overline{U},\overline{U}}$ is a block diagonal matrix and all the diagonal blocks have non-zero determinant.

From Observation 5.12, the rows and columns of $[H'_T]_{\overline{U},\overline{U}}$ are labelled by variables in $\biguplus_{i\in[\chi]} \biguplus_{2\leq j\leq s_i} \mathrm{var}(T_{x,i,j})$. We claim that $\frac{\partial^2 T}{\partial x_1 \partial x_2} = 0$ for all $x_1 \in \mathrm{var}(T_{x,i,j})$ and $x_2 \in \mathrm{var}(T_{x,i',j'})$, where $i \neq i'$ or $j \neq j'$. If $i = i'$, then both $T_{x,i,j}$ and $T_{x,i',j'}$ are children of the gate $Q_{x,i,1}$, and $\mathrm{fca}(x_1, x_2) = Q_{x,i,j}$. As $Q_{x,i,j}$ is a $+$ gate, from Observation 5.10, $\frac{\partial^2 T}{\partial x_1 \partial x_2} = 0$. On the other hand, if $i \neq i'$, assume without loss of generality that $i < i'$. Then, observe that $T_{x,i',j'}$ is a sub-ROF of $T_{x,i,1}$. Thus $\mathrm{fca}(x_1, x_2)$ is again $Q_{x,i,j}$, and just as before $\frac{\partial^2 T}{\partial x_1 \partial x_2} = 0$. This implies that $[H'_T]_{\overline{U},\overline{U}}$ is a block diagonal matrix with diagonal blocks $[H'_T]_{\mathrm{var}(T_{x,i,j}),\mathrm{var}(T_{x,i,j})}$ for $i \in [\chi]$ and $2 \leq j \leq s_i$.

We now show that for all $i \in [\chi]$ and $2 \leq j \leq s_i$, the determinant of $[H'_T]_{\mathrm{var}(T_{x,i,j}),\mathrm{var}(T_{x,i,j})}$ is non-zero; this would prove the claim. Fix an $i \in [\chi]$ and a $j \in \{2, \ldots, s_i\}$. Observe that for any $x_1, x_2 \in \mathrm{var}(T_{x,i,j})$, $\frac{\partial^2 T}{\partial x_1 \partial x_2} = \frac{\partial^2 T_{x,i,j}}{\partial x_1 \partial x_2} \cdot \prod_{i'\in[i]} \prod_{2\leq j'\leq m_{i'}} Q_{x,i',j'}$. So, $[H'_T]_{\mathrm{var}(T_{x,i,j}),\mathrm{var}(T_{x,i,j})} = \prod_{i'\in[i]} \prod_{2\leq j'\leq m_{i'}} Q_{x,i',j'} \cdot [H'_{T_{x,i,j}}]$, and it is sufficient to prove that $\det([H'_{T_{x,i,j}}]) \neq 0$.[1] We claim that $T_{x,i,j}$ is not a single variable. The only way it can be a single variable is if it is a dangling variable. If $x$ is not a dangling variable, then because of the way we picked $x$, there is no dangling variable inside $Q_{x,1,1}$. As $T_{x,i,j}$ is a sub-ROF of $Q_{x,1,1}$, it is not a dangling variable. Otherwise, as $x$ is a dangling variable in $Q_{x,1,1}$ with the smallest product-depth, for all $i' \leq \Delta_x - 1$, and $2 \leq j' \leq s_{i'}$, $T_{x,i',j'}$ can not be a dangling variable. Also, $x$ and $T_{x,\Delta_x,2}, \ldots, T_{x,\Delta_x,s_{\Delta_x}}$ are children of the same gate, viz. $Q_{x,\Delta_x,1}$. Because $T$ is a canonical ROF, $T_{x,\Delta_x,2}, \ldots, T_{x,\Delta_x,s_{\Delta_x}}$ can not be dangling variables. Thus, $T_{x,i,j}$ is not a dangling variable, and is a $(\Delta', m')$ ROF for some $\Delta' < \Delta$ such that if $\Delta' = 1$, then $m' \geq 2$. Then it follows from the induction hypothesis that

[1] Since $T_{x,i,j}$ is a $\times$-rooted sub-ROF of $T$, we can define $[H'_{T_{x,i,j}}]$ in the same way as $[H'_T]$.

$\det([H'_{T_{x,i,j}}]) \neq 0$, proving the claim. □

Because of Claim 5.4, Observation 5.15, and Claim 5.5, to prove that $\deg_x(h) = |U|$, we only need to show that for $g := \sum_{\sigma_1 \in S_U} (-1)^{sgn(\sigma_1)} \prod_{y_1 \in U} [H'_T]_{y_1,\sigma_1(y_1)}$, $\deg_x(g) = |U|$. Let $T'$ be the ROF obtained from $T$ by replacing $T_{x,i,2} + \cdots + T_{x,i,s_i} + \gamma_i$ by 0 for all $i \in [\chi]$. Notice that $T' = x \prod_{i \in [\Delta_x]} \prod_{2 \leq j \leq m_i} Q_{x,i,j} = x \cdot \frac{\partial T}{\partial x}$. Hence, $\frac{\partial T}{\partial x} = \frac{\partial T'}{\partial x}$. Also, from Observation 5.12, $\mathrm{var}(T') = U$.

**Claim 5.6** *When $g$ and $\det(H'_{T'})$ are viewed as polynomials over $\mathbb{F}[\mathbf{x} \setminus \{x\}]$, the coefficient of $x^{|U|}$ is same in both the polynomials.*

**Proof:** $g = \sum_{\sigma_1 \in S_U} (-1)^{sgn(\sigma_1)} \prod_{y_1 \in U} [H'_T]_{y_1,\sigma_1(y_1)}$, $\det(H'_{T'}) = \sum_{\sigma_1 \in S_U} (-1)^{sgn(\sigma_1)} \prod_{y_1 \in U} [H'_{T'}]_{y_1,\sigma_1(y_1)}$ and for all $y_1, y_2 \in U$, $[H'_T]_{y_1,y_2}$ and $[H'_{T'}]_{y_1,y_2}$ are multilinear. Thus, it is sufficient to show that the coefficient of $x$ is same in $[H'_T]_{y_1,y_2}$ and $[H'_{T'}]_{y_1,y_2}$ for all $y_1, y_2 \in U$. This is the same as showing that $\frac{\partial [H'_T]_{y_1,y_2}}{\partial x} = \frac{\partial [H'_{T'}]_{y_1,y_2}}{\partial x}$ for all $y_1, y_2 \in U$. There are three cases.

**Case 1:** Neither $y_1$ nor $y_2$ is $x$. Then,

$$\frac{\partial [H'_T]_{y_1,y_2}}{\partial x} = \frac{\partial}{\partial x}\left(y_1 y_2 \frac{\partial^2 T}{\partial y_1 \partial y_2}\right) = y_1 y_2 \frac{\partial^2}{\partial y_1 \partial y_2}\left(\frac{\partial T}{\partial x}\right) = y_1 y_2 \frac{\partial^2}{\partial y_1 \partial y_2}\left(\frac{\partial T'}{\partial x}\right) = \frac{\partial [H'_T]_{y_1,y_2}}{\partial x}.$$

**Case 2:** Exactly one of $y_1$ and $y_2$ is $x$; say $y_1 = x$. Then,

$$\frac{\partial [H'_T]_{y_1,y_2}}{\partial x} = \frac{\partial}{\partial x}\left(x y_2 \frac{\partial^2 T}{\partial x \partial y_2}\right) = y_2 \frac{\partial}{\partial y_2}\left(\frac{\partial T}{\partial x}\right) = y_2 \frac{\partial}{\partial y_2}\left(\frac{\partial T'}{\partial x}\right) = \frac{\partial}{\partial x}\left(x y_2 \frac{\partial^2 T'}{\partial x \partial y_2}\right) = \frac{\partial [H'_{T'}]_{y_1,y_2}}{\partial x}.$$

**Case 3:** $y_1 = y_2 = x$. In this case, both $[H'_T]_{y_1,y_2}$ and $[H'_{T'}]_{y_1,y_2}$ are 0. So, $\frac{\partial [H'_T]_{y_1,y_2}}{\partial x} = \frac{\partial [H'_{T'}]_{y_1,y_2}}{\partial x} = 0$. □

Now, every non-zero entry of $H'_{T'}$ contains $x$ and the rows of $H'_{T'}$ are labelled by variables in $U$. Because we can take $x$ common from all the rows of $H'_{T'}$, if $\det(H'_{T'}) \neq 0$, then $\deg_x(\det(H'_{T'})) = |U|$. Thus Claim 5.6 implies that, $\deg_x(g) = |U|$ if and only if $\det(H'_{T'}) \neq 0$. Recall that $T$ is a $(\Delta, m)$ ROF. If $m \geq 2$, then it follows from the definition of $T'$ that it is a $(\Delta, m-1)$ ROF. Otherwise, if $m = 1$, i.e., if $Q_{x,i,1}$ is the only sub-ROF of $T$ of product-depth $\Delta$, then $T'$ is a $(\Delta', m')$ ROF for some $\Delta' < \Delta$ and $m' \leq |\mathbf{x}|$. Also as $T$ is a $\times$-rooted ROF, its fan-in, $m_1 \geq 2$. Thus, if $\Delta' = 1$, then $m' \geq 2$. So from the induction

hypothesis, we have that $\det(H'_{T'}) \neq 0$. This proves the lemma. □

This concludes the proof of the theorem. □

For our equivalence test, the mere non-zeroness of the Hessian determinant is not enough; we also need some knowledge about its factors. The following claim gives us some of these factors.

**Claim 5.7 (Factors of $\det(H_{\mathtt{C}})$)** *Let $\mathbb{F}$ be an arbitrary field and $\mathtt{C}$ a canonical ROF over $\mathbb{F}$. Let $Q$ be a $+$-rooted sub-ROF of $\mathtt{C}$ or a variable connected to a $\times$ gate in $\mathtt{C}$. Let $Q_1, \ldots, Q_m$ be the siblings of $Q$, i.e., for every $l \in [m]$, $Q_l$ is either a variable or a $+$-rooted sub-ROF of $\mathtt{C}$ and has the same parent as $Q$. If $|\mathrm{var}(Q_1)| + \cdots + |\mathrm{var}(Q_m)| = e$, then the multiplicity of $Q$ as a factor of $\det(H_{\mathtt{C}})$ is at least $(e-1)$.*

**Proof:** Let $T = Q \cdot Q_1 \cdots Q_m$ be a $\times$-rooted sub-ROF of $\mathtt{C}$. Let $\mathtt{C} = T_1 + \cdots + T_s + \gamma$. We saw in the proof of Lemma 5.1 that $\det(H_{\mathtt{C}}) = \prod_{k\in[s]} \det(H_{T_k})$. Thus, if $T$ is a sub-ROF of $T_k$, then it is sufficient to show that $Q^{e-1}$ is a factor of $\det(H_{T_k})$. Let $x \in \biguplus_{l\in[m]} \mathrm{var}(Q_l)$ and consider the $x$-th row of $H_{T_k}$. Like in the proof of Lemma 5.1, we expand $T_k$ along the $\mathrm{path}(x)$. Then for any $y \in \mathrm{var}(T_k)$, $[H_{T_k}]_{x,y} = \frac{\partial}{\partial y}\left(\prod_{i\in[\Delta_x]} \prod_{2\leq j\leq m_i} Q_{x,i,j}\right)$. Notice that for some $\lambda_x \in [\Delta_x]$ and $j \in [m_{\lambda_x}]$, say for $j = 2$, $Q = Q_{x,\lambda_x,2}$. Thus, $Q$ is not a factor of $[H_{T_k}]_{x,y}$ only if $y \in \mathrm{var}(Q)$. For such a $y$,

$$[H_{T_k}]_{x,y} = \frac{\partial Q}{\partial y} \cdot \prod_{i\in[\lambda_x-1]} \prod_{2\leq j\leq m_i} Q_{x,i,j} \cdot \prod_{3\leq j\leq m_{\lambda_x}} Q_{x,\lambda_x,j}. \tag{5.6}$$

We take $Q$ common from every row of $H_{T_k}$ labelled by variables in $\biguplus_{l\in[m]} \mathrm{var}(Q_l)$ to obtain a matrix $H''_{T_k}$. Now, $\det(H_{T_k}) = Q^e \det(H''_{T_k})$. So it suffices to show that $\det(H''_{T_k})$ is either a polynomial, or if it has a denominator, the denominator is just $Q$. Notice that the only entries of $H''_{T_k}$ which are not polynomials but rational functions are $[H''_{T_k}]_{x,y}$, where $x \in \biguplus_{l\in[m]} \mathrm{var}(Q_l)$ and $y \in \mathrm{var}(Q)$. Let $\sigma \in S_{\mathrm{var}(T_k)}$ be any permutation that maps $x_1 \neq x_2 \in \biguplus_{l\in[m]} \mathrm{var}(Q_l)$ to $y_1 \neq y_2 \in \mathrm{var}(Q)$. Define $\sigma' \in S_{\mathrm{var}(T_k)}$ such that it maps $x_1$ to $y_2$, $x_2$ to $y_1$, and for all other $x \in \mathrm{var}(T_k)$, $\sigma'(x) = \sigma(x)$. Then,

$$(-1)^{sgn(\sigma)} \prod_{x \in \text{var}(T_k)} [H''_{T_k}]_{x,\sigma(x)} - (-1)^{sgn(\sigma')} \prod_{x \in \text{var}(T_k)} [H''_{T_k}]_{x,\sigma'(x)}$$
$$= (-1)^{sgn(\sigma)} \prod_{x \in \text{var}(T_k) \setminus \{x_1,x_2\}} [H''_{T_k}]_{x,\sigma(x)} \left( [H''_{T_k}]_{x_1,y_1} [H''_{T_k}]_{x_2,y_2} - [H''_{T_k}]_{x_1,y_2} [H''_{T_k}]_{x_2,y_1} \right).$$

Now, expanding $T$ along path$(x_1)$ as well as path$(x_2)$ we get,

$$Q^2 \left( [H''_{T_k}]_{x_1,y_1} [H''_{T_k}]_{x_2,y_2} - [H''_{T_k}]_{x_1,y_2} [H''_{T_k}]_{x_2,y_1} \right)$$
$$= \left( \frac{\partial Q}{\partial y_1} \cdot \prod_{i \in [\lambda_{x_1}-1]} \prod_{2 \le j \le m_i} Q_{x,i,j} \cdot \prod_{3 \le j \le m_{\lambda_{x_1}}} Q_{x_1,\lambda_{x_1},j} \right) \left( \frac{\partial Q}{\partial y_2} \cdot \prod_{i \in [\lambda_{x_2}-1]} \prod_{2 \le j \le m'_i} Q_{x,i,j} \cdot \prod_{3 \le j \le m'_{\lambda_{x_2}}} Q_{x_2,\lambda_{x_2},j} \right)$$
$$- \left( \frac{\partial Q}{\partial y_2} \cdot \prod_{i \in [\lambda_{x_1}-1]} \prod_{2 \le j \le m_i} Q_{x,i,j} \cdot \prod_{3 \le j \le m_{\lambda_{x_1}}} Q_{x_1,\lambda_{x_1},j} \right) \left( \frac{\partial Q}{\partial y_1} \cdot \prod_{i \in [\lambda_{x_2}-1]} \prod_{2 \le j \le m'_i} Q_{x,i,j} \cdot \prod_{3 \le j \le m'_{\lambda_{x_2}}} Q_{x_2,\lambda_{x_2},j} \right)$$
(from Equation (5.6))
$$= 0.$$

Let $U$ be the set of all permutations $\sigma$ such that $\sigma$ maps at most one variable in $\biguplus_{l \in [m]} \text{var}(Q_l)$ to a variable in var$(Q)$. Then,

$$\det[H''_{T_k}] = \sum_{\sigma \in U} (-1)^{sgn(\sigma)} \prod_{x \in \text{var}(T_k)} [H''_{T_k}]_{x,\sigma(x)}.$$

As at most one of the $\left\{ [H''_{T_k}]_{x,\sigma(x)} : x \in \text{var}(T_k) \right\}$ has a denominator and this denominator is $Q$, either $\det[H''_{T_k}]$ is a polynomial, or if it has a denominator, the denominator is just $Q$. Thus, $Q^{e-1}$ is a factor of $\det(H_{T_k})$. □

The following corollary is an immediate consequence of the above claim.

**Corollary 5.1** *Let $\mathbb{F}$ be an arbitrary field and $\mathsf{C} = T_1 + \cdots + T_s + \gamma$ a canonical ROF over $\mathbb{F}$, where for every $k \in [s]$, $T_k$ is a $\times$-rooted canonical ROF and $\gamma \in \mathbb{F}$. If $k \in [s]$ is such that $T_k$ computes a polynomial of degree at least 3, then there is a $+$-rooted or a variable child $Q$ of $T_k$ such that $Q$ is a factor of* $\det(H_{\mathsf{C}})$.

In the remainder of this section, we describe the variables that are essential for $\det(H_{\mathsf{C}})$ and the variables that do not appear in it. We first define the notion of "skewed paths" which

helps us in characterizing these variables.

**Definition 5.1 (Skewed path)** *Let $Q$ be a $+$-rooted sub-ROF of $\mathtt{C}$ and $T_1, \ldots, T_m$ be the product gates on the path from the root of $\mathtt{C}$ to $Q$. If for all $i \in [m]$, $T_i$ has just two children – a $+$-rooted ROF containing $Q$ and a variable $x_i$ – then we say that the path to $Q$ is* skewed *and identify this path with the "marker" monomial $\mu = \prod_{i \in [m]} x_i$. We say $x_1, \ldots, x_m$ are in the skewed path.*

**Few other terminologies.** We call a variable $x$ a *dangling variable* if its parent in $\mathtt{C}$ is a $+$ gate. For a $+$-rooted sub-ROF $Q = T_1 + \cdots + T_s + \gamma$, where at most one of $T_1, \ldots, T_s$ is a variable and the rest are $\times$-rooted ROFs, we call the sum of all $T_i$ computing a degree two monomial the *quadratic form* of $Q$. Also, a variable $x$ is said to be in the quadratic form of $Q$ if it is in $\mathrm{var}(T_i)$ for some $T_i$ computing a degree two monomial. Suppose that the path to a $+$-rooted sub-ROF $Q$ is skewed, and the skewed path to $Q$ is identified by the monomial $\mu$. Then, if $x$ is a dangling variable connected to the top-most $+$ gate in $Q$, we say that $x$ is the *dangling variable along the skewed path $\mu$*. Similarly, we call the quadratic form of $Q$ the *quadratic form along the skewed path $\mu$*. We now describe the essential variables of $\det(H_{\mathtt{C}})$ using these terminologies.

**Claim 5.8 (Essential variables of $\det(H_{\mathtt{C}})$)** *Let $n \in \mathbb{N}$, $\mathrm{char}(\mathbb{F}) = 0$ or $\geq n$, and $\mathtt{C} = T_1 + \cdots + T_s + \gamma$ be a canonical ROF computing an $n$-variate polynomial such that for all $k \in [s]$, $\deg(T_k) \geq 2$ and $\gamma \in \mathbb{F}$. Then, every variable in $\mathrm{var}(\mathtt{C})$ other than the variables in the quadratic form of the top-most $+$ gate of $\mathtt{C}$, the dangling variables along skewed paths and the variables appearing in the quadratic forms along skewed paths is truly essential for $\det(H_{\mathtt{C}})$.*

**Proof:** From Lemma 5.1, $\det(H_{\mathtt{C}}) \neq 0$. Suppose that $x \in \mathrm{var}(\mathtt{C})$ is not a variable in the quadratic form of the top-most $+$ gate of $\mathtt{C}$, nor a dangling variable along some skewed path, nor a variable appearing in a quadratic form along some skewed path. There there exists a $\times$-gate $T$ on the path from the root of $\mathtt{C}$ to the leaf labelled by $x$ such that if $T = Q_1 \cdots Q_{m'}$, then $x \in \mathrm{var}(Q_1)$ and $|\mathrm{var}(Q_2)| + \cdots + |\mathrm{var}(Q_{m'})| \geq 2$. Then, Claim 5.7 implies that $Q_1$ is a factor of $\det(H_{\mathtt{C}})$. Now $Q_1$ is either a variable or a $+$-rooted sub-ROF, and therefore is irreducible (Fact 2.1). Then, the claim immediately follows from Observation 2.9. □

**Claim 5.9 (Variables of quadratic forms)** *Let $Q$ be a $+$-rooted sub-ROF of $\mathtt{C}$ and $\mathbf{y}$ be the set of all variables in the quadratic form of $Q$. Then, either all $\mathbf{y}$-variables are present in $\det(H_{\mathtt{C}})$ or all are absent. Further if all $\mathbf{y}$-variables are present in $\det(H_{\mathtt{C}})$, then they are also truly essential for $\det(H_{\mathtt{C}})$.*

**Proof:** Suppose that $y_1 \in \mathbf{y}$ is present in $\det(H_{\mathtt{C}})$. Let the quadratic form of $Q$ be $y_1y_2 + \ldots + y_{l-1}y_l$. Let $P$ be a permutation matrix acting on $\mathbf{x} := \mathrm{var}(\mathtt{C})$ such that $P$ maps $y_1$ to $y_2$, $y_2$ to $y_1$, and every other variable to itself. As, $\mathtt{C} = \mathtt{C}(P\mathbf{x})$, $\det\left(H_{\mathtt{C}}\right) = \det\left(H_{\mathtt{C}(P\mathbf{x})}\right)$. Also, from Fact 2.4, $\det\left(H_{\mathtt{C}(P\mathbf{x})}\right) = \det\left(H_{\mathtt{C}}\right)(P\mathbf{x})$. As $y_1$ is present in $\det\left(H_{\mathtt{C}}\right)$, $y_2$ is present in $\det\left(H_{\mathtt{C}}\right)(P\mathbf{x}) = \det\left(H_{\mathtt{C}}\right)$. For any odd $i \leq l-1$, let $P$ be a permutation matrix mapping $y_1$ to $y_i$, $y_2$ to $y_{i+1}$, $y_i$ to $y_1$, $y_{i+1}$ to $y_2$, and all other variables to themselves. Again $\det\left(H_{\mathtt{C}}\right) = \det\left(H_{\mathtt{C}(P\mathbf{x})}\right)$ and $\det\left(H_{\mathtt{C}(P\mathbf{x})}\right) = \det\left(H_{\mathtt{C}}\right)(P\mathbf{x})$. As $y_1$ and $y_2$ appear in $\det\left(H_{\mathtt{C}}\right)$, $y_i$ and $y_{i+1}$ also appear in $\det\left(H_{\mathtt{C}}\right)(P\mathbf{x}) = \det\left(H_{\mathtt{C}}\right)$.

For any odd $i \leq l-1$, let $S$ be a scaling matrix mapping $y_i$ to $2y_i$, $y_{i+1}$ to $\frac{y_{i+1}}{2}$ and every other variable to itself. $\mathtt{C} = \mathtt{C}(S\mathbf{x})$, and hence $\det\left(H_{\mathtt{C}}\right) = \det\left(H_{\mathtt{C}(S\mathbf{x})}\right)$. Also, from Fact 2.4, $\det\left(H_{\mathtt{C}(S\mathbf{x})}\right) = \det(H_{\mathtt{C}})(S\mathbf{x})$. Consider a monomial $\mu$ of $\det\left(H_{\mathtt{C}}\right)$ in which the degree of $y_i$ is $d_i$, that of $y_{i+1}$ is $d_{i+1}$, and whose coefficient is $\beta$. In $\det\left(H_{\mathtt{C}}\right)(S\mathbf{x})$, the coefficient of $\mu$ is $\beta \cdot 2^{d_i - d_{i+1}}$. Thus, $d_i = d_{i+1}$. Then from Observation 2.10 $y_i$ and $y_{i+1}$ are truly essential for $\det\left(H_{\mathtt{C}}\right)$. □

**Claim 5.10 (Missing dangling variables)** *Let $\mathtt{C} = T_1 + \cdots + T_s + \gamma$, where $T_1, \ldots, T_s$ are $\times$-rooted sub-ROFs and $\gamma \in \mathbb{F}$. If for any $k \in [m]$, $T_k = xQ$ for a $+$-rooted sub-ROF $Q$, and $y$ is a dangling variable connected to the top-most $+$ gate of $Q$, then $y$ is not present in* $\det(H_{\mathtt{C}})$.

**Proof:** As argued in the proof of Theorem 5.1, $\det(H_{\mathtt{C}}) = \prod_{k\in[s]} \det\left(H_{T_k}\right)$. It is sufficient to show that $y$ is not present in $\det\left(H_{T_k}\right)$. It follows from Observations 5.10 and 5.11, that $y$ does not appear in any entry of $H_{T_k}$ because the only $x' \in \mathbf{x}$ for which $\frac{\partial^2 T_k}{\partial x' \partial y} \neq 0$ is $x$. But $\frac{\partial^2 T_k}{\partial x' \partial y} = 1$. Hence, $y \notin \mathrm{var}\left(\det\left(H_{T_k}\right)\right)$. □

## 5.5 Equivalence test for ROFs

In this section, we prove Theorem 1.6. Suppose that we are given black-box access to an $f \in \mathbb{F}[\mathbf{x}]$ in the orbit of an unknown canonical ROF $\mathtt{C}$. We can assume that $\mathtt{C}$ is $+$-rooted: Suppose the root of $\mathtt{C}$ is a $\times$ gate and $\mathtt{C} = g_1 \cdots g_m$, where for every $i \in [m]$, $g_i$ is either a variable or a $+$-rooted canonical ROF. We obtain black-box access to the irreducible factors $f_1, \ldots, f_{m'}$ of $f$ using the algorithm in [KT90]. Fact 2.1 implies $m = m'$. We can assume that for every $i \in [m]$, $f_i \in \mathrm{orb}(g_i)$ [1]. Then, we apply the algorithm given in Claim 5.2 on $f_1, \ldots, f_m$ to compute an $A_0 \in \mathrm{GL}(n, \mathbb{F})$ such that $f_1(A_0\mathbf{x}), \ldots, f_m(A_0\mathbf{x})$ are pairwise variable disjoint. For $i \in [m]$, let $\mathbf{x}_i = \mathrm{var}(f_i(A_0\mathbf{x}))$ and $f_i'(\mathbf{x}_i) = f_i(A_0\mathbf{x})$. Suppose, for every $i \in [m]$, we could compute an $A_i \in \mathrm{GL}(|\mathbf{x}_i|, \mathbb{F})$ such that $f_i'(A_i\mathbf{x}_i) \in \text{PS-orb}(g_i)$.

[1] Here we are using a slightly general definition of orbit; see the remark after Definition 2.31.

Let $A := \mathrm{diag}(A_1, \ldots, A_m)$, which is block-diagonal. Then, $f(A_0 A\mathbf{x}) \in \text{PS-orb}(\mathtt{C})$. Thus, the problem reduces to performing equivalence tests for +-rooted canonical ROFs. Before giving the equivalence test, we first give a high-level description of it.

### 5.5.1 An overview of the algorithm

We are given black-box access to an $f \in \mathbb{F}[\mathbf{x}]$ such that there exist a $B \in \mathrm{GL}(n, \mathbb{F})$, a $\mathbf{d} \in \mathbb{F}^n$, and a canonical ROF $\mathtt{C}$ satisfying $f = \mathtt{C}(B\mathbf{x} + \mathbf{d})$. Let $\mathtt{C} = T_1 + \cdots + T_s + \gamma$, where at most one of the *terms* $T_1, \ldots, T_s$ is a variable and the rest are $\times$-rooted ROFs, and $\gamma \in \mathbb{F}$. Also, $f = \widehat{T}_1 + \cdots + \widehat{T}_s + \gamma$, where for all $k \in [s]$, $\widehat{T}_k = T_k(B\mathbf{x} + \mathbf{d})$. The equivalence test can be divided into two phases. In the first phase, we compute an $A_0 \in \mathrm{GL}(n, \mathbb{F})$ such that $\widehat{T}_1(A_0\mathbf{x}), \ldots, \widehat{T}_s(A_0\mathbf{x})$ are variable disjoint. In the second phase, we recursively perform equivalence test on the factors of $\widehat{T}_1(A_0\mathbf{x}), \ldots, \widehat{T}_s(A_0\mathbf{x})$. A pictorial overview of the algorithm is given in Section 5.7.

#### Phase 1: Making terms variable disjoint

We rearrange and divide the terms of $\mathtt{C}$ and of $f$ into four groups: Terms $T_1, \ldots, T_{s_1}$ are called the "good" terms of $\mathtt{C}$ if none of them is a dangling variable, nor a degree 2 monomial, nor does it look like $x \cdot Q$ for some $x \in \mathbf{x}$ and a $+$-rooted ROF $Q$. Similarly, $\widehat{T}_1, \ldots, \widehat{T}_{s_1}$ are the good terms of $f$. Terms $T_{s_1+1}, \ldots, T_{s_2}$ are called the "bad" terms of $\mathtt{C}$ if each of them looks like $x \cdot Q$ for some $x \in \mathbf{x}$ and a $+$-rooted ROF $Q$; similarly $\widehat{T}_{s_1+1}, \ldots, \widehat{T}_{s_2}$ are the bad terms of $f$. Observe that the skewed paths in $\mathtt{C}$ occur only in the bad terms of $\mathtt{C}$. If $\mathtt{C}$ has a top dangling variable, then without loss of generality $T_s = x_n$, and $T_{s_2+1} + \cdots + T_{s'}$ is the top quadratic form where $s' = s - 1$. If $\mathtt{C}$ does not have a top dangling variable, then $T_{s_2+1} + \cdots + T_{s'}$ is the top quadratic form where $s' = s$. If $\mathtt{C}$ has a top dangling variable, then let $\ell := \widehat{T}_s$. This phase can be divided into three steps. In the first step, we make all the good terms variable disjoint. In the second step, we make all the bad and quadratic terms variable disjoint and ensure that $\sum_{k=s_2+1}^{s'} \widehat{T}_k$ maps to $(y_1 + c_1)(y_2 + c_2) + \cdots (y_{l-1} + c_{l-1})(y_l + c_l)$ for some $y_1, \ldots, y_l \in \mathbf{x}$ and $c_1, \ldots, c_l \in \mathbb{F}$. If $\mathtt{C}$ has a top dangling variable, then in the third step, we map $\ell$ to an affine form in a single variable.

**Step 1: Making the good terms variable disjoint.** To make the terms variable disjoint, we make extensive use of the Hessian determinant. If $\mathtt{C}$ does not have a top dangling variable, then $h = \det(H_f) \neq 0$ (see Lemma 5.1 and Fact 2.4). Otherwise, we apply a random transformation $R \in \mathbb{F}^{n \times n}$ to $f$ and compute $h = \det(H_1)$; here $H_1$ is the Hessian

of $f(R\mathbf{x})$ with respect to $\{x_1, \ldots, x_{n-1}\}$. In this case, we refer to $x_n$ as $u_0$. If $\mathtt{C}$ does not have a top dangling variable, then let $R = I_{n\times n}$ and $H_1 = H_f$. Let $H_2$ be the Hessian of $\sum_{k\in[s']} T_k$.[1] Note that $H_1$ and $H_2$ are $n \times n$ matrices if $\mathtt{C}$ has no top dangling variable and $(n-1) \times (n-1)$ matrices otherwise. We show in Claim 5.12 that in both cases, $h$ is a non-zero constant multiple of $\det(H_2)(BR\mathbf{x} + \mathbf{d})$. We then invoke Make-Factors-Variable-Disjoint() (see Claim 5.3) on $h$ to compute an $A_0 \in \mathrm{GL}(n, \mathbb{F})$ that makes the factors of $h$, i.e., $h_1 = \det(H_{T_1})(BR\mathbf{x} + \mathbf{d}), \ldots, h_{s_2} = \det(H_{T_{s_2}})(BR\mathbf{x} + \mathbf{d})$ variable disjoint.[2]

For all $k \in [s_2]$, let $\mathrm{var}(h_k(A_0\mathbf{x})) = \mathbf{z}_k$; as $h_k(A_0\mathbf{x})$ has no redundant variables, all variables in $\mathbf{z}_k$ are essential for it. Let $\mathbf{z}'_k \subseteq \mathbf{z}_k$ be the set of truly essential variables and $\mathbf{z}''_k := \mathbf{z}_k \setminus \mathbf{z}'_k$ the set of ordinary essential variables in $\mathbf{z}_k$. Let $\mathbf{z} = \uplus_{k\in[s_2]}\mathbf{z}_k$ and $\mathbf{y} = \mathbf{x} \setminus \mathbf{z}$. From Claim 5.8, all variables in $\mathtt{C}$ other than the top dangling variable, the variables appearing in the top quadratic form, the dangling variables along skewed paths (see Definition 5.1) and the variables appearing in the quadratic forms along skewed paths are truly essential for $\det(H_2)$. In particular, for all good terms $T_k$, $|\mathrm{var}(T_k)| = |\mathbf{z}'_k| = |\mathbf{z}_k|$; by applying a permutation on the variables in $\mathtt{C}$ if necessary, we can assume that $\mathrm{var}(T_k) = \mathbf{z}'_k = \mathbf{z}_k$. We then argue (using Observation 2.8) that for all $k \in [s_1]$ and all $z \in \mathbf{z}_k$, $BRA_0 \circ z \in \mathbb{F}[\mathbf{z}_k]$. Hence, the good terms $\widehat{T}_1(RA_0\mathbf{x}), \ldots, \widehat{T}_{s_1}(RA_0\mathbf{x})$ are variable disjoint.

We also use Claim 5.1 to compute an affine transformation[3] $C\mathbf{x} + \mathbf{b}$ that maps all the "good" linear factors of $h(A_0\mathbf{x})$ to constant multiples of distinct variables (while preserving the variable disjointness of $\widehat{T}_1(RA_0\mathbf{x}), \ldots, \widehat{T}_{s_1}(RA_0\mathbf{x})$). A linear factor of $h(A_0\mathbf{x})$ is good, if there exists an $x \in \mathbf{x}$ connected to a $\times$ gate (of $\mathtt{C}$) computing a polynomial of degree at least 3 such that $BRA_0\mathbf{x} + \mathbf{d}$ maps $x$ to a constant multiple of that factor. Finally, we update $A_0$ to $RA_0C$ and $\mathbf{b}$ to $RA_0\mathbf{b}$.

**Step 2: Making the bad and quadratic terms variable disjoint.** The only variables in a bad term $T_k$ that need not be truly essential for $\det(H_2)$ are the dangling variables along skewed paths and the variables appearing in the quadratic forms along skewed paths – call these the "bad" variables. We show (using Observation 2.8) that all other variables are already mapped to affine forms in $\mathbf{z}_k$ by $BA_0\mathbf{x} + B\mathbf{b} + \mathbf{d}$. Thus, we only need to handle the linear forms that these bad variables map to. Here skewed paths help us. If $z \in \mathbf{z}'_k$ is a variable in a skewed path in $T_k$ and its sibling in $T_k$ is $Q$, then by "absorbing" an appropriate constant in

[1] We stress that the Hessian of a polynomial $g$ is with respect to $\mathrm{var}(g)$ (unless mentioned otherwise).

[2] We need not mention $\det\left(H_{T_{s_2+1}}\right), \ldots, \det\left(H_{T_{s'}}\right)$ as these are nonzero constants.

[3] Although Phase 1 computes an affine transformation $A_0\mathbf{x} + \mathbf{b}$, it only outputs $A_0$. Indeed, the terms of $f(A_0\mathbf{x} + \mathbf{b})$ are variable disjoint if and only if the terms of $f(A_0\mathbf{x})$ are variable disjoint.

$Q(BA_0\mathbf{x} + B\mathbf{b} + \mathbf{d})$, we can assume that $BA_0\mathbf{x} + B\mathbf{b} + \mathbf{d}$ maps $z$ to a variable $z'$.[1] In fact, by permuting the variables in $\mathsf{C}$ if necessary, we can assume that $z' = z$. Hence, each skewed path in $f(A_0\mathbf{x} + \mathbf{b})$ is a "marker" monomial in $\mathbf{z}$.

*Step 2.1 (Processing quadratic forms along skewed paths).* At first, we treat $f(A_0\mathbf{x} + \mathbf{b})$ as a polynomial in $\mathbf{y} = \mathbf{x} \setminus \mathbf{z}$ over $\mathbb{F}[\mathbf{z}]$ and obtain black-box access to the homogeneous degree-2 component $\widehat{q}$ in $\mathbf{y}$ of $f(A_0\mathbf{x} + \mathbf{b})$. The coefficients of the $\mathbf{y}$-monomials of $\widehat{q}$ are $n$-sparse polynomials in $\mathbb{F}[\mathbf{z}]$; the monomials of these coefficients correspond to skewed paths and the constant terms of these coefficients originate from the top quadratic form of $\mathsf{C}$. As $\widehat{q}$ is an $n^3$-sparse polynomial in $\mathbb{F}[\mathbf{z}, \mathbf{y}]$, we can interpolate it using the sparse polynomial interpolation algorithm in [KS01]. Now, by treating $\widehat{q}$ as a polynomial in $\mathbf{z}$ over $\mathbb{F}[\mathbf{y}]$, we see that the coefficients of the $\mathbf{z}$-monomials of $\widehat{q}$ are related to the "unprocessed" quadratic forms along skewed paths as follows.

Let $q_0$ be the top quadratic form of $\mathsf{C}$, $q_1, \ldots, q_m$ the quadratic forms along skewed paths whose variables do not appear in $\det(H_2)$ (see Claim 5.9), and $\mu_1, \ldots, \mu_m$ the corresponding skewed paths. If $q_i = y_1y_2 + \cdots + y_{l-1}y_l$, then we show that the coefficient of $\mu_i$ in $\widehat{q}$ (if $i = 0$, then the $\mathbb{F}[\mathbf{y}]$-constant term in $\widehat{q}$) is $\widetilde{q}_i := \left[\ell_{y_1}\right]_{\mathbf{y}} \left[\ell_{y_2}\right]_{\mathbf{y}} + \cdots + \left[\ell_{y_{l-1}}\right]_{\mathbf{y}} \left[\ell_{y_l}\right]_{\mathbf{y}}$. Here, for any $x \in \mathbf{x}$, $\ell_x = BA_0 \circ x$ and $[\ell_x]_{\mathbf{y}}$ is $\ell_x$ restricted to the $\mathbf{y}$-variables. So, we can use Claim 5.2 and QFE (see Fact 5.3) to map the coefficients of all the $\mathbf{z}$-monomials of $\widehat{q}$ to variable disjoint, canonical quadratic forms (i.e., quadratic forms that look like $y_1y_2 + \cdots + y_{l-1}y_l$). We then argue (in Claim 5.15) that if $A_1'$ is the matrix obtained by combining the matrices output by QFE on $\widetilde{q}_1, \ldots, \widetilde{q}_m$ and $A_1 = A_0A_1'$, then for $\widehat{q}_i := q_i(B\mathbf{x} + \mathbf{d})$, $\widehat{q}_i(A_1\mathbf{x} + \mathbf{b}) = (y_1 + h_1)(y_2 + h_2) + \cdots + (y_{l-1} + h_{l-1})(y_l + h_l)$ for some (hitherto unknown) affine forms $h_1, \ldots, h_l \in \mathbb{F}[\mathbf{z}]$.[2] We can assume that the $\mathbf{y}$-variables in $q_i$ and $\widehat{q}_i(A_1\mathbf{x} + \mathbf{b})$ are the same by applying a permutation on the variables of $\mathsf{C}$ if necessary.

*Step 2.2 (Handling dangling variables along skewed paths).* Call the $\mathbf{y}$-variables not appearing in $\widehat{q}_0(A_1\mathbf{x} + \mathbf{b}), \ldots, \widehat{q}_m(A_1\mathbf{x} + \mathbf{b})$ the $\mathbf{u}$-variables. The $\mathbf{u}$-variables appear only in the linear forms corresponding to the dangling variables along skewed paths (that are not truly essential for $\det(H_2)$) and the top dangling variable. We treat $f(A_1\mathbf{x} + \mathbf{b})$ as a polynomial in $\mathbf{u}$ over $\mathbb{F}[\mathbf{x} \setminus \mathbf{u}]$ and obtain black-box access to the homogeneous degree-1 component $\widehat{\ell}$ in $\mathbf{u}$ of $f(A_1\mathbf{x} + \mathbf{b})$. The coefficients of the $\mathbf{u}$-variables of $\widehat{\ell}$ are $n$-sparse polynomials in $\mathbb{F}[\mathbf{z}]$; the

[1] Absorbing an appropriate constant into $Q$ (i.e., rescaling $Q$) essentially means that we are starting with a different (but equally valid) $B$ and $\mathbf{d}$. But this is fine as the algorithm is oblivious to the choice of $B$ and $\mathbf{d}$.

[2] The affine form $h_i$ in Step 2.1 should not be confused with the Hessian determinant $h_i$ in Step 1.

monomials of these coefficients correspond to skewed paths and the constant terms of these coefficients originate from the top dangling variable. As $\widehat{\ell}$ is an $n^2$-sparse polynomial in $\mathbb{F}[\mathbf{z}, \mathbf{u}]$, we interpolate it using [KS01]. Now, by treating $\widehat{\ell}$ as a polynomial in $\mathbf{z}$ over $\mathbb{F}[\mathbf{u}]$, we see that the coefficients of the $\mathbf{z}$-monomials of $\widehat{\ell}$ are related to the "unprocessed" dangling variables along skewed paths as follows.

Let $u_0$ be the top dangling variable of $\mathsf{C}$, $x_1, \ldots, x_{m'}$ the dangling variables along skewed paths in $\mathsf{C}$ that are not truly essential for $\det(H_2)$, and $\mu_1, \ldots, \mu_{m'}$ the corresponding skewed paths. Then, $[\ell_{x_i}]_{\mathbf{u}}$ (where $\ell_{x_i}$ is the linear form that $BA_1\mathbf{x}$ maps $x_i$ to) is the coefficient of $\mu_i$ in $\widehat{\ell}$ and the $\mathbb{F}[\mathbf{u}]$-constant term in $\widehat{\ell}$ is $[\ell_{u_0}]_{\mathbf{u}}$. We then find a basis $\mathcal{B}$ of the coefficients of $\mathbf{z}$-monomials (which are linear forms in $\mathbf{u}$) and compute an $A_2'$ that maps these basis elements to distinct $\mathbf{u}$-variables. Now there is a *subtle point* to address here: The size of $\mathcal{B}$ can possibly be strictly less than $m'+1$, which is the number of "unprocessed" dangling variables. And yet we wish to show that $A_2'$ takes care of all the $m'+1$ dangling variables. Let us see (at a high level) how this works.

We argue that the elements of $\mathcal{B}$ are of two kinds. First, we show (in Observation 5.17) that for $x \in \{u_0, x_1, \ldots, x_{m'}\}$, $[\ell_x]_{\mathbf{u}}$ is always in $\mathcal{B}$ if $x$ does not appear in $\det(H_2)$; this part of $\mathcal{B}$ is independent of the choice of the basis $\mathcal{B}$. Second, we show in Claim 5.17 that the remaining elements of $\mathcal{B}$ correspond to a set of redundant variables of $\det(H_2)$ among the variables appearing in $\det(H_2)$; this part varies with the choice of $\mathcal{B}$. This structure of $\mathcal{B}$ helps us prove the following: For $k \in \{s_1+1, \ldots, s_2\}$, let $\mathbf{x}'_k$ be the set of all $x \in \mathrm{var}(T_k)$ such that $[\ell_x]_{\mathbf{u}}$ is in $\mathcal{B}$. Let $\mathbf{y}_k$ be the union of the $\mathbf{y}$-variables present in the quadratic forms along skewed paths in $\widehat{T}_k(A_1A_2'\mathbf{x} + \mathbf{b})$ and the $\mathbf{u}$-variables in $\ell_x(A_2'\mathbf{x})$ for $x \in \mathbf{x}'_k$. Then, we show in Observation 5.19 that as long as we map $\ell_x(A_2'\mathbf{x})$ to a linear form in $\mathbb{F}[\mathbf{z}_k \uplus \mathbf{y}_k]$ for all $x \in \mathbf{x}'_k$, we would have mapped all $\ell_{x'}$, where $x' \in \mathrm{var}(T_k)$ is a dangling variable along a skewed path, to linear forms in $\mathbb{F}[\mathbf{z}_k \uplus \mathbf{y}_k]$.

It follows that $[\ell_{u_0}]_{\mathbf{u}}$ is in $\mathcal{B}$ (as $u_0 \notin \mathrm{var}(\det(H_2))$) and $A_2'$ maps it to $u_0$ (without loss of generality by applying a permutation on $\mathrm{var}(\mathsf{C})$ if required). Apart from $[\ell_{u_0}]_{\mathbf{u}}$, let $[\ell_{x_1}]_{\mathbf{u}}, \ldots, [\ell_{x_m}]_{\mathbf{u}}$ be the other $\mathcal{B}$-elements. Then, $\ell_{u_0}(A_2'\mathbf{x})$ looks like $u_0 + h_{0,1}(\mathbf{y} \setminus \mathbf{u}) + h_{0,2}(\mathbf{z})$ and $\ell_{x_i}(A_2'\mathbf{x})$ looks like $u_i + h_{i,1}(\mathbf{y} \setminus \mathbf{u}) + h_{i,2}(\mathbf{z})$, where $h_{i,1}(\mathbf{y} \setminus \mathbf{u})$ is an affine form in $\mathbf{y} \setminus \mathbf{u}$ and $h_{i,2}(\mathbf{z})$ is an affine form in $\mathbf{z}$ for all $0 \le i \le m$. In fact, by renaming the variables of $\mathsf{C}$ if required, we can assume $x_i = u_i$ for all $i \in [m]$. So, we will refer to $\mathbf{x}'_k$ as $\mathbf{u}_k$. We save $V := \{(1, u_0), (\mu_1, u_1), \ldots, (\mu_m, u_m)\}$ for Step 2.3. Let $A_2 = A_1A_2'$ and $\ell_x$ be the linear form that $BA_2$ maps $x$ to. Our goal in the next step is to compute a linear transformation that for all $k \in \{s_1+1, \ldots, s_2\}$ removes "external" variables, i.e., variables not in $\mathbf{z}_k \uplus \mathbf{y}_k$, from all $\ell_y$ and $\ell_u$ for $y \in \mathbf{y}_k \setminus \mathbf{u}_k$ and $u \in \mathbf{u}_k$. Also, we want to map the top quadratic form to an

expression $(y_1 + c_1)(y_2 + c_2) + \cdots (y_{l-1} + c_{l-1})(y_l + c_l)$, where $c_i \in \mathbb{F}$.

*Step 2.3 (Removing external variables from terms).* We first remove external $\mathbf{z}$-variables from $\ell_y$ for $y \in \mathbf{y} \setminus \mathbf{u}$; such an $\ell_y$ does not have external $\mathbf{y}$-variables. We also remove external $(\mathbf{y} \setminus \mathbf{u})$-variables from $\ell_u$ for $u \in \mathbf{u}$; such an $\ell_u$ does not have external $\mathbf{u}$-variables. This is done as follows: For all $y \in \mathbf{y} \setminus \mathbf{u}$, compute $g = \frac{\partial f(A_2\mathbf{x}+\mathbf{b})}{\partial y}$; it will contain only one $\mathbf{y}$-variable, say $y'$. The sparsity of $g$ is at most $2n$, so we can interpolate it. If $y'$ is multiplied by the $\mathbf{z}$-monomial $\mu$ in $g$ ($\mu$ can be 1), then we express $g$ as $\mu(y' + \ell') + r(\mathbf{z})$, where $\ell'$ is an affine form in $\mathbf{z}$, and $r(\mathbf{z}) \in \mathbb{F}[\mathbf{z}]$. Suppose $y \in \mathbf{y}_k \setminus \mathbf{u}_k$ ($k$ can be figured out from $\mu$). We show in Observation 5.20 that for every $z \notin \mathbf{z}_k$ in $\ell'$, the coefficient $\beta$ of $z$ in $\ell'$ can be assumed to be the same as its coefficient in $\ell_{y'}$. So, by translating $y'$ by $-\beta z$ we can remove $z$ from $\ell_{y'}$. Then, we show in Observation 5.21 that for all monomials $\mu_i$ in $r(\mathbf{z})$ ($\mu_i$ can be 1) such that $(\mu_i, u_i) \in V$ and $u_i \notin \mathbf{u}_k$, the coefficient $\beta$ of $\mu_i$ in $r(\mathbf{z})$ can be assumed to be the same as the coefficient of $y$ in $\ell_{u_i}$. So, to remove $y$ from the latter, we just need to translate $u_i$ by $-\beta y$. The transformation $A_3'$ computed thus removes external $\mathbf{z}$-variables from $\ell_y$ for $y \in \mathbf{y} \setminus \mathbf{u}$ and external $(\mathbf{y} \setminus \mathbf{u})$-variables from $\ell_u$ for $u \in \mathbf{u}$. It also follows from the disambiguation argument in Observations 5.20 and 5.21 that $A_3'$ maps the top quadratic form to $(y_1 + c_1)(y_2 + c_2) + \cdots (y_{l-1} + c_{l-1})(y_l + c_l)$ for $c_i \in \mathbb{F}$.

Let $A_3 = A_2A_3'$ and $\ell_x$ be the linear form that $BA_3$ maps $x$ to. Then, we only need to remove the external $\mathbf{z}$-variables from $\ell_u$ for all $u \in \mathbf{u}_k$ and $k \in \{s_1 + 1, \ldots, s_2\}$. Let $u_i \in \mathbf{u}_k$. To remove $z \notin \mathbf{z}_k$ from $\ell_{u_i}$, we obtain $g$ from $f(A_3\mathbf{x} + \mathbf{b})$ by setting all variables other than $\mathrm{var}(\mu_i)$ and $z$ to 0. Then, using the disambiguation argument in Observation 5.22, we show that the coefficient $\beta$ of $z$ in $\ell_{u_i}$ can be readily derived from the coefficient of $\mu_i$ in $\frac{\partial g}{\partial z}$. Thus, by translating $u_i$ by $-\beta z$, we can remove $z$ from the $\ell_{u_i}$. If $A_4'$ is the transformation computed this way and $A_4 = A_3A_4'$, then in $f(A_4\mathbf{x} + \mathbf{b})$ all non-linear terms are variable disjoint.

**Step 3: Learning the top linear form.** Let $\ell_x$ be the linear form that $BA_4$ maps $x$ to. If $\mathtt{C}$ has a top dangling variable, then Steps 1 and 2 ensure that $u_0$ is only present in the linear form $\ell_{u_0}$. Moreover, Step 2.3 implies that $\ell_{u_0}$ is free of $\mathbf{y} \setminus \{u_0\}$ variables. In particular, none of the variables in the quadratic term $\sum_{k=s_2+1}^{s'} \widehat{T}_k(A_4\mathbf{x} + \mathbf{b})$ is in $\ell_{u_0}$. So, we only need to remove the variables in $\widehat{T}_1(A_4\mathbf{x} + \mathbf{b}), \ldots, \widehat{T}_{s_2}(A_4\mathbf{x} + \mathbf{b})$ from $\ell_{u_0}$. Towards this, we first use second derivatives (in Claim 5.23) to learn $\mathrm{var}(\widehat{T}_1(A_4\mathbf{x} + \mathbf{b})), \ldots, \mathrm{var}(\widehat{T}_{s_2}(A_4\mathbf{x} + \mathbf{b}))$; let these variable sets be $\mathbf{z}_1, \ldots, \mathbf{z}_{s_2}$.[1] Then, we iteratively learn $\ell_{u_0}$ in $s_2$ iterations. In the $k$-

[1] Here, we are overloading the notation a bit. In Steps 1 and 2, $\mathbf{z}_k$ was the variables of the Hessian determinant of $T_k$ evaluated at $BRA_0\mathbf{x} + \mathbf{d}$. In other words, the new set of $\mathbf{z}_k$ variables is the disjoint union of the old

th iteration, we learn $[\ell_{u_0}]_{\mathbf{z}_k}$. To do this, we first obtain $\widehat{T} := \widehat{T}_k(A_4\mathbf{x}+\mathbf{b}) + [\ell_{u_0}]_{\mathbf{z}_k} + \gamma'$, where $\gamma' \in \mathbb{F}$, by setting all but $\mathbf{z}_k$-variables to zero. The argument to learn $[\ell_{u_0}]_{\mathbf{z}_k}$ from $\widehat{T}$ is a generalization of the argument used for '*Finding c*' under Hurdle 3 in Section 5.2.2. However, it is more involved as (unlike the constant $c$ in Section 5.2.2) $[\ell_{u_0}]_{\mathbf{z}_k}$ is *not* uniquely determined. This leads to a couple of complications: One, the circuit that we derive by "learning" $[\ell_{u_0}]_{\mathbf{z}_k}$ is strictly speaking not canonical. Two, it is now unclear how to test if a chosen factor of the Hessian determinant of $\widehat{T}$ is "good". We elaborate on these next to show how the non-uniqueness of $[\ell_{u_0}]_{\mathbf{z}_k}$ is handled.

We use the factors of $h'$, the Hessian determinant of $\widehat{T}$ with respect to the $\mathbf{z}_k$-variables, to learn an affine form $\ell_k$ such that $\widehat{T} - \ell_k$ is reducible. Observe that $h'$ is the Hessian determinant of $T_k$ evaluated at $BA_4\mathbf{x} + B\mathbf{b} + \mathbf{d}$. Corollary 5.1 implies that at least one of the factors of $\widehat{T}_k(A_4\mathbf{x}+\mathbf{b})$ is a factor of $h'$. We will refer to the factors of $h'$ that are also factors of $\widehat{T}_k(A_4\mathbf{x}+\mathbf{b})$ as "good" factors of $h'$. Let $\widehat{Q}$ be a constant multiple of a good factor of $h'$. We now show how to learn $\ell_k$ using $\widehat{Q}$.

*$\widehat{Q}$ is not linear.* Let $\mathbf{a}_1, \ldots, \mathbf{a}_{|\mathbf{z}_k|}$ be random $\mathbb{F}$-vectors of size $|\mathbf{z}_k|$ and $t$ be a fresh variable. For all $i \in [|\mathbf{z}_k|]$, interpolate $\widehat{Q}(t\mathbf{a}_i)$ and $\widehat{T}(t\mathbf{a}_i)$. Discover $\widehat{Q}'_i(t)$ of degree at most $n$ and $\beta_{i,0},\ \beta_{i,1} \in \mathbb{F}$ such that $\widehat{Q}(t\mathbf{a}_i)\cdot\widehat{Q}'_i(t) + \beta_{i,1}\cdot t + \beta_{i,0} = \widehat{T}(t\mathbf{a}_i)$ by solving a system of linear equations in the coefficients of $\widehat{Q}'_i(t)$ and $\beta_{i,0},\ \beta_{i,1}$. One solution is $\widehat{Q}'_i = \left(\widehat{T}_k(A_4\mathbf{x}+\mathbf{b})/\widehat{Q}\right)(t\mathbf{a}_i)$, $\beta_{i,1} = [\ell_{u_0}]_{\mathbf{z}_k}(\mathbf{a}_i)$, and $\beta_{i,0} = \gamma'$. We show in the proof of Claim 5.25 that this solution is unique with high probability over the randomness of $\mathbf{a}_1, \ldots, \mathbf{a}_{|\mathbf{z}_k|}$. Then, we set $\ell_k$ to be the affine form obtained by interpolation using $\beta_{1,1}, \ldots, \beta_{|\mathbf{z}_k|,1}$ and $\beta_{i,0} = \gamma'$. Hence, $\ell_k = [\ell_{u_0}]_{\mathbf{z}_k} + \gamma'$, and $\widehat{T} - \ell_k = \widehat{T}_k(A_4\mathbf{x}+\mathbf{b})$ is reducible.

*$\widehat{Q}$ is linear.* Suppose that $\widehat{Q} = z$ (recall that in Step 1, we would have mapped $\widehat{Q}$ to a single variable). Let $\mathbf{a}_1, \ldots, \mathbf{a}_{|\mathbf{z}_k|-1}$ be random $\mathbb{F}$-vectors of size $|\mathbf{z}_k| - 1$ and $t$ a fresh variable. For all $i \in [|\mathbf{z}_k| - 1]$, interpolate the bivariate polynomial $\widehat{T}(z, \mathbf{z}_k \setminus \{z\} = t\mathbf{a}_i)$. Find $\widehat{Q}'_i(z,t)$ of degree at most $n$ and $\beta_{i,0},\ \beta_{i,1},\ \beta_{i,2} \in \mathbb{F}$ such that $z\widehat{Q}'_i(z,t) + \beta_{i,2}\cdot z + \beta_{i,1}\cdot t + \beta_{i,0} = \widehat{T}(z, \mathbf{z}_k \setminus \{z\} = t\mathbf{a}_i)$ by solving a system of linear equations in the coefficients of $\widehat{Q}'_i(z,t)$ and $\beta_{i,0},\ \beta_{i,1},\ \beta_{i,2}$. One such solution is $\widehat{Q}'_i = \left(\widehat{T}_k(A_4\mathbf{x}+\mathbf{b})/\widehat{Q}\right)(z, t\mathbf{a}_i)$, $\beta_{i,2} = c_z$, the coefficient of $z$ in $[\ell_{u_0}]_{\mathbf{z}_k}$, $\beta_{i,1} = [\ell_{u_0}]_{\mathbf{z}_k\setminus\{z\}}(\mathbf{a}_i)$ and $\beta_{i,0} = \gamma'$. We show in the proof of Claim 5.26 that $\beta_{i,1}$ and $\beta_{i,0}$ are unique with high probability over the randomness of $\mathbf{a}_1, \ldots, \mathbf{a}_{|\mathbf{z}_k|-1}$. So, after using $\beta_{1,1}, \ldots, \beta_{|\mathbf{z}_k|-1,1}$ and $\beta_{i,0} = \gamma'$ to interpolate an affine form $\ell_k$, we get $\ell_k = [\ell_{u_0}]_{\mathbf{z}_k\setminus\{z\}} + \gamma'$. Hence, $\widehat{T} - \ell_k = z(\widehat{T}_k(A_4\mathbf{x}+\mathbf{b})/z + c_z)$ is reducible.

---

$\mathbf{z}_k$ and $\mathbf{y}_k$.

*What if $\widehat{Q}$ is not good?* As at least one factor $\widehat{Q}$ of $h'$ is good, by iterating over all its factors, we ultimately find an $\ell_k$ such that $\widehat{T} - \ell_k$ is reducible. If $\widehat{Q}$ is good, then either $\ell_k = [\ell_{u_0}]_{\mathbf{z}_k} + \gamma'$ and $\widehat{T} - \ell_k = \widehat{T}_k(A_4\mathbf{x} + \mathbf{b})$ or $\ell_k = [\ell_{u_0}]_{\mathbf{z}_k \setminus \{z\}} + \gamma'$ and $\widehat{T} - \ell_k = z(\widehat{T}_k(A_4\mathbf{x} + \mathbf{b})/z + c_z)$ (where $\widehat{Q} = z$). But, what if $\widehat{Q}$ is not good? It turns out that the algorithm only needs to care about finding an $\ell_k$ such that $\widehat{T} - \ell_k$ is reducible; such an $\ell_k$ is always the desired one. This is because, it is implied by the proof of Claim 5.27 that if $\widehat{T} - \ell_k$ is reducible, then the following holds: If $\widehat{T}_k(A_4\mathbf{x} + \mathbf{b})$ has no linear factors, then $\ell_k = [\ell_{u_0}]_{\mathbf{z}_k} + \gamma'$. On the other hand, if $\widehat{T}_k(A_4\mathbf{x} + \mathbf{b}) = \widehat{Q}_{k,1} \cdots \widehat{Q}_{k,m_k}$ and $\widehat{Q}_{k,1} = z$, then $\ell_k$ and $[\ell_{u_0}]_{\mathbf{z}_k}$ must agree on the coefficients of all variables in $\mathbf{z}_k$ except perhaps that of $z$. In both the cases, $\widehat{T} - \ell_k = \widehat{Q}_{k,1}\left(\widehat{Q}_{k,2} \cdots \widehat{Q}_{k,m_k} + c\right)$ for some $c \in \mathbb{F}$. Thus, if we redefine $\widehat{T}_k(A_4\mathbf{x} + \mathbf{b})$ as $\widehat{Q}_{k,1}\left(\widehat{Q}_{k,2} \cdots \widehat{Q}_{k,m_k} + c\right)$, then $\widehat{T}_k(A_4\mathbf{x} + \mathbf{b}) = T'_k(BA_4\mathbf{x} + B\mathbf{b} + \mathbf{d})$, where if $T_k = Q_{k,1} \cdots Q_{k,m_k}$ then $T'_k := Q_{k,1}(Q_{k,2} \cdots Q_{k,m_k} + c)$. Let $\mathtt{C}'$ be obtained from $\mathtt{C}$ by replacing $T_k$ with $T'_k$ whenever necessary. If $A_5$ is obtained from $A_4$ by translating $u_0$ by the linear part of $\sum_{k \in s_2} \ell_k$, then $f(A_5\mathbf{x} + \mathbf{b}) = \widehat{T}_1(A_5\mathbf{x} + \mathbf{b}) + \cdots + \widehat{T}_{s'}(A_5\mathbf{x} + \mathbf{b}) + u_0 + \gamma$ for $\gamma \in \mathbb{F}$, and $f(A_5\mathbf{x} + \mathbf{b}) = \mathtt{C}'(BA_5\mathbf{x} + B\mathbf{b} + \mathbf{d})$. Notice that all the terms of $f(A_5\mathbf{x} + \mathbf{b})$ are variable disjoint.

*Re-canonizing the ROF.* Circuit $\mathtt{C}'$ need not be a canonical ROF as Property 6 of Definition 2.23 may fail. However, for all $k \in [s_2]$, all the factors of $T'_k$ are canonical. As we recursively perform ET on only the factors of $\widehat{T}_k(A_5\mathbf{x})$, $\mathtt{C}'$ not being canonical is not a problem during recursion. However, at the end of the recursion, we are left with reconstructing a canonical ROF where Property 6 may not hold. But this is not an issue as the ROF reconstruction algorithm (Algorithm 9) in Section 5.6 works for canonical ROFs that may not satisfy Property 6. Once an ROF is constructed using Algorithm 9, we go over the ROF in linear time to ensure that Property 6 is satisfied.

**Phase 2: Recursively performing equivalence test on the factors of variable disjoint terms**

To perform equivalence test on the factors $\widehat{Q}_{k,1}(A_5\mathbf{x}), \ldots, \widehat{Q}_{k,m_k}(A_5\mathbf{x})$ of $\widehat{T}_k(A_5\mathbf{x})$ we need to obtain black-box access to each of the factors using *only one query* to the black-box of $f$. It is important that a single query to $f$ is used, or else, the running time of the algorithm will blow up exponentially with the product depth of the ROF. A detailed overview of this phase is already provided in the discussion following Hurdle 3 in Section 5.2.2. Other details are given in Sections 5.5.3.4 and 5.5.3.6.

### 5.5.2 The algorithm

Having given a high level overview of the algorithm, we now describe it formally.

---

**Algorithm 3** Find-Equivalence($f(\mathbf{x})$)

---

**Input:** Black-box access to an $n$-variate polynomial $f$ in the orbit of an unknown $+$-rooted canonical ROF $\mathtt{C}$ such that every $x \in \mathbf{x}$ is essential for $f$.

**Output:** An $A \in \mathrm{GL}(n, \mathbb{F})$ such that $f(A\mathbf{x}) \in \text{PS-orb}(\mathtt{C})$.

/* The base case. */

If $\deg(f) = 1$, return $I_{n\times n}$. Making the non-linear terms of $f$ variable disjoint.

$A_0, \mathbf{b}, \mathbf{z}, \{u_0\}, \{\widehat{\mathbf{z}}_1, \ldots, \widehat{\mathbf{z}}_m\} \leftarrow$ Make-Good-Terms-Var-Disjoint($f$) (Algorithm 4).

$A \leftarrow$ Make-Bad-And-Quadratic-Terms-Var-Disjoint($f(\mathbf{x}), A_0, \mathbf{b}, \mathbf{z}, u_0$) (Algorithm 5).

/* Learning $\mathrm{var}\left(\widehat{T}_1(A\mathbf{x}+\mathbf{b})\right), \ldots, \mathrm{var}\left(\widehat{T}_{s_2}(A\mathbf{x}+\mathbf{b})\right)$. */

$\mathbf{y} \leftarrow \mathbf{x} \setminus \mathbf{z}$. $E \leftarrow \emptyset$, $G \leftarrow (\{\widehat{\mathbf{z}}_1, \ldots, \widehat{\mathbf{z}}_m\} \uplus \mathbf{y}, E)$, a graph.

**for** $i,\ j \in [m]$ **do**

  If for any $z_1 \in \widehat{\mathbf{z}}_i$ and $z_2 \in \widehat{\mathbf{z}}_j$, $\frac{\partial^2 f(A\mathbf{x}+\mathbf{b})}{\partial z_1 \partial z_2} \neq 0$, add edge $\{\widehat{\mathbf{z}}_i, \widehat{\mathbf{z}}_j\}$ to $E$.

**end for**

**for** $i \in [m]$ and $y \in \mathbf{y}$ **do**

  If for any $z \in \widehat{\mathbf{z}}_i$ and $y \in \mathbf{y}$, $\frac{\partial^2 f(A\mathbf{x}+\mathbf{b})}{\partial z \partial y} \neq 0$, add edge $\{\widehat{\mathbf{z}}_i, y\}$ to $E$.

**end for**

$\mathbf{z}_1, \ldots, \mathbf{z}_{s_2} \leftarrow$ the variable sets of size more than 1 corresponding to the different connected components of $G$. $\mathbf{z} \leftarrow \uplus_{k=1}^{s_2} \mathbf{z}_k$, $\mathbf{y} \leftarrow \mathbf{x} \setminus \mathbf{z}$.

/* Learning the top linear form if it exits. */

**if** $\{u_0\} \neq \emptyset$ **then**

  $\ell' \leftarrow$ Find-Top-Linear-Form($f(A\mathbf{x}+\mathbf{b})$) (Algorithm 7). Update $A$ to map $u_0$ to $u_0 - \ell'$.

**end if**

**for** $k \in [s_2]$ **do**

  $\widehat{T} \leftarrow$ Compute-Term-Black-Box($f\left(A\left(\mathbf{z}_k, \mathbf{x} \setminus \mathbf{z}_k = \mathbf{0}\right)\right)$) (Algorithm 8).

/* Making the factors of $\widehat{T}_k(A\mathbf{x})$ variable disjoint. */

  $\widehat{Q}_1, \ldots, \widehat{Q}_{m_k} \leftarrow$ black-boxes of the factors of $\widehat{T}$ obtained using the algorithm in [KT90].

  Compute an $A_{k,0} \in \mathrm{GL}(|\mathbf{z}_k|, \mathbb{F})$ s.t. $\widehat{Q}_1(A_{k,0}\mathbf{z}_k), \ldots, \widehat{Q}_{m_k}(A_{k,0}\mathbf{z}_k)$ are variable disjoint

using Make-Polys-Var-Disjoint() (see Claim 5.2). $\forall l \in [m_k]$, $\widehat{Q}_l \leftarrow \widehat{Q}_l\left(A_{k,0}\mathbf{z}_k\right)$, $\mathbf{z}_{k,l} \leftarrow \text{var}\left(\widehat{Q}_l\right)$.

/* Performing equivalence test on $\widehat{Q}_1, \ldots, \widehat{Q}_{m_k}$. */

$F \leftarrow$ a subset of $\mathbb{F}$ of size $n^5$.[1] $\mathbf{a} \leftarrow$ a vector of size $\mathbf{z}_k$ containing random elements from $F$.

**for** $l \in [m_k]$ **do**

$\mathbf{a}' \leftarrow \mathbf{a}$ restricted to entries corresponding to $\mathbf{z} \setminus \mathbf{z}_{k,l}$.3em

$\beta_l \leftarrow \prod_{l' \in [m_k] \setminus \{l\}} \widehat{Q}_{l'}\left(\mathbf{z}_{k,l}, \mathbf{z}_k \setminus \mathbf{z}_{k,l} = \mathbf{a}'\right)$. $\widehat{Q}_{l'} \leftarrow \beta_l^{-1} \cdot \widehat{T}\left(A_{k,0}\left(\mathbf{z}_{k,l}, \mathbf{z}_k \setminus \mathbf{z}_{k,l} = \mathbf{a}'\right)\right)$. $A_{k,l} \leftarrow \text{Find-Equivalence}\left(\widehat{Q}_l\right)$.

**end for**

Construct an $A'_{k,0} \in \text{GL}(|\mathbf{z}_k|, \mathbb{F})$ that maps every $z \in \mathbf{z}_{k,l}$ to $A_{k,l} \circ z$, $\forall l \in [m_k]$. $A_k \leftarrow A_{k,0}A'_{k,0}$.

**end for**

Construct an $A' \in \text{GL}(n, \mathbb{F})$ that maps all $z \in \mathbf{z}_k$ to $A_k \circ z$ , $\forall k \in [s_2]$ and all $y \in \mathbf{y}$ to $y$. $A \leftarrow AA'$.

**if** $\{u_0\} \neq \emptyset$ **then**

$f' \leftarrow$ Reconstruct-ROF($f(A\mathbf{x})$) (see Section 5.6, Algorithm 9).

For every term of $f'$ that looks like $(\alpha_1 x + \alpha_0)Q$ and $Q$ has a constant $\beta$ attached to the top $+$ gate but not a dangling variable, modify $A$ to map $u_0$ to $u_0 - \alpha_1 \cdot \beta \cdot x$.

**end if**

Return $A$.

---

Recall that the algorithm is given black-box access to an $f \in \mathbb{F}[\mathbf{x}]$ such that there exist a $B \in \text{GL}(n, \mathbb{F})$, a $\mathbf{d} \in \mathbb{F}^n$, and a canonical ROF $\mathtt{C}$ satisfying $f = \mathtt{C}(B\mathbf{x} + \mathbf{d})$. Also, there are no redundant variables in $f$. Further $\mathtt{C} = T_1 + \cdots + T_s + \gamma$, where $T_1, \cdots, T_s$ are $\times$-rooted canonical ROFs and $\gamma \in \mathbb{F}$. Also, $f = \widehat{T}_1 + \cdots + \widehat{T}_s + \gamma$, where for all $k \in [s]$, $T_k(B\mathbf{x} + \mathbf{d}) = \widehat{T}_k$. $T_1, \ldots, T_{s_1}$ are the "good" terms of $\mathtt{C}$, i.e. none of them is a dangling variable, nor a degree 2 monomial, nor does it look like $x \cdot Q$ for some $x \in \mathbf{x}$ and a $+$-rooted ROF $Q$. Similarly, $\widehat{T}_1, \ldots, \widehat{T}_{s_1}$ are the good terms of $f$. $T_{s_1+1}, \ldots, T_{s_2}$ are the "bad" terms of $\mathtt{C}$, i.e. they look like $x \cdot Q$, while $\widehat{T}_{s_1+1}, \ldots, \widehat{T}_{s_2}$ are the bad terms of $f$. If $\mathtt{C}$ has a top dangling variable, $T_s = x_n$, $s' := s - 1$, and $T_{s_2+1} + \cdots + T_{s-1}$ is the top quadratic form. If $\mathtt{C}$ does not have a top dangling variable, then $T_{s_2+1} + \cdots + T_s$ is the top quadratic form and $s' := s$. If $\mathtt{C}$ has a top dangling

[1] Here $n^5$ is somewhat arbitrary. We simply want to ensure that after we apply union bound to the error probabilities in different steps of the algorithm, the total error probability is still small.

variable, then $\ell := \widehat{T}_s$.

We shall give a formal description of the algorithms Make-Good-Terms-Var-Disjoint(), Make-Bad-And-Quadratic-Terms-Var-Disjoint(), Find-Top-Linear-Form(), and Compute-Term-Black-Box() while analysing the algorithm. Make-Good-Terms-Var-Disjoint() outputs an $A_0 \in \mathrm{GL}(n, \mathbb{F})$ and a $\mathbf{b} \in \mathbb{F}^n$ such that $\widehat{T}_1(A_0\mathbf{x}), \ldots, \widehat{T}_{s_1}(A_0\mathbf{x})$ are variable disjoint. Make-Bad-And-Quadratic-Terms-Var-Disjoint() outputs an $A \in \mathrm{GL}(n, \mathbb{F})$ and $\mathbf{b} \in \mathbb{F}^n$ such that $\widehat{T}_1(A\mathbf{x}+\mathbf{b}), \ldots, \widehat{T}_{s'}(A\mathbf{x}+\mathbf{b})$ are variable disjoint and

$$\sum_{k=s_2+1}^{s'} \widehat{T}_k(A\mathbf{x}+\mathbf{b}) = \sum_{k=s_2+1}^{s'} (y_{k,1}+c_{k,1})\,(y_{k,2}+c_{k,2})\,,$$

where $c_{k,1}, c_{k,2} \in \mathbb{F}$. Find-Top-Linear-Form() maps $\ell(A\mathbf{x})$ to a single variable $u_0$. Compute-Term-Black-Box() helps obtain black-box access to $\widehat{T}_1(A\mathbf{x}), \ldots, \widehat{T}_{s_2}(A\mathbf{x})$ using just one query to the black-box of $f$. Finally Steps 21-34 recursively perform equivalence test on the factors of $\widehat{T}_1(A\mathbf{x}), \ldots, \widehat{T}_{s_2}(A\mathbf{x})$.

### 5.5.3 Analysis of the algorithm

In this section, we prove the following lemma. This lemma along with the analysis of the running time in Section 5.5.3.6 proves Theorem 1.6.

**Lemma 5.2 (Correctness of Algorithm 3)** *Let $\mathbb{F}$ be a field of $\mathrm{char}(\mathbb{F}) = 0$ or $\geq n^2$ and $|\mathbb{F}| \geq n^{13}$. Given black-box access to an $n$-variate polynomial $f$ in the orbit of an unknown $+$-rooted canonical ROF $\mathtt{C}$ such that every $x \in \mathbf{x}$ is essential for $f$, Algorithm 3 outputs an $A \in \mathrm{GL}(n, \mathbb{F})$ such that $f(A\mathbf{x}) \in \text{PS-orb}(\mathtt{C})$.*

Towards proving this lemma, we first formally describe the Make-Good-Terms-Var-Disjoint(), Make-Bad-And-Quadratic-Terms-Var-Disjoint(), Find-Top-Linear-Form(), and Compute-Term-Black-Box() algorithms in the following sections.

#### 5.5.3.1 Making the good terms variable disjoint

The following algorithm is used to make all the good terms variable disjoint.

We now prove the following lemma which establishes the correctness of the above algorithm.

**Lemma 5.3 (Correctness of Algorithm 4)** *Make-Good-Terms-Variable-Disjoint($f(\mathbf{x})$) outputs an $A_0 \in \mathrm{GL}(n, \mathbb{F})$ and a $\mathbf{b} \in \mathbb{F}^n$ such that $\widehat{T}_1(A_0\mathbf{x}+\mathbf{b}), \ldots, \widehat{T}_{s_1}(A_0\mathbf{x}+\mathbf{b})$ are pairwise variable disjoint. Further for all $x \in \mathbf{x}$ connected to a $\times$-gate in $\mathtt{C}$ computing a polynomial of degree at*

**Algorithm 4** Make-Good-Terms-Var-Disjoint($f(\mathbf{x})$)

**Input.** $f(\mathbf{x})$, a polynomial in the orbit of a $+$-rooted canonical ROF $\mathtt{C}$.
**Output.**

1. $A_0 \in \mathrm{GL}(n, \mathbb{F}), \mathbf{b} \in \mathbb{F}^n$ such that for $k \neq k' \in [s_1]$, $\mathrm{var}(\widehat{T}_k(A_0\mathbf{x} + \mathbf{b})) \cap \mathrm{var}(\widehat{T}_{k'}(A_0\mathbf{x} + \mathbf{b})) = \emptyset$.

2. For all $x \in \mathbf{x}$ connected to a $\times$-gate in $\mathtt{C}$ computing a polynomial of degree $\geq 3$, $BA_0\mathbf{x} + B\mathbf{b} + \mathbf{d}$ maps $x$ to constant multiple of a variable. If $\mathtt{C}$ has a top dangling variable, then $u_0 = x_n$.

3. $\{\widehat{\mathbf{z}}_1, \ldots, \widehat{\mathbf{z}}_m\}$ such that there exists a partition $I_1, \ldots, I_{s_2}$ of $[m]$ such that for all $k \in [s_2]$, $\uplus_{i \in I_k} \widehat{\mathbf{z}}_i = \mathrm{var}\left(\det(H_{T_k})(BA_0\mathbf{x} + B\mathbf{b} + \mathbf{d})\right)$. $\mathbf{z} = \uplus_{i \in [m]} \widehat{\mathbf{z}}_i$.

1: /* Computing the Hessian determinant of $f$. */
2: **if** $\det(H_f) = 0$ **then**
3: $F \leftarrow$ a subset of $\mathbb{F}$ of size at least $n^5$. $R \leftarrow$ an $n \times n$ random matrix with entries picked independently and uniformly from $F$. $u_0 \leftarrow x_n$.
4: $h \leftarrow$ the Hessian determinant of $f(R\mathbf{x})$ with respect to $\mathbf{x} \setminus \{u_0\}$ variables.
5: $A_0 \leftarrow$ Remove-Redundant-Vars$(h, u_0)$ (see Claim 2.1).
6: **else**
7: $\{u_0\} \leftarrow \emptyset$, $h \leftarrow \det(H_f)$, $R \leftarrow I_{n\times n}$, $A_0 \leftarrow I_{n\times n}$.
8: **end if**
9: $A_0', \widehat{\mathbf{z}}_1, \ldots, \widehat{\mathbf{z}}_m \leftarrow$ Make-Factors-Var-Disjoint$(h(A_0\mathbf{x}))$ (see Claim 5.3). $A_0 \leftarrow A_0A_0'$. $\mathbf{z} \leftarrow \mathrm{var}\left(h(A_0\mathbf{x})\right)$.
10: /* Mapping the good linear factors of $h(A_0\mathbf{x})$ to distinct variables. */
11: $V \leftarrow$ the set of all linear factors of $h(A_0\mathbf{x})$. $C \leftarrow I_{n\times n}$, $\mathbf{b}' \leftarrow \mathbf{0}$.
12: **for** $\ell' \in V$ **do**
13: If $N_{ess}\left(f(RA_0\mathbf{x}) \mod \ell'\right) < n - 2$, pick a $z \in \mathrm{var}(\ell')$ and update $C$ and $\mathbf{b}'$ such that $\ell(C\mathbf{x} + \mathbf{b}') = z$ (see Claim 5.1).
14: **end for**
15: $\mathbf{b} \leftarrow RA_0\mathbf{b}'$, $A_0 \leftarrow RA_0C$. Return $A_0, \mathbf{b}, \mathbf{z}, \{u_0\}, \{\widehat{\mathbf{z}}_1, \ldots, \widehat{\mathbf{z}}_m\}$.

*least 3, $BA_0\mathbf{x} + B\mathbf{b} + \mathbf{d}$ maps $x$ to constant multiple of a variable. Moreover, there exists a partition $I_1, \ldots, I_{s_2}$ of $[m]$ such that for all $k \in [s_2]$, $\uplus_{i \in I_k} \widehat{\mathbf{z}}_i = \text{var}\left(\det(H_{T_k})(BA_0\mathbf{x} + B\mathbf{b} + \mathbf{d})\right)$ and $\mathbf{z} = \uplus_{i \in [m]} \widehat{\mathbf{z}}_i$.*

**Proof:** If $\mathtt{C}$ has no dangling variable, then we know from Lemma 5.1 that $\det(H_{\mathtt{C}}) \neq 0$; from Observation 2.4 this implies $\det(H_f) \neq 0$. Otherwise, we apply a random transformation $R$ to $\mathbf{x}$ in $f$ and compute the Hessian determinant of $f(R\mathbf{x})$ with respect to $\mathbf{x} \setminus \{u_0\} = \{x_1, \ldots, x_{n-1}\}$ variables. Notice that $f(R\mathbf{x}) = \mathtt{C}(BR\mathbf{x} + \mathbf{d})$. The following two claims show that this Hessian determinant is non-zero with high probability.

**Claim 5.11** *The sub-matrix $[BR]_{\mathbf{x}\setminus\{u_0\},\mathbf{x}\setminus\{u_0\}}$ of $BR$, whose rows and columns are labelled by $\mathbf{x} \setminus \{u_0\}$-variables is invertible with high probability.*

**Proof:** $[BR]_{\mathbf{x}\setminus\{u_0\},\mathbf{x}\setminus\{u_0\}} = [B]_{\mathbf{x}\setminus\{u_0\},\mathbf{x}} [R]_{\mathbf{x},\mathbf{x}\setminus\{u_0\}}$, where $[B]_{\mathbf{x}\setminus\{u_0\},\mathbf{x}}$ is the sub-matrix of $B$ whose rows and columns are labelled by variables in $\mathbf{x} \setminus \{u_0\}$ and $\mathbf{x}$, respectively, while $[R]_{\mathbf{x},\mathbf{x}\setminus\{u_0\}}$ is the sub-matrix of $R$ whose rows and columns are labelled by variables in $\mathbf{x}$ and $\mathbf{x} \setminus \{u_0\}$, respectively. For $x \in \mathbf{x} \setminus \{u_0\}$, let $\ell_x$ be the linear form that $x$ is mapped to by $B$. Let $R = (r_{x,x'})_{x,x' \in \mathbf{x}}$. Then the $(x, x')$-the entry of $[BR]_{\mathbf{x}\setminus\{u_0\},\mathbf{x}\setminus\{u_0\}}$ is $\ell_x(\mathbf{r}_{x'})$, where $\mathbf{r}_{x'} = \{r_{x,x'} : x \in \mathbf{x}\}$. As $B$ is invertible, $\{\ell_x(\mathbf{x}) : x \in \mathbf{x} \setminus \{u_0\}\}$ is linearly independent. Thus, the columns of $[BR]_{\mathbf{x}\setminus\{u_0\},\mathbf{x}\setminus\{u_0\}}$ are evaluations of linearly independent, degree 1 polynomials at independently chosen random points from $F^n$, where $|F| \geq n^5$. It is well known (see for instance Claim 2.2 of [KNS19]) that any such matrix is invertible with probability at least $1 - \frac{1}{n^4}$. □

**Claim 5.12** *Let $H_1, H_2$ be the Hessians of $f(R\mathbf{x})$ and $\mathtt{C}$ with respect to $\mathbf{x} \setminus \{u_0\}$-variables, respectively. Then, $h = \det(H_1) = \beta^2 \det(H_2)(BR\mathbf{x} + \mathbf{d})$, where $\beta = \det\left([BR]_{\mathbf{x}\setminus\{u_0\},\mathbf{x}\setminus\{u_0\}}\right)$ and $h \neq 0$ with high probability. Also, $u_0$ is redundant for $h$ with high probability.*

**Proof:** Observe that $H_2$ is the Hessian of $\sum_{k \in [s']} T_k + \gamma$. Then $H_{\mathtt{C}} = \begin{bmatrix} H_2 & 0 \\ 0 & 0 \end{bmatrix}$. Fact 2.2 implies that $H_{f(R\mathbf{x})} = (BR)^T \cdot H_{\mathtt{C}}(BR\mathbf{x} + \mathbf{d}) \cdot (BR)$. It is easy to that $H_1 = [BR]^T_{\mathbf{x}\setminus\{u_0\},\mathbf{x}\setminus\{u_0\}} \cdot H_2(BR\mathbf{x} + \mathbf{d}) \cdot [BR]_{\mathbf{x}\setminus\{u_0\},\mathbf{x}\setminus\{u_0\}}$, which implies $h = \det(H_1) = \beta^2 \det(H_2)(BR\mathbf{x} + \mathbf{d})$, where $\beta$ is the determinant of $[BR]_{\mathbf{x}\setminus\{u_0\},\mathbf{x}\setminus\{u_0\}}$. From Lemma 5.1, $\det(H_2) \neq 0$ and from Claim 5.11 $\beta \neq 0$ with high probability. Hence, $h$ is also non-zero with high probability. Also, $u_0 \notin \text{var}(\det(H_2))$ and hence $N_{ess}(\det(H_2)) \leq n - 1$. Now, $[\nabla h]_{\mathbf{x}\setminus\{u_0\}} = [BR]^T_{\mathbf{x}\setminus\{u_0\},\mathbf{x}\setminus\{u_0\}} \cdot [\nabla \det(H_2)]_{\mathbf{x}\setminus\{u_0\}} (BR\mathbf{x} + \mathbf{d})$, where $[\nabla h]_{\mathbf{x}\setminus\{u_0\}}$ and $[\nabla \det(H_2)]_{\mathbf{x}\setminus\{u_0\}}$ are the gradient vectors of $h$ and $\det(H_2)$ restricted to the entries corresponding to variables in $\mathbf{x} \setminus \{u_0\}$. From Claim

5.11, $[BR]^T_{\mathbf{x}\setminus\{u_0\},\mathbf{x}\setminus\{u_0\}}$ is invertible with high probability. So the spaces $\left\langle \frac{\partial h}{\partial x} : x \in \mathbf{x}\setminus\{u_0\}\right\rangle$ and $\left\langle \frac{\partial}{\partial x}\det(H_2) : x \in \mathbf{x}\setminus\{u_0\}\right\rangle$ have the same dimension with high probability. Then Facts 2.7 and 2.5 imply that a subset of $\mathbf{x}\setminus\{u_0\}$ contains a set of essential variables of $h$ with high probability. Thus, $u_0$ is redundant for $h$ with high probability. □

It is easy to see that $H_2$ is the Hessian of $\sum_{k\in[s']} T_k + \gamma$. Since $H_2$ is a block-diagonal matrix with $H_{T_k}, k \in [s']$ as the diagonal blocks, Claim 5.12 implies that $h = \beta^2 \cdot \prod_{k\in[s']} \det(H_{T_k})(BR\mathbf{x} + \mathbf{d})$. Observe that for every $k \in [s_2]$, $\det(H_{T_k})$ is non-constant. So for every $k \in [s_2]$, $\det\left(H_{T_k}\right)(BRA_0\mathbf{x} + \mathbf{d})$ is a non-constant factor of $h(A_0\mathbf{x})$. Thus, after we compute $A_0'$ by invoking Make-Factors-Var-Disjoint() on $h(A_0\mathbf{x})$ and update $A_0$ to be $A_0A_0'$ in Step 9, Claim 5.3 implies that for $k_1 \neq k_2 \in [s_2]$, $\det\left(H_{T_{k_1}}\right)(BRA_0\mathbf{x} + \mathbf{d})$ and $\det\left(H_{T_{k_2}}\right)(BRA_0\mathbf{x} + \mathbf{d})$ are variable disjoint.

For all $k \in [s]$, let $\mathbf{x}_k = \mathrm{var}(T_k)$, $h_k = \det\left(H_{T_k}\right)(BR\mathbf{x} + \mathbf{d})$, and $g_k = h_k(A_0\mathbf{x})$, where $A_0$ is as after Step 9. Then from Claim 5.3, $g_k$ has no redundant variables. Let $\mathbf{z}_k = \mathrm{var}(g_k)$. Fix a $k \in [s']$. By permuting the variables of $\mathbb{C}$ if necessary, we can assume that $\mathbf{z}_k$ is also a set of essential variables for $h_k' := \det(H_{T_k})$. Let $C_k \in \mathrm{GL}(n, \mathbb{F})$ be a matrix that removes redundant variables from $h_k'$ and $g_k' = h_k'(C_k\mathbf{x})$. Then, $\mathrm{var}(g_k') = \mathrm{var}(g_k) = \mathbf{z}_k$. Let $\mathbf{z}_k'$ be the set of truly essential variables, $\mathbf{z}_k'' = \mathbf{z}_k \setminus \mathbf{z}_k'$ be a set of ordinary essential variables, and $\mathbf{y}_k = \mathbf{x}_k \setminus \mathbf{z}_k$ be a set of redundant variables for $h_k'$. Let $\mathbf{y} = \uplus_{k\in[s']}\mathbf{y}_k \uplus \{u_0\}$. Note that $\mathbf{z} = \mathrm{var}\left(h(A_0\mathbf{x})\right) = \uplus_{k\in[s']}\mathbf{z}_k$, and define $\mathbf{z}' = \uplus_{k\in[s']}\mathbf{z}_k'$, $\mathbf{z}'' = \uplus_{k\in[s']}\mathbf{z}_k''$. Notice that $\mathbf{x} = \mathbf{z}' \uplus \mathbf{z}'' \uplus \mathbf{y}$. For all $x \in \mathbf{x}$ let $\ell_x^{(0)}$ be the linear form that $x$ is mapped to by $BRA_0$.

**Claim 5.13 (Structure of $BRA_0$)** *For every $k \in [s']$,*

1. *For all $z \in \mathbf{z}_k'$, $\ell_z^{(0)} \in \mathbb{F}[\mathbf{z}_k]$.*
2. *For all $z \in \mathbf{z}_k''$, $\ell_z^{(0)} = \ell_z' + \sum_{\substack{y\in\mathbf{y}_k\cap \\ \mathrm{var}\left(h_k'\right)}} \alpha_y \ell_y^{(0)}$, where $\ell_z' \in \mathbb{F}[\mathbf{z}_k]$ and for all $y \in \mathbf{y}_k \cap \mathrm{var}\left(h_k'\right)$, $\alpha_y \in \mathbb{F}$.*

**Proof:** $\widehat{T}_k(R\mathbf{x}) = T_k(BR\mathbf{x} + \mathbf{d})$ implies that $g_k(\mathbf{x}) = g_k'(C_k^{-1}BRA_0\mathbf{x} + C_k^{-1}\mathbf{d})$. As $\mathrm{var}(g_k) = \mathrm{var}(g_k') = \mathbf{z}_k$ and none of them have any redundant variables, $\mathbf{z}_k$ are the truly essential variables of $g$ and $g'$. Thus Observation 2.8 implies that $C_k^{-1}BRA_0$ maps every $z \in \mathbf{z}_k$ to a linear form in $\mathbf{z}_k$. Also, from Fact 2.6 we have that $C_k$ maps every $z \in \mathbf{z}_k'$ and every $y \in \mathbf{y}_k$ to itself, and every $z \in \mathbf{z}_k''$ to a linear form that looks like $z + \sum_{y\in\mathbf{y}_k\cap\mathrm{var}\left(h_k'\right)} \alpha_y y$. Multiplying $C_k$ to $C_k^{-1}BRA_0$ yields the claim. □

Claim 5.8 implies that $\mathbf{z}_k = \mathbf{z}'_k = \mathbf{x}_k$. Hence, the above claim immediately implies that $\widehat{T}_1(A_0\mathbf{x}), \ldots, \widehat{T}_{s_1}(A_0\mathbf{x})$ are pairwise variable disjoint.

Now consider the for loop of lines 12-14. Claim 5.8 implies that for all $x \in \mathbf{x}$ connected to a $\times$ gate in $\mathtt{C}$ computing a polynomial of degree at least three, a constant multiple of the affine form $\ell_x^{(0)} + d_x$ that $BA_0\mathbf{x} + \mathbf{d}$ maps $x$ to is in $V$. Also, as $N_{ess}\left(\mathtt{C} \mod x\right) < n-2$, Fact 2.8 implies that $N_{ess}\left(f(RA_0\mathbf{x}) \mod \left(\ell_x^{(0)} + d_x\right)\right) < n-2$. Conversely, if $\ell' \in V$ is such that $N_{ess}\left(f(RA_0\mathbf{x}) \mod \ell'\right) < n-2$, then it follows and from Claim 5.1 and Fact 2.8 that there exists an $x \in \mathbf{x}$ such that $\ell'$ is a constant multiple of $\ell_x^{(0)} + d_x$. Observe that if $x$ is not connected to a product gate computing a polynomial of degree at least three, then $N_{ess}(\mathtt{C} \mod \alpha x) \geq n-2$ for any $\alpha \in \mathbb{F}^\times$. Hence from Fact 2.8, we have that $x$ is connected to a product gate computing a polynomial of degree at least three. Thus the affine forms $\ell' \in V$ such that $N_{ess}\left(f(RA_0\mathbf{x}) \mod \ell'\right) < n-2$ are exactly the constant multiples of $\ell_x^{(0)} + d_x$, where $x$ is connected to a $\times$ gate computing a polynomial of degree at least three. Because $\left\{\ell_x^{(0)} : x \in \mathbf{x}\right\}$ is linearly independent, it is possible to map all such $\ell'$ to distinct variables.

Observation 2.9 implies that every $x$ connected to a $\times$ gate computing a polynomial of degree at least three is in $\mathbf{z}'$. Also, Claim 5.13 implies that if $x \in \mathbf{z}'_k$, then $\ell' \in \mathbb{F}[\mathbf{z}_k]$. Thus after the loop has been executed and $A_0$ updated to be $RA_0C$, the affine transformation $BA_0\mathbf{x} + B\mathbf{b} + \mathbf{d}$ maps every $x \in \mathbf{x}_k$ connected to a $\times$ gate computing a polynomial of degree at least three to a constant multiple of a $\mathbf{z}_k$-variable. Also this means that $\widehat{T}_1(A_0\mathbf{x}+\mathbf{b}), \ldots, \widehat{T}_{s_1}(A_0\mathbf{x}+\mathbf{b})$ are still variable disjoint.

Immediately before Step 9 is executed, $\det(H_{T_1})(BRA_0\mathbf{x}+\mathbf{d}), \ldots, \det(H_{T_{s_2}})(BRA_0\mathbf{x}+\mathbf{d})$ are non-constant factors of $h(A_0\mathbf{x})$. So from Point 2 of Claim 5.3 there exists a partition $I_1, \ldots, I_{s_2}$ of $[m]$ such that after $A_0$ has been updated to be $A_0A'_0$, for all $k \in [s_2]$, $\uplus_{i\in I_k}\widehat{\mathbf{z}}_i = \mathrm{var}\left(\det(H_{T_k})(BRA_0\mathbf{x}+\mathbf{d})\right)$ and $\mathbf{z} = \uplus_{i\in[m]}\widehat{\mathbf{z}}_i$. As for all $k \in [s_2]$, $C$ only maps some variables in $\mathbf{z}'_k$ to linear forms in $\mathbb{F}[\mathbf{z}'_k]$, we have that $\mathrm{var}\left(\det(H_{T_k})(BRA_0\mathbf{x}+\mathbf{d})\right) = \mathrm{var}\left(\det(H_{T_k})(BRA_0(C\mathbf{x}+\mathbf{b}')+\mathbf{d})\right)$. Because $A_0$ is updated to be $RA_0C$ and $\mathbf{b} := RA_0\mathbf{b}'$ in Step 15, the moreover part of the lemma follows. □

*Remark.* After $A_0$ has been updated to be $RA_0A'_0$ in Step 15, for all $x \in \mathbf{x}$, we redefine $\ell_x^{(0)}$ to be the linear form that $x$ is mapped to by $BA_0$. Notice that Claim 5.13 continues to hold.

#### 5.5.3.2 Making bad and quadratic terms variable disjoint

The following algorithm is used to make all the bad and quadratic terms variable disjoint as well. After this algorithm is executed, all the non-linear terms will be variable disjoint.

For now, we postpone describing the Remove-External-Vars() algorithm and start the

**Algorithm 5** Make-Bad-And-Quadratic-Terms-Var-Disjoint($f(\mathbf{x}), A_0, \mathbf{b}, \mathbf{z}, u_0$)

**Input.** $f(\mathbf{x})$. $A_0$, $\mathbf{b}$, $\mathbf{z}$, and $u_0$ are as returned by Make-Good-Terms-Var-Disjoint($f(\mathbf{x})$).
**Output.** $A \in \mathrm{GL}(n, \mathbb{F})$ such that all the non-linear terms of $f(A\mathbf{x} + \mathbf{b})$ are variable disjoint.

1: $\mathbf{y} \leftarrow \mathbf{x} \setminus \mathbf{z}$. {Discovering the $\mathbf{y}$ parts of quadratic forms.}
2: $\widehat{q} \leftarrow$ the degree-2 homogeneous component in $\mathbf{y}$ of $f(A_0\mathbf{x} + \mathbf{b})$ when it is viewed as a polynomial over $\mathbb{F}[\mathbf{z}]$.
3: Use sparse polynomial interpolation to interpolate $\widehat{q}$. $\{\widetilde{q}_1, \ldots, \widetilde{q}_m\} \leftarrow$ the coefficients of non-constant $\mathbf{z}$-monomials when $\widehat{q}$ is treated as a polynomial in $\mathbb{F}[\mathbf{y}]$. $\widetilde{q}_0 \leftarrow$ coefficient of 1.
4: $A_1' \leftarrow$Make-Polys-Var-Disjoint($\widetilde{q}_0, \ldots, \widetilde{q}_m$) (see Claim 5.2). $\mathbf{u} \leftarrow \mathbf{y}$.
5: **for** $i = 0, \ldots, m$ **do**
6: $\quad p \leftarrow$ the canonical quadratic form in var $(\widetilde{q}_i(A_1'\mathbf{x}))$. $C_i \leftarrow \mathrm{QFE}(\widetilde{q}_i(A_1'\mathbf{x}), p)$. Extend $C_i$ to map every variable in $\mathbf{x} \setminus$ var $(\widetilde{q}_i(A_1'\mathbf{x}))$ to itself. $\mathbf{u} \leftarrow \mathbf{u} \setminus$ var $(\widetilde{q}_i(A_1'\mathbf{x}))$.
7: **end for**
8: $A_1' \leftarrow A_1' \prod_{i=0}^{m} C_i$. $A_1 \leftarrow A_0 A_1'$.
9:
10: /* Discovering the $\mathbf{u}$ parts of dangling linear forms. */
11: $\widehat{\ell} \leftarrow$ the degree-1 homogeneous component in $\mathbf{u}$ of $f(A_1\mathbf{x} + \mathbf{b})$ when it is viewed as a polynomial over $\mathbb{F}[\mathbf{z}]$.
12: Use sparse polynomial interpolation to interpolate $\widehat{\ell}$. Let $\mu_1', \ldots, \mu_{m'}'$ be the non-constant $\mathbf{z}$-monomials of $\widehat{\ell}$, and $\widehat{\ell}_1 \ldots, \widehat{\ell}_{m'}$ be their coefficients. $\widehat{\ell}_0 \leftarrow$ the coefficient of 1.
13: $\widehat{\ell}_{i_1}, \ldots, \widehat{\ell}_{i_m} \leftarrow$ a basis of $\left\langle \widehat{\ell}_1, \ldots, \widehat{\ell}_{m'} \right\rangle$. Construct a matrix $A_2'$ that maps $\widehat{\ell}_{i_1}, \ldots, \widehat{\ell}_{i_m}$ to distinct $\mathbf{u}$ variables, say $u_{i_1}, \ldots, u_{i_m}$, such that if $\widehat{\ell}_0$ maps to $u_0$. Also, $A_2'$ acts as identity on $\mathbf{x} \setminus \mathbf{u}$.
14: $V \leftarrow \left\{ (\mu_{i_1}', u_{i_1}), \ldots, (\mu_{i_m}', u_{i_m}) \right\}$. $A_2 \leftarrow A_1 A_2'$.
15: $C \leftarrow$ Remove-External-Vars($f(\mathbf{x}), A_2, \mathbf{b}, \mathbf{z}, \mathbf{y}, \mathbf{u}, u_0, V$) (Algorithm 6).
16: $A \leftarrow A_2 C$. Return $A$.

proof of correctness of the Make-Bad-And-Quadratic-Terms-Var-Disjoint() algorithm. In particular, we prove the following lemma.

**Lemma 5.4 (Correctness of Algorithm 5)** *Make-Bad-And-Quadratic-Terms-Var-Disjoint($f(\mathbf{x}), A_0, \mathbf{b}, \mathbf{z}, u_0$), where $A_0, \mathbf{b}, \mathbf{z}$, and $u_0$ are as returned by Make-Good-Terms-Var-Disjoint($f(\mathbf{x})$) outputs an $A \in \mathrm{GL}(n, \mathbb{F})$ such that $\widehat{T}_1(A\mathbf{x}+\mathbf{b}), \ldots, \widehat{T}_{s'}(A\mathbf{x}+\mathbf{b})$ are pairwise variable disjoint. Also, $\sum_{k=s_2+1}^{s'} \widehat{T}_k(A\mathbf{x}+\mathbf{b}) = \sum_{k=s_2+1}^{s'}(y_{k,1}+c_{k,1})(y_{k,2}+c_{k,2})$, where for all $k \in \{s_2+1, \ldots, s'\}$, $c_{k,1}, c_{k,2} \in \mathbb{F}$. Further, for all $x \in \mathbf{x}$ connected to a $\times$-gate in $\mathtt{C}$ computing a polynomial of degree at least 3, $BA\mathbf{x} + B\mathbf{b} + \mathbf{d}$ maps $x$ to constant multiple of a variable. Also, if $\mathtt{C}$ has a top dangling variable, then $u_0$ only appears in $\ell(A\mathbf{x}+\mathbf{b})$.*

**Proof:** We begin by stating the following useful claim. For a linear form $\ell'$, we denote its projection to variables in a variable set $\mathbf{x}''$ by $[\ell']_{\mathbf{x}''}$. Recall that for an $x \in \mathbf{x}$, $\ell_x^{(0)}$ is the linear form that $x$ is mapped to by $BA_0$.

**Claim 5.14** $\left\{ \left[\ell_y^{(0)}\right]_{\mathbf{y}} : y \in \mathbf{y} \right\}$ *is linearly independent.*

**Proof:** $BA_0$ is invertible and from Claim 5.13, for every $z \in \mathbf{z}'$, $\left[\ell_z^{(0)}\right]_{\mathbf{y}} = 0$. Hence, the sub-matrix $[BA_0]_{\mathbf{z}'' \uplus \mathbf{y}, \mathbf{y}}$ of $BA_0$ containing rows corresponding to variables in $\mathbf{z}'' \uplus \mathbf{y}$ and columns corresponding to variables in $\mathbf{y}$ is full rank. From Claim 5.13, we have that all rows of $[BA_0]_{\mathbf{z}'', \mathbf{y}}$ are in the $\mathbb{F}$-span of the rows of $[BA_0]_{\mathbf{y},\mathbf{y}}$. Thus $[BA_0]_{\mathbf{y},\mathbf{y}}$ is full rank. The claim follows by noticing that the entries of $[BA_0]_{\mathbf{y},\mathbf{y}}\,\mathbf{y}$ are exactly the linear forms $\left[\ell_y^{(0)}\right]_{\mathbf{y}}$ for all $y \in \mathbf{y}$. □

From Claim 5.8, only the top dangling variable, the variables in the top quadratic form, the dangling variables along skewed paths, and variables appearing in the quadratic forms along the skewed paths in $\mathtt{C}$ need not be truly essential for $\det(H_2)$. Hence, from Claim 5.13, if for some $k \in [s']$, $x \in \mathbf{x}_k$ is such that $\ell_x^{(0)} \notin \mathbb{F}[\mathbf{z}_k]$, then $k \in \{s_1+1, \ldots, s_2\}$ and $x$ is either a dangling variable along a skewed path or a variable appearing in some quadratic form along a skewed path in $T_k$ which is not truly essential for $\det(H_2)$, or $k \in \{s_2+1, \ldots, s'\}$ and $x$ is a variable appearing in the top quadratic form of $\mathtt{C}$. Also, if $x$ is a variable appearing in a skewed path in $T_k$, then from Lemma 5.3, $\ell_x^{(0)} + b_x = \alpha z$ for some $z \in \mathbf{z}_k$ and $\alpha \in \mathbb{F}^\times$. Suppose that the other gate connected to the parent of $x$ is $Q$. Then by 'absorbing' $\beta$ inside $Q(BA_0\mathbf{x} + B\mathbf{b} + \mathbf{d})$, we can assume without loss of generality that $\beta = 1$.[1] Thus each

[1] As mentioned in Section 5.5.1, absorbing $\beta$ in $Q$ means that we are starting with a different but equally valid $B$. This 'new' $B$ is obtained from the 'old' $B$ by scaling rows labelled by appropriate variables in $\mathbf{z}'$. Hence, Claim 5.13 continues to hold.

skewed path is a monomial in $\mathbb{F}[\mathbf{z}]$. Also, we can assume without loss of generality that every variable appearing in a skewed path in $\mathsf{C}$ is mapped to itself by the affine transformation $BA_0\mathbf{x} + B\mathbf{b} + \mathbf{d}$, i.e. the skewed paths in $\mathsf{C}$ and $f(A_0\mathbf{x} + \mathbf{b})$ are the same. This is so because if a variable $x$ appearing in a skewed path in $\mathsf{C}$ is mapped to $z \neq x$, then we can permute the variables in $\mathsf{C}$ so that the leaf labelled by $x$ is now labelled by $z$.[1]

Let $q_0$ be the top quadratic form of $\mathsf{C}$ and $\mu_1, \ldots, \mu_m$ be all the skewed paths in $f(A_0\mathbf{x} + \mathbf{b})$ such that no variable of the quadratic forms $q_1, \ldots, q_m$ corresponding to these skewed paths in $\mathsf{C}$ appears in $\det(H_2)$ (see Claim 5.9). Also, let the corresponding quadratic forms in $f(A_0\mathbf{x} + \mathbf{b})$ be $\widehat{q}_0, \ldots, \widehat{q}_m$. Then for all $i \in \{0, \ldots, m\}$, $\widehat{q}_i = q_i(BA_0\mathbf{x} + B\mathbf{b} + \mathbf{d})$. Suppose that $\widehat{q}_i = \ell^{(0)}_{y_{i,1,1}}\ell^{(0)}_{y_{i,1,2}} + \cdots + \ell^{(0)}_{y_{i,m_i,1}}\ell^{(0)}_{y_{i,m_i,2}}$. It follows from the discussion in the above paragraph that $\widehat{q} = \widetilde{q}_0 + \mu_1\widetilde{q}_1 + \cdots + \mu_m\widetilde{q}_m$, where $\widetilde{q}_i = \left[\ell^{(0)}_{y_{i,1,1}}\right]_{\mathbf{y}}\left[\ell^{(0)}_{y_{i,1,2}}\right]_{\mathbf{y}} + \cdots + \left[\ell^{(0)}_{y_{i,m_i,1}}\right]_{\mathbf{y}}\left[\ell^{(0)}_{y_{i,m_i,2}}\right]_{\mathbf{y}}$. Observe that each $\widetilde{q}_i$ is an $n^2$-sparse polynomial. As there are at most $n$ skewed paths in $\mathsf{C}$, this means that $\widehat{q}$ is an $n^3$-sparse polynomial and can be interpolated efficiently. Claim 5.2 ensures that after Step 4 is executed, $\widetilde{q}_0(A'_1\mathbf{x}), \ldots, \widetilde{q}_m(A'_1\mathbf{x})$ are variable disjoint and have no redundant variables.

**Claim 5.15** *After the for loop of lines 5-7 has been executed and $A'_1$ updated to be $A'_1\prod_{i=0}^{m} C_i$, for all $i \in \{0, \ldots, m\}$, $\widehat{q}_i(A'_1\mathbf{x}) = \left(y'_{i,1,1} + h_{i,1,1}\right)\left(y'_{i,1,2} + h_{i,1,2}\right) + \cdots + \left(y'_{i,m_i,1} + h_{i,m_i,1}\right)\left(y'_{i,m_i,2} + h_{i,m_i,2}\right)$, for some $y'_{i,1,1}, y'_{i,1,2}, \ldots, y'_{i,m_i,1}, y'_{i,m_i,2} \in \mathbf{y}$ and affine forms $h_{i,1,1}, h_{i,1,2}, \ldots, h_{i,m_i,1}, h_{i,m_i,2} \in \mathbb{F}[\mathbf{z}]$.*

**Proof:** Observe that for every $i \in [m]$, the $i$-th iteration of the loop only works with $\widetilde{q}_i(A'_1\mathbf{x})$ (where $A'_1$ is as in Step 4) and computes a $C_i$ which only acts non-trivially on var $(\widetilde{q}_i(A'_1\mathbf{x}))$. Thus, we can analyse every iteration of the loop in isolation, and it sufficient to prove that after the $i$-th iteration,

$$\widehat{q}_i(A'_1C_i\mathbf{x}) = \left(y'_{i,1,1} + h_{i,1,1}\right)\left(y'_{i,1,2} + h_{i,1,2}\right) + \cdots + \left(y'_{i,m_i,1} + h_{i,m_i,1}\right)\left(y'_{i,m_i,2} + h_{i,m_i,2}\right).$$

Fix an $i \in \{0, \ldots, m\}$ and let var $(\widetilde{q}_i(A'_1\mathbf{x})) = \left\{y'_{i,1,1}, y'_{i,1,2}, \ldots, y'_{i,m_i,1}, y'_{i,m_i,2}\right\}$. As mentioned before, $\widetilde{q}_i(A'_1\mathbf{x})$ has no redundant variables. Thus, $C_i \in \mathrm{GL}(2m_i, \mathbb{F})$ output by the QFE

[1] If the permutation matrix that we need to apply to $\mathsf{C}$ is $P$, then the new ROF is $\mathsf{C}(P\mathbf{x})$ and the matrix transforming it to $f(A_0\mathbf{x} + \mathbf{b})$ is $P^{-1}BA_0$. While we proved Claim 5.13 for $\mathsf{C}$ and $BA_0$, it continues to hold for $\mathsf{C}(P\mathbf{x})$ and $P^{-1}BA_0$. This is so, because if the leaf in $\mathsf{C}$ labelled by $x$ is labelled by $x'$ in $\mathsf{C}(P\mathbf{x})$, then the linear form that $P^{-1}BA_0$ maps $x'$ to, i.e., the 'new' $\ell^{(0)}_{x'}$, is the 'old' $\ell^{(0)}_{x}$. In other words, permuting the variables in $\mathsf{C}$ just results in the leafs of $\mathsf{C}$ and the rows of $BA_0$ being relabelled consistently. Through out the analysis, we shall permute the variables of $\mathsf{C}$ many times, however each time we do this, everything that we have proved up to that point for $\mathsf{C}$ and the matrix transforming it to $f(A_0\mathbf{x} + \mathbf{b})$ would continue to hold for the new $\mathsf{C}$ and the new matrix.

algorithm is such that after it has been extended to map every variable in $\mathbf{x} \setminus \text{var}(\widetilde{q}_i(A_1'\mathbf{x}))$ to itself, $\widetilde{q}_i(A_1'C_i\mathbf{x}) = y'_{i,1,1}y'_{i,1,2} + \cdots + y'_{i,m_i,1}y'_{i,m_i,2}$.

For all $j \in [m_i]$ and $l \in [2]$, let $\alpha_{l,j}$ be the $y_{i,j,l}$-th entry of $B\mathbf{b} + \mathbf{d}$ and $p_{j,l} = \left[\ell^{(0)}_{y_{i,j,l}}\right]_{\mathbf{z}} + \alpha_{l,j}$. Then,

$$\sum_{j\in[m_i]} \left(\left[\ell^{(0)}_{y_{i,j,1}}\right]_{\mathbf{y}} + p_{j,1}\right)\left(\left[\ell^{(0)}_{y_{i,j,2}}\right]_{\mathbf{y}} + p_{j,2}\right)(A_1'C_i\mathbf{x}) = \sum_{j\in[m_i]} (\ell_{j,1} + p_{j,1})\,(\ell_{j,2} + p_{j,2})\,,$$

where for $j \in [m_i]$, $l \in [2]$, $\ell_{j,l} := \left[\ell^{(0)}_{y_{i,j,1}}\right]_{\mathbf{y}}(A_1'C_i\mathbf{x})$. Since $A_1'C_i \in \mathrm{GL}(n, \mathbb{F})$, Claim 5.14 implies that $\{\ell_{j,l} : j \in [m_i],\ l \in [2]\}$ is linearly independent. Now $\widetilde{q}_i(A_1'C_i) = \sum_{j\in[m_i]} \ell_{j,1}\ell_{j,2}$ and $\widetilde{q} \in \mathrm{orb}(q_i)$. Also neither $q_i$ nor $\widetilde{q}_i$ have any redundant variables. Hence from Observation 2.8, for all $j \in [m_i]$ and $l \in [2]$, $\ell_{j,l}$ is a linear form solely in $\left\{y'_{i,1,1},\ y'_{i,1,2}, \ldots, y'_{i,m_i,1},\ y'_{i,m_i,2}\right\}$. Expanding the right hand side of the above equation,

$$\sum_{j\in[m_i]} (\ell_{j,1} + p_{j,1})(\ell_{j,2} + p_{j,2}) = \sum_{j\in[m_i]} \ell_{j,1}\ell_{j,2} + \sum_{j\in[m_i]} (\ell_{j,1}p_{j,2} + \ell_{j,2}p_{j,1}) + \sum_{j\in[m_i]} p_{j,1}p_{j,2}. \tag{5.7}$$

For $j \in [m_i]$, let $h_{i,j,1}$ and $h_{i,j,2}$ be the coefficients of $y_{i,j,2}$ and $y_{i,j,1}$ in $\sum_{j\in[m_i]}(\ell_{j,1}p_{j,2} + \ell_{j,2}p_{j,1})$ respectively. Then, $h_{i,j,1}, h_{i,j,2} \in \mathbb{F}[\mathbf{z}]$ are linear polynomials and $\sum_{j\in[m_i]}(\ell_{j,1}p_{j,2} + \ell_{j,2}p_{j,1}) = \sum_{j\in[m_i]}(y_{i,j,1}h_{i,j,2} + y_{i,j,2}h_{i,j,1})$. Now, $\sum_{j\in[m_i]} \ell_{j,1}\ell_{j,2} = \sum_{j\in[m_i]} y_{i,j,1}y_{i,j,2}$. Putting these in equation (5.7),

$$\sum_{j\in[m_i]} (\ell_{j,1} + p_{j,1})(\ell_{j,2} + p_{j,2}) = \sum_{j\in[m_i]} (y_{i,j,1} + h_{i,j,1})(y_{i,j,2} + h_{i,j,2}) + \sum_{j\in[m_i]} (p_{j,1}p_{j,2} - h_{i,j,1}h_{i,j,2}) \tag{5.8}$$

Substitute $y_{i,j,l} = y_{i,j,l} - h_{i,j,l}$ for every $j \in [m_i], l \in [2]$ in the above equation. Then we get

$$\sum_{j\in[m_i]} (\ell_{j,1} + p'_{j,1})(\ell_{j,2} + p'_{j,2}) = \sum_{j\in[m_i]} y_{i,j,1}y_{i,j,2} + \sum_{j\in[m_i]} (p_{j,1}p_{j,2} - h_{i,j,1}h_{i,j,2}),$$

where for every $j \in [m_i]$, $l \in [2]$, $p'_{j,l} \in \mathbb{F}[\mathbf{z}]$ is a linear polynomial. Note that the right hand side of the above equation does not have a monomial containing variables from both $\mathbf{y}$ and $\mathbf{z}$. Thus we get $\sum_{j\in[m_i]}(\ell_{j,1}p'_{j,2} + \ell_{j,2}p'_{j,1}) = 0$. Since $\{\ell_{j,l} : j \in [m_i],\ l \in [2]\}$ is linearly independent, it is easy to see that for every $j \in [m_i], p'_{j,1} = p'_{j,2} = 0$, which implies

$\sum_{j\in[m_i]}(p_{j,1}p_{j,2} - h_{i,j,1}h_{i,j,2}) = 0$. Hence,

$$\begin{aligned}
\widehat{q}_i(A_1'C_i\mathbf{x}) &= \sum_{j\in[m_i]} \left(\left[\ell^{(0)}_{y_{i,j,1}}\right]_{\mathbf{y}} + p_{j,1}\right)\left(\left[\ell^{(0)}_{y_{i,j,2}}\right]_{\mathbf{y}} + p_{j,2}\right)(A_1'C_i\mathbf{x}) \\
&= \sum_{j\in[m_i]} (\ell_{j,1} + p_{j,1})(\ell_{j,2} + p_{j,2}) \\
&= \sum_{j\in[m_i]} (y_{i,j,1} + h_{i,j,1})(y_{i,j,2} + h_{i,j,2}) \qquad \text{(from Equation (5.8)).}
\end{aligned}$$

□

The following observation is easy to see.

**Observation 5.16** *All the* $\mathbf{y}$*-variables appearing in* $\widehat{q}_1(A_1'\mathbf{x}), \ldots, \widehat{q}_m(A_1'\mathbf{x})$ *are distinct. Also,* $A_1' \in \mathrm{GL}(n, \mathbb{F})$ *and acts as identity on variables not in* $\mathrm{var}(\widetilde{q}_1(A_1\mathbf{x})) \uplus \cdots \uplus \mathrm{var}(\widetilde{q}_m(A_1\mathbf{x}))$ *i.e. on* $\mathbf{z} \uplus \mathbf{u}$.

In Step 8, $A_1 := A_0A_1'$. For every $k \in \{s_1 + 1, \ldots, s'\}$ let $\mathbf{u}_k$ be an arbitrary subset of $\mathbf{u} \setminus \{u_0\}$ of size equal to the number of dangling variables in $T_k$ which are redundant for $\det(H_2)$. While defining $\mathbf{u}_k$s we ensure that for $k \neq k'$, $\mathbf{u}_k$ and $\mathbf{u}_{k'}$ are disjoint. Redefine $\mathbf{y}_k$ to be the union of the set of $\mathbf{y}$ variables appearing in the quadratic forms in $\widehat{T}_k(A_1\mathbf{x} + \mathbf{b})$ and $\mathbf{u}_k$. Then, by permuting the variables in $\mathbb{C}$ if necessary, we can assume that $\mathbf{y}_k \setminus \mathbf{u}_k$ are the $\mathbf{y}$ variables appearing in the quadratic forms in $T_k$ that are redundant for $\det(H_2)$ and $\mathbf{u}_k$ are the dangling variables in $T_k$ that are redundant for $\det(H_2)$. For all $x \in \mathbf{x}$, let $\ell^{(1)}_x$ be the linear part of the affine form that replaces $x$ in $f(A_1\mathbf{x} + \mathbf{b})$. That is, for all $x \in \mathbf{z} \uplus \mathbf{u}$, $\ell^{(1)}_x = \ell^{(0)}_x(A_1'\mathbf{x})$ is the linear form that $x$ is mapped to by $BA_1$, while for all $y \in \mathbf{y} \setminus \mathbf{u}$, if $y = y_{i,j,l}$, then $\ell^{(1)}_y = y'_{i,j,l} + h_{i,j,l}$.[1] Also by permuting the variables in $\mathbb{C}$ if required, we can assume that $y = y'_{i,j,l}$.

**Claim 5.16** $\left\{\left[\ell^{(1)}_u\right]_{\mathbf{u}} : u \in \mathbf{u}\right\}$ *is linearly independent.*

**Proof:** $\left\{\ell^{(1)}_x : x \in \mathbf{x}\right\}$ is linearly independent. As $A_1'$ acts as identity on $\mathbf{z} \uplus \mathbf{u}$ (Observation 5.16), from Claim 5.13, we get that $\ell^{(1)}_z \in \mathbb{F}[\mathbf{z}]$ for all $z \in \mathbf{z}'$. Hence, $\dim\left\langle \left[\ell^{(1)}_x\right]_{\mathbf{y}} : x \in \mathbf{z}'' \uplus \mathbf{y} \right\rangle = |\mathbf{y}|$. From Claim 5.9, no $y \in \mathbf{y} \setminus \mathbf{u}$ is in $\mathrm{var}(\det(H_2))$. Thus, by applying $A_1'$ on both sides of the equation in the second point of Claim 5.13, we get that for all $z \in \mathbf{z}''$, $\ell^{(1)}_z = \ell'_z + \sum_{u\in\mathbf{u}} \alpha_u \ell^{(1)}_u$, where $\ell'_z \in \mathbb{F}[\mathbf{z}]$. Hence, $\left\{\left[\ell^{(1)}_z\right]_{\mathbf{y}} : z \in \mathbf{z}''\right\} \in \mathbb{F}\text{-span}\left\{\left[\ell^{(1)}_y\right]_{\mathbf{y}} : y \in \mathbf{y}\right\}$;

[1] $y'_{i,j,l} + h_{i,j,l}$ need not necessarily be $\ell^{(0)}_y(A_1'\mathbf{x})$. This is so, because for any $i \in \{0, \ldots, m\}$, an invertible matrix mapping $q_i$ to $\widetilde{q}_i$ need not be unique.

so $\left\{\left[\ell_y^{(1)}\right]_{\mathbf{y}} : y \in \mathbf{y}\right\}$ is linearly independent. Now, $\left\{\left[\ell_y^{(1)}\right]_{\mathbf{y}} : y \in \mathbf{y} \setminus \mathbf{u}\right\} = \mathbf{y} \setminus \mathbf{u}$. Thus, $\left\{\left[\ell_u^{(1)}\right]_{\mathbf{u}} : u \in \mathbf{u}\right\}$ is linearly independent. □

Recall that in $f(A_0\mathbf{x} + \mathbf{b})$, $\mathbf{y}$-variables are only present in $\ell_x^{(0)}$ if $x$ the top dangling variable, a variable in the top quadratic form, a dangling variable along a skewed path or a variable in some quadratic form along a skewed path which is not truly essential for $\det(H_2)$. Also, $A_1'$ acts as identity on $\mathbf{z}$, and $\left[\ell_y^{(1)}\right]_{\mathbf{y}}$ is a single variable in $\mathbf{y} \setminus \mathbf{u}$ for all $y \in \mathbf{y} \setminus \mathbf{u}$. Hence, a $\mathbf{u}$-variable is only present in $\ell_x^{(1)}$ if $x$ is the top dangling variable (i.e. $u_0$) or a dangling variable along a skewed path which is not truly essential for $\det(H_2)$. So, if $u_1, \ldots, u_{m'}$ are dangling variables that are not truly essential for $\det(H_2)$ and $\mu_1', \ldots, \mu_m'$ are the corresponding skewed paths, then $\widehat{\ell}$ in Step 12 looks like $\left[\ell_{u_0}^{(1)}\right]_{\mathbf{u}} + \mu_1'\left[\ell_{u_1}^{(1)}\right]_{\mathbf{u}} + \cdots + \mu_{m'}'\left[\ell_{u_{m'}}^{(1)}\right]_{\mathbf{u}}$. Because $m' \leq n$, $\widehat{\ell}$ is $n^2$-sparse and can be interpolated efficiently. If $\mathcal{B} = \left\{\widehat{\ell}_{i_1}, \ldots, \widehat{\ell}_{i_m}\right\}$ is a basis of $\left\langle \widehat{\ell}_1, \ldots, \widehat{\ell}_{m'} \right\rangle = \left\langle \left[\ell_{u_0}^{(1)}\right]_{\mathbf{u}}, \ldots, \left[\ell_{u_{m'}}^{(1)}\right]_{\mathbf{u}} \right\rangle$, then it is clearly possible to map the linear forms $\widehat{\ell}_{i_1}, \ldots, \widehat{\ell}_{i_m}$ to distinct $\mathbf{u}$-variables. Claims 5.13 and 5.9 imply that

$$\left\langle \left[\ell_x^{(1)}\right]_{\mathbf{u}} : x \in \mathbf{z}_k'' \uplus (\mathbf{u}_k \cap \mathrm{var}(h_k')) \right\rangle = \left\langle \left[\ell_u^{(1)}\right]_{\mathbf{u}} : u \in \mathbf{u}_k \cap \mathrm{var}(h_k') \right\rangle$$

for every $k \in \{s_1 + 1, \ldots, s_2\}$. Thus, Claim 5.16 implies the following observation.

**Observation 5.17** *For any $u \notin \mathrm{var}(\det(H_2))$, $\left[\ell_u^{(1)}\right]_{\mathbf{u}} \in \mathcal{B}$. Also, for any $k \in \{s_1 + 1, \ldots, s_2\}$, $\mathcal{B}$ contains $|\mathbf{u}_k \cap \mathrm{var}(h_k')|$ many linear forms from $\left\{\left[\ell_u^{(1)}\right]_{\mathbf{u}} : u \in \mathbf{u}_k \cap \mathrm{var}(h_k')\right\}$.*

In particular, if $\mathtt{C}$ has a top dangling variable, then $\widehat{\ell}_0 = \left[\ell_{u_0}^{(1)}\right]_{\mathbf{u}} \in \mathcal{B}$ and it is mapped to $u_0$ in Step 13. So, after $A_2$ is set to $A_1A_2'$, $\ell(A_2\mathbf{x} + \mathbf{b})$, i.e. the top linear form in $f(A_2\mathbf{x} + \mathbf{b})$ contains $u_0$. Also, because $A_1'$ acts as identity on $\mathbf{z} \uplus \mathbf{u}$ and $A_2'$ act as identity on $\mathbf{z} \uplus \mathbf{y}$, $A_1'A_2'$ is identity on $\mathbf{z}$. For every $k \in \{s_1 + 1, \ldots, s_2\}$, let $\mathbf{x}_k' \subseteq \mathbf{z}_k'' \uplus \mathbf{u}_k$, $|\mathbf{x}_k'| = |\mathbf{u}_k|$ be such that $\left\{\left[\ell_x^{(1)}\right]_{\mathbf{u}} : x \in \mathbf{x}_k'\right\} \subseteq \mathcal{B}$. Let $\mathbf{x}' = \uplus_{s_1+1 \leq k \leq s_2} \mathbf{x}_k' \uplus \{u_0\}$. Then the following observation is easy to see.

**Observation 5.18** *For all $x \in \mathbf{x}'$, $\ell_x^{(1)}(A_2'\mathbf{x})$ looks like $u + h_u'$, for some $u \in \mathbf{u}$ and $h_u' \in \mathbb{F}[\mathbf{z}, \mathbf{y} \setminus \mathbf{u}]$. Also, if the skewed path corresponding to $x$ is $\mu$, then $(\mu, u) \in V$.*

We now show that $\mathbf{x}'$ is in a set of redundant variables for $\det(H_2)$.

**Claim 5.17** *$\mathbf{x}' \uplus (\mathbf{y} \setminus \mathbf{u})$ is a set of redundant variables for $\det(H_2)$.*

**Proof:** Immediately after Step 9 of Algorithm 4 is executed, $\det(H_2)(BRA_0\mathbf{x}+\mathbf{d}) = h(A_0\mathbf{x}) \in \mathbb{F}[\mathbf{z}]$. Because $C$ computed in the for loop of lines 12-14 only maps some variables in $\mathbf{z}'$ to linear forms in $\mathbb{F}[\mathbf{z}]$, after this loop is executed, $\det(H_2)(BRA_0(C\mathbf{x}+\mathbf{b})+\mathbf{d}) \in \mathbb{F}[\mathbf{z}]$. Thus, after $A_0$ is updated to be $RA_0C$ and $\mathbf{b} := RA_0\mathbf{b}'$ in Step 15 of Algorithm 4, $\det(H_2)(BA_0\mathbf{x}+B\mathbf{b}+\mathbf{d}) \in \mathbb{F}[\mathbf{z}]$. Since $A_1'$ acts as identity on $\mathbf{z}$ and $A_1 = A_0A_1'$, $h' := \det(H_2)(BA_1\mathbf{x}+B\mathbf{b}+\mathbf{d}) \in \mathbb{F}[\mathbf{z}]$. Now, from the chain rule of derivatives we have that

$$\nabla h' = (BA_1)^T \left[\nabla \det(H_2)\right](BA_1\mathbf{x}+B\mathbf{b}+\mathbf{d}),$$

where $\nabla h'$ and $\nabla \det(H_2)$ are gradients of $h'$ and $\det(H_2)$ with respect to $\mathbf{x}$, respectively. As $h'$ does not contain any $\mathbf{u}$-variable,

$$\mathbf{0} = \left[\nabla h'\right]_{\mathbf{u}} = [(BA_1)^T]_{\mathbf{u}}\left[\nabla \det(H_2)\right](BA_1\mathbf{x}+B\mathbf{b}+\mathbf{d}),$$

where $[\nabla h']_{\mathbf{u}}$ is $\nabla h'$ restricted to entries corresponding to $\mathbf{u}$ and $[(BA_1)^T]_{\mathbf{u}}$ is $(BA_1)^T$ restricted to rows corresponding to $\mathbf{u}$. Thus,

$$[(BA_1)^T]_{\mathbf{u},\mathbf{x}'}\left[\nabla \det(H_2)\right]_{\mathbf{x}'} = -[(BA_1)^T]_{\mathbf{u},\mathbf{x}\setminus\mathbf{x}'}\left[\nabla \det(H_2)\right]_{\mathbf{x}\setminus\mathbf{x}'},$$

where $[(BA_1)^T]_{\mathbf{u},\mathbf{x}'}$ and $[(BA_1)^T]_{\mathbf{u},\mathbf{x}\setminus\mathbf{x}'}$ are the sub-matrices of $(BA_1)^T$ whose rows and columns are labelled by variables in $\mathbf{u}$, $\mathbf{x}'$ and $\mathbf{u}$, $\mathbf{x}\setminus\mathbf{x}'$ variables respectively. As $\mathcal{B} = [BA_1]_{\mathbf{x}',\mathbf{u}}\mathbf{u}$, $[(BA_1)^T]_{\mathbf{u},\mathbf{x}'}$ is invertible. By right multiplying its inverse on both sides of the above equation, we get that for all $x' \in \mathbf{x}'$, $\frac{\partial}{\partial x'}\det(H_2) \in \mathbb{F}\text{-span}\left\{\frac{\partial}{\partial x}\det(H_2) : x \in \text{var}(\det(H_2))\setminus\mathbf{x}'\right\}$. Hence, $\mathbf{x}'$ is redundant for $\det(H_2)$. Then, as no variable $\mathbf{y}\setminus\mathbf{u}$ is present in $\det(H_2)$, the claim follows. □

Using an argument similar to the one used to prove Claim 5.13, we can prove the following observation.

**Observation 5.19** *For all $k \in \{s_1+1,\ldots,s_2\}$ and all $x \in \mathbf{z}_k'' \uplus (\mathbf{u}_k \cap \text{var}(h_k'))$, $\ell_x^{(1)} = \ell_x'' + \sum_{\substack{x'\in\mathbf{x}'\cap\\ \text{var}(h_k')}} \alpha_{x'}'\ell_{x'}^{(1)}$, where $\ell_x'' \in \mathbb{F}[\mathbf{z}_k]$ and $\alpha_{x'}' \in \mathbb{F}$ for all $x' \in \mathbf{x}_k' \cap \text{var}(h_k')$.*

For all $x \in \mathbf{x}$, let $\ell_x^{(2)} = \ell_x^{(1)}(A_2'\mathbf{x})$ be the linear part of the affine from that replaces $x$ in $f(A_2\mathbf{x}+\mathbf{b})$. Because $A_1'A_2'$ acts as identity on $\mathbf{z}_k$, Observations 5.17 and 5.19 imply that if for all $k \in \{s_1+1,\ldots,s_2\}$ and $x \in \mathbf{x}_k'$, we can remove variables not in $\mathbf{z}_k \uplus \mathbf{y}_k$ - i.e. "external" variables - from $\ell_x^{(2)}$, then all linear forms corresponding to dangling variables

along skewed paths in $T_k$ would just be in $\mathbf{z}_k \uplus \mathbf{y}_k$ variables. Similarly, because $A'_2$ acts as identity on $\mathbf{y}$, Claim 5.15 implies that if we can remove variables not in $\mathbf{z}_k$ from $\ell_y^{(2)}$ for all $y \in \mathbf{y}_k \setminus \mathbf{u}_k$, then all linear forms corresponding to variables appearing in quadratic forms along skewed paths in $T_k$ would just be in $\mathbf{z}_k \uplus \mathbf{y}_k$ variables. Then all the non-linear terms of $f(A_2\mathbf{x} + \mathbf{b})$ would become variable disjoint. We now describe the Remove-External-Vars() algorithm and show that it does just this.

---

**Algorithm 6** Remove-External-Vars($f(\mathbf{x}), A_2, \mathbf{b}, \mathbf{z}, \mathbf{y}, \mathbf{u}, u_0, V$)

---

**Input.** $f(\mathbf{x})$, $A_2$, $\mathbf{b}$, $\mathbf{z}$, $\mathbf{y}$, $\mathbf{u}$, $u_0$, and $V$ are as in Step 14 of Algorithm 5.

**Output.** $A \in \mathrm{GL}(n, \mathbb{F})$ such that all the non-linear terms of $f(A\mathbf{x} + \mathbf{b})$ are variable disjoint.

/* Removing external $\mathbf{z}$-variables from quadratic forms and external $\mathbf{y}$-variables from linear forms. */

$A'_3 \leftarrow I_{n\times n}$.

**for** $y \in \mathbf{y} \setminus \mathbf{u}$ **do**

  Interpolate $g \leftarrow \frac{\partial f(A_2\mathbf{x}+\mathbf{b})}{\partial y}$. If $\mu$ is the $\mathbf{z}$-monomial multiplied to the only $\mathbf{y}$-variable, say $y'$, in $g$, then write $g = \mu(y' + \ell'_{y'} + \alpha_{y'}) + r(\mathbf{z})$, where $\ell'_{y'} \in \mathbb{F}[\mathbf{z}]$ is a linear form, $\alpha_{y'} \in \mathbb{F}$, and $r(\mathbf{z}) \in \mathbb{F}[\mathbf{z}]$.

  Update $A'_3$ so that it maps $y'$ to $y' - \ell'_{y'}$.

  **for** every monomial $\mu'$ in $r(\mathbf{z})$ **do**

    If there exists a $u'$ such that $(\mu', u') \in V$, update $A'_3$ to map $u'$ to $u' - \beta y$, where $\beta$ is the coefficient of $\mu'$ in $r(\mathbf{z})$.

  **end for**

**end for**

$A_3 \leftarrow A_2 A'_3$.

/* Removing external $\mathbf{z}$ variables from linear forms. */

$A'_4 \leftarrow I_{n\times n}$. $F \leftarrow$ a subset of $\mathbb{F}$ of size $n^5$.

**for** $(\mu, u) \in V$ such that $\deg(\mu) \geq 2$ **do**

  **for** $z \in \mathbf{z}$ **do**

    $g \leftarrow f(A_3(\mathrm{var}(\mu),\ z,\ \mathbf{x} \setminus (\mathrm{var}(\mu) \uplus \{z\}) = \mathbf{0}) + \mathbf{b})$.

    Interpolate $\frac{\partial g}{\partial z}$. Let $\alpha$ be the coefficient of $\mu$ in $\frac{\partial g}{\partial z}$. Update $A'_4$ to map $u$ to $u - \alpha z$.

  **end for**

**end for**

**for** $(\mu, u) \in V$ such that $\deg(\mu) = 1$ **do**

21: **for** $z \in \mathbf{z}$ **do**
22: Set all variables in $f(A_3A_4'\mathbf{x}+\mathbf{b})$ other then $\mathrm{var}(\mu)$ and $z$ to random elements from $F$. $g \leftarrow$ the resulting polynomial.
23: Interpolate $\frac{\partial g}{\partial z}$. Let $\alpha$ be the coefficient of $\mu$ in $\frac{\partial g}{\partial z}$. Update $A_4'$ to map $u$ to $u - \alpha z$.
24: **end for**
25: **end for**
26: $A \leftarrow A_3A_4'$. Return $A$.

---

We first consider the for loop of lines 3-9 and show that the matrix $A_3'$ computed by this loop is such that it removes all variables not in $\mathbf{z}_k \uplus \mathbf{y}_k$ from $\ell_y^{(2)}$ for all $y \in \mathbf{y}_k$ and $k \in \{s_1+1,\ldots,s'\}$, and removes all variables in $\mathbf{y} \setminus \mathbf{y}_k$ from $\ell_x^{(2)}$ for all $x \in \mathbf{x}_k'$ and $k \in \{s_1+1,\ldots,s_2\}$. Before arguing this, we remark that in any iteration of this loop, $g$ is a $2n$-sparse polynomial. This is so, because any $y \in \mathbf{y}$ is only present in $\ell_y^{(2)}$ and in $\ell_x^{(2)}$ for $x \in \mathbf{x}'$. Thus, every monomial in $r(\mathbf{z})$ is a skewed path and there are at most $n$ skewed paths.

**Claim 5.18** *The matrix $A_3'$ computed after the execution of the for loop of lines 3-9 is such that for every $k \in \{s_1+1,\ldots,s_2\}$ and every $y' \in \mathbf{y}_k \setminus \mathbf{u}_k$, $\ell_{y'}^{(2)}(A_3'\mathbf{x}) \in \mathbb{F}[\mathbf{z}_k \uplus \mathbf{y}_k]$.*

**Proof:** Pick any arbitrary $k \in \{s_1+1,\ldots,s_2\}$ and $y' \in \mathbf{y}_k$. If $\ell_{y'}^{(2)}$ contains a variable not in $\mathbf{z}_k \uplus \mathbf{y}_k$, because of Claim 5.15 and the fact that $A_2'$ acts as identity on $\mathbf{z} \uplus (\mathbf{y} \setminus \mathbf{u})$, it must be in $\mathbf{z} \setminus \mathbf{z}_k$. Suppose that $yy'$ is a term in the quadratic form along a skewed $\mu$ in $T_k$. Fix any $z \in \mathbf{z} \setminus \mathbf{z}_k$; if $z \in \mathrm{var}\left(\ell_{y'}^{(2)}\right)$, then during the $y$-th iteration of the for loop of lines 3-9, $g$ contains the monomial $\mu z$. We now argue that the only place in $f(A_2\mathbf{x}+\mathbf{b})$ which can contribute $\mu z$ to $g$ is $\mu\ell_{y'}^{(2)}$. This implies that the coefficient of $\mu z$ in $g$, i.e., the coefficient of $z$ in $\ell_{y'}'$ after Step 4 is equal to its coefficient in $\ell_{y'}^{(2)}$.

For $\mu z$ to be present in $g$, $\mu zy$ must be present in $f(A_2\mathbf{x}+\mathbf{b})$. We claim that $\mu zy$ is not present in $\widehat{T}_{k'}(A_2\mathbf{x}+\mathbf{b})$ for any $k' \neq k$. If $k' \in [s_1]$, then this directly follows from Claim 5.13 and the fact that $A_1'A_2'$ act as identity on $\mathbf{z}_k$. For a $k' \in \{s_1+1,\ldots,s\}$, note that as $k \in \{s_1+1,\ldots,s_2\}$, $\deg(\mu) \geq 1$. Because $y$ and variables in $\mathrm{var}(\mu)$ are not in $\mathbf{z}_{k'} \uplus \mathbf{y}_{k'}$, $y$ can only be present in $\ell_x^{(2)}$ for some $x \in \mathbf{x}_{k'}$ if $x$ is a dangling variable along some skewed path. Hence any monomial of $\widehat{T}_{k'}(A_2\mathbf{x}+\mathbf{b})$ containing $y$ can not contain any other variable in $\mathbf{z}_k \uplus \mathbf{y}_k$. So, $\mu yz$ is not present in $\widehat{T}_{k'}(A_2\mathbf{x}+\mathbf{b})$.

Now, apart from $\ell_y^{(2)}$, $y$ can only appear in $\ell_x^{(2)}$ for some $x \in \mathbf{x}_k$, if $x$ is a dangling variable along some skewed path, say $\mu'$. However, since $z \notin \mathbf{z}_k$, $z \notin \mathrm{var}(\mu')$. So, $\mu zy$ can not be

present in $\mu' \ell_x^{(2)}$. This only leaves $\mu \ell_{y'}^{(2)}$ as the place that can contribute $\mu z$. Hence the coefficient of $z$ in $\ell'_{y'}$ is equal to its coefficient in $\ell_{y_1}^{(2)}$. Now, notice that $y'$ is not present in $g$ in any iteration of the for loop other than the $y$-th iteration. Hence, through out the execution of the loop, $A'_3$ only acts on $y'$ during the $y$-th iteration. In this iteration, after Step 5 is executed, $\ell_{y'}^{(2)}(A'_3\mathbf{x} + \mathbf{b}')$ does not contain $z$ as $A'_3$ is updated to map $y'$ to $y' - \ell'_{y'}$. Since this is true for any $z \in \mathbf{z} \setminus \mathbf{z}_k$, the claim follows. □

To show that $BA_2A'_3$ maps the top quadratic form to $\sum_{k=s_2+1}^{s'}(y_{k,1} + c_{k,1})(y_{k,2} + c_{k,2})$ and $A'_3$ removes external $\mathbf{y}$ variables from $\ell_{u_0}^{(2)}$, we shall use the following two observations.

**Observation 5.20** *For any $y \in \uplus_{k'=s_2+1}^{s'} \mathbf{y}_{k'}$ and any $k \in \{s_1+1, \ldots, s_2\}$, if $T_k = zQ$ and the top dangling variable of $Q$ is $x$, then it can be assumed without loss of generality that $y$ is not present in $\ell_x^{(2)}$.*

**Proof:** Suppose that $yy'$ is a term in $\sum_{k'=s_2+1}^{s'} T_{k'}$ and that the coefficient of $y$ in $\ell_x^{(2)}$ is $\beta$. Then we 'absorb' $\beta z$ in $\ell_{y'}^{(2)}$ and subtract $\beta\left(\ell_y^{(2)} + c - y\right)$ from $\ell_x^{(2)}$; here $c$ is the constant term of the affine form that replaces $y$ in $f(A_2\mathbf{x} + \mathbf{b})$. This does not change $f(A_2\mathbf{x} + \mathbf{b})$. □

**Observation 5.21** *If $\mathsf{C}$ has a top dangling variable, then for any $y \in \uplus_{k=s_2+1}^{s'} \mathbf{y}_k$, it can be assumed without loss of generality that the affine form replacing $y$ in $f(A_2\mathbf{x} + \mathbf{b})$ has no constant.*

**Proof:** Suppose that $T_k = y_1y_2$ for some $k \in \{s_2+1, \ldots, s'\}$ and that $\widehat{T}_k(A_2\mathbf{x} + \mathbf{b}) = (\ell_{y_1}^{(2)} + c_1)(\ell_{y_2}^{(2)} + c_2)$, where $c_1, c_2 \in \mathbb{F}$. Then we add $c_2\ell_{y_1}^{(2)} + c_1\ell_{y_2}^{(2)}$ to $\ell_{u_0}^{(2)}$ and add $c_1c_2$ to the constant of the affine form replacing $u_0$ in $f(A_2\mathbf{x} + \mathbf{b})$. This does not change $f(A_2\mathbf{x} + \mathbf{b})$. □

We call every $x \in \mathbf{x}'$ such that some $T_k = zQ$ and $x$ is the top dangling variable of $Q$, a *bad dangling variable*. For every $y \in \uplus_{k'=s_2+1}^{s'} \mathbf{y}_{k'}$, every bad dangling variable $x$, and $u_0$ we redefine $\ell_y^{(2)}$, $\ell_x^{(2)}$ and $\ell_{u_0}^{(2)}$ as mentioned in the proofs of the above observations.

**Claim 5.19** *The matrix $A'_3$ computed after the execution of the for loop of lines 3-9 is such that for every $y' \in \uplus_{s_2+1 \le k \le s'} \mathbf{y}_k$, $\ell_{y'}^{(2)}(A'_3\mathbf{x}) = y' + c$ for some $c \in \mathbb{F}$.*

**Proof:** Pick any arbitrary $k \in \{s_2+1, \ldots, s'\}$ and $y' \in \mathbf{y}_k$. If $\ell_{y'}^{(2)}$ contains a variable not in $\mathbf{y}_k$ (as $\mathbf{z}_k = \emptyset$), because of Claim 5.15 and the fact that $A'_2$ acts as identity on $\mathbf{z} \uplus (\mathbf{y} \setminus \mathbf{u})$, it must be in $\mathbf{z}$. Suppose that $T_k = yy'$. Fix any $z \in \mathbf{z}$; if $z \in \text{var}\left(\ell_{y'}^{(2)}\right)$, then during the $y$-th iteration of the for loop of lines 3-9, $g$ contains $z$. We now show that the only place in $f(A_2\mathbf{x} + \mathbf{b})$ that can contribute $z$ to $g$ is $\ell_{y'}^{(2)}$. Observe that for $z$ to be present in $g$, $yz$ must be present

in $f(A_2\mathbf{x}+\mathbf{b})$. Apart from $\ell_y^{(2)}$, $y$ is only present in $\ell_x^{(2)}$ if $x$ is a dangling variable along a skewed path in some bad term $T_{k'}$ or $x = u_0$. However, Observation 5.20 implies that we can assume without loss of generality that the only place in $f(A_2\mathbf{x}+\mathbf{b})$ that contains $zy$ is $\ell_y^{(2)}\ell_{y'}^{(2)}$. Thus, after Step 5 is executed, $y'$ is mapped to an affine form in $y'$ in $f(A_2A_3'\mathbf{x}+\mathbf{b})$. □

Claims 5.18 and 5.19 ensure that external variables are removed from the linear forms corresponding to variables appearing in quadratic forms along skewed paths and the top quadratic form. The following claim proves that external $\mathbf{y}$ variables are removed from linear forms corresponding to dangling variables along skewed paths and the top dangling variable.

**Claim 5.20** *The matrix $A_3'$ computed after the execution of the for loop of lines 3-9 is such that for every $k \in \{s_1+1,\ldots,s_2\}$ and $x \in \mathbf{x}'_k$, $\ell_x^{(2)}(A_3'\mathbf{x})$ does not contain any variable from $\mathbf{y}\setminus\mathbf{y}_k$. Also if $\mathsf{C}$ has a top dangling variable, then $\ell_{u_0}^{(2)}(A_3'\mathbf{x})$ does not contain any $\mathbf{y}$ variable other than $u_0$.*

**Proof:** Fix a $k \in \{s_1+1,\ldots,s_2\}$ and an $x \in \mathbf{x}'_k$. Suppose that $y \in \mathbf{y}\setminus\mathbf{y}_k$ is present in $\ell_x^{(2)}$. Then, in the $y$-th iteration of the for loop of lines 3-9, the monomial $\mu'$ representing the skewed path corresponding to $x$ is in $r(\mathbf{z})$. Observe that $\deg(\mu') \geq 1$. Note that the only time $A_3'$ translates the sole $\mathbf{u}$-variable $u'$ in $\ell_x^{(2)}$ by a multiple of $y$ is in the $y$-th iteration of the loop. So it suffices to prove that in this iteration, $\beta$ in Step 7 is the coefficient of $y$ in $\ell_x^{(2)}$. We do this by showing that the only place in $f(A_2\mathbf{x}+\mathbf{b})$ from which $\mu'$ can appear in $g$ is from $\mu'\ell_x^{(2)}$.

For $\mu'$ to be present in $g$, $\mu'y$ must be present in $f(A_2\mathbf{x}+\mathbf{b})$. We claim that $\mu'y$ can not be present in $\widehat{T}_{k'}(A_2\mathbf{x}+\mathbf{b})$ for any $k' \neq k$. Because of Claims 5.18 and 5.19, variables in $\mathrm{var}(\mu')$ can only be present in $\ell_{x'}^{(2)}$ for some $x' \in \mathbf{x}_{k'}$ if $x'$ is a dangling variable along some skewed path in $T_{k'}$ or $x' = u_0$. Hence any monomial of $\widehat{T}_{k'}(A_2\mathbf{x}+\mathbf{b})$ containing a variable in $\mathrm{var}(\mu')$ can not contain any other variable in $\mathbf{z}_k \uplus \mathbf{y}_k$. So, $\mu'y$ is not present in $\widehat{T}_{k'}(A_2\mathbf{x}+\mathbf{b})$.

Now, in $\widehat{T}_k(A_2\mathbf{x}+\mathbf{b})$, $y$ is only present in $\ell_{x'}^{(2)}$ for some $x' \in \mathbf{x}_k$ if $x'$ is a dangling variable along some skewed path. However, if that skewed path is $\mu''$, then the monomial present in $\widehat{T}_k(A_2\mathbf{x}+\mathbf{b})$ is $\mu''y$. Hence $\mu''\ell_{x'}^{(2)}$ can contain the monomial $\mu'y$ only if $\mu'' = \mu'$ and $x' = x$. Thus only $\mu'\ell_x^{(2)}$ contributes $\mu'$ to $g$, and $\beta$ is precisely the coefficient of $y$ in $\ell_x^{(2)}$. Hence, after $A_3'$ has been updated to map $u'$ to $u' - \beta y$ in Step 7, $\ell_x^{(2)}(A_3'\mathbf{x})$ does not contain $y$.

For any $y \in \ell_{u_0}^{(2)}$, in the $y$-th iteration of the loop, $r(\mathbf{z})$ contains a constant, say $\beta$. Because $u_0$ is only translated by a constant multiple of $y$ in the $y$-th iteration of the loop, it is sufficient to show that $\beta$ is the coefficient of $y$ in $\ell_{u_0}^{(2)}$. If $y \in \mathbf{y}_k$ for some $k \in \{s_1+1,\ldots,s_2\}$, then every monomial in $f(A_2\mathbf{x}+\mathbf{b})$ containing $y$ must also contain a skewed path. Observation 5.21

implies that $A'_3$ maps the top quadratic from to $\sum_{k=s_2+1}^{s'} y_{k,1}y_{k,2}$. Thus $\beta y$ is also not present in the top quadratic form. Hence $\beta$ is exactly the coefficient of $y$ in $\ell^{(2)}_{u_0}$. Also, in this case when Step 7 is executed, $u' = u_0$. So after $A'_3$ has been updated to map $u'$ to $u' - \beta y$, $\ell^{(2)}_{u_0}(A'_3\mathbf{x})$ does not contain $y$. □

After $A_3$ has been defined as $A_2A'_3$, let $\ell^{(3)}_x$ be the linear part of the affine form replacing $x$ in $f(A_3\mathbf{x}+\mathbf{b})$. Note that for all $x$ other than those in $\uplus_{s_2+1\leq k\leq s'}\mathbf{y}_k$, bad dangling variables, and $u_0$, $\ell^{(3)}_x = \ell^{(2)}_x(A'_3\mathbf{x})$. Now from Claim 5.18, for all $k \in \{s_1+1,\ldots,s_2\}$, the only $x \in \mathbf{x}_k$ for which $\ell^{(3)}_x$ contains variables not in $\mathbf{z}_k \uplus \mathbf{y}_k$ are dangling variables along skewed paths. Also, for all $x \in \mathbf{x}'_k$, Claim 5.20 implies that $\ell^{(3)}_x \in \mathbb{F}[\mathbf{z} \uplus \mathbf{y}_k]$. Now $A'_2A'_3$ acts as identity on $\mathbf{z}$ and Claim 5.10 implies that no bad dangling variable is in $\mathrm{var}(\det(H_2))$. Thus Observation 5.19 implies that for all $k \in \{s_1+1,\ldots,s_2\}$, $x \in \mathbf{z}''_k \uplus \mathbf{u}_k$, $\ell^{(3)}_x \in \mathbb{F}[\mathbf{z} \uplus \mathbf{y}_k]$.

**Claim 5.21** *The matrix $A'_4$ computed after the execution of the for loop of lines 14-19 is such that for every $k \in \{s_1+1,\ldots,s_2\}$, $x \in \mathbf{x}'_k$ is not a bad dangling variable, $\ell^{(3)}_x(A'_4\mathbf{x}) \in \mathbb{F}[\mathbf{z}_k \uplus \mathbf{y}_k]$.*

**Proof:** Fix any $k \in \{s_1+1,\ldots,s_2\}$ and $x \in \mathbf{x}'_k$ which is not a bad dangling variable. If the corresponding skewed path is $\mu$, then $\deg(\mu) \geq 2$. Also, from Observation 5.18, if the sole $\mathbf{u}$ variable in $\ell^{(3)}_u$ is $u$, then $(\mu, u) \in V$. We analyse the iteration of the for loop of lines 14-19 corresponding to $(\mu, u)$. Fix any $z \in \mathbf{z} \setminus \mathbf{z}_k$. We show that in the $z$-th iteration of the for loop of lines 15-18 the coefficient of $\mu z$ in $g$ is exactly the coefficient of $z$ in $\ell^{(3)}_x$.

As $\deg(\mu) \geq 2$, $\mu z$ is not present in $\widehat{T}_{k'}(A_3\mathbf{x}+\mathbf{b})$ for any $k' \neq k$. Also, in $\widehat{T}_k(A_3\mathbf{x}+\mathbf{b})$, $z$ is only present in $\ell^{(3)}_x$ for some $x \in \mathbf{x}_k$, if $x$ is a dangling variable along a skewed path. However, if the skewed path corresponding to $x$ is $\mu'$, then we get the monomial $\mu' z$ form $\mu'\ell^{(3)}_x$. This means that $\mu' = \mu$ and hence in $\widehat{T}_k(A_3\mathbf{x}+\mathbf{b})$, $\mu z$ is only obtained from $\mu\ell^{(3)}_x$. This means that in Step 17, $\alpha$ is precisely the coefficient of $z$ in $\ell^{(3)}_x$. Hence, after that step is executed and $A'_4$ updated to map $u$ to $u - \alpha z$, $\ell^{(3)}_x(A'_4\mathbf{x})$ does not contain $z$. Also, observe that the only monomials containing $z$ in $\widehat{T}_{k'}(A_3(\mathrm{var}(\mu'),\ z,\ \mathbf{x} \setminus (\mathrm{var}(\mu') \uplus \{z\}) = \mathbf{0}) + \mathbf{b})$, for $k' \neq k$ can be of degree at most 2. Further any monomial containing $z$ in $\widehat{T}_k(A_3(\mathrm{var}(\mu'),\ z,\ \mathbf{x} \setminus (\mathrm{var}(\mu') \uplus \{z\}) = \mathbf{0}) + \mathbf{b})$ must look like $\mu' z$, where $\mu'$ is a submonomial of $\mu$. Hence, $\frac{\partial g}{\partial z}$ is sparse and can be interpolated efficiently. □

The above claim immediately implies that for all $k \in \{s_1+1,\ldots,s_2\}$, $\ell^{(3)}_x(A'_4\mathbf{x}) \in \mathbb{F}[\mathbf{z}_k \uplus \mathbf{y}_k]$ for all $x \in \mathbf{z}''_k \uplus \mathbf{u}_k$ which is not a bad dangling variable. To prove an analogous statement for the bad dangling variables we need the following observation. Note that Claim 5.10 and Observation 5.17 imply that every bad dangling variable is in $\mathbf{x}'$.

**Observation 5.22** *Suppose that $x_1, \ldots, x_m$ are all the bad dangling variables, the corresponding skewed paths are $\mu_1, \ldots, \mu_m$, the sole $\mathbf{u}$ variables in $\ell^{(3)}_{x_1}, \ldots, \ell^{(3)}_{x_m}$ are $u_1, \ldots, u_m$, and the for loop of lines 20-25 processes $(\mu_1, u_1), \ldots, (\mu_m, u_m)$ in that order. Then, it can be assumed without loss of generality that for all $i \in [m]$ and all $j < i$, $\ell^{(3)}_{x_i}$ does not contain $z_j$.*

**Proof:** For all $i \in [m]$ and all $j < i$, let the coefficient of $z_j$ in $\ell^{(3)}_{x_i}$ be $\beta_{i,j}$. For all $j \in [m]$, we 'absorb' $\sum_{i=j+1}^{m} \beta_{i,j} z_i$ in $\ell^{(3)}_{x_j}$ and remove $\beta_{i,j} z_j$ from $\ell^{(3)}_{x_i}$ for all $i > j$. This does not change $f(A_3\mathbf{x} + \mathbf{b})$. □

For every bad dangling variable $x$, we redefine $\ell^{(3)}_x$ as mentioned in the proof of the above observation. The following claim shows that variables in $\mathbf{z} \setminus \mathbf{z}_k$ are removed from $\ell^{(3)}_x$ for every bad dangling variable $x \in \mathbf{x}'_k$ as well.

**Claim 5.22** *The matrix $A'_4$ computed after the execution of the for loop of lines 20-25 is such that for every $k \in \{s_1 + 1, \ldots, s_2\}$, and every bad dangling variable $x \in \mathbf{x}'_k$, $\ell^{(3)}_x(A'_4) \in \mathbb{F}[\mathbf{z}_k \uplus \mathbf{y}_k]$.*

**Proof:** We prove the claim by showing that the following loop invariant holds: The matrix $A'_4$ computed after the $i$-th iteration of the loop is such that for all $j \leq i$, if $x_j \in \mathbf{x}'_k$, then $\widehat{\ell}^{(3)}_{x_j}(A'_4\mathbf{x}) \in \mathbb{F}[\mathbf{z}_k \uplus \mathbf{y}_k]$. Suppose that the invariant is true before the execution of the $i$-th iteration of the loop; it is trivially true before the first iteration. Suppose that $x_i \in \mathbf{x}'_k$. First we consider the $z$-th iteration of the for loop of lines 21-24 for a $z \notin \mathbf{z}_k \cup \{z_1, \ldots, z_m\}$. Since for any $k \in [s]$, the only $x \in \mathbf{x}_k$ such that $\ell^{(3)}_x(A'_4\mathbf{x})$ contains variables not in $\mathbf{z}_k$ are $x \in \{u_0, x_1, \ldots, x_m\}$, the only place in $f(A_3A'_4\mathbf{x} + \mathbf{b})$ that can contribute $z_i z$ to $g$ is $z_i \cdot \ell^{(3)}_{x_i}(A'_4\mathbf{x})$. Hence, in Step 23 $\alpha$ is exactly the coefficient of $z$ in $\ell^{(3)}_{u_i}(A'_4\mathbf{x})$. Thus after that step is executed and $A'_4$ is updated to map the sole $\mathbf{u}$ variable $u_i$ in $\ell^{(3)}_{x_i}(A'_4\mathbf{x})$ to $u_i - \alpha z$, $\ell^{(3)}_{x_i}(A'_4\mathbf{x})$ does not contain $z$. For a $z \in \{z_1, \ldots, z_{i-i}\}$, the assumption that the loop invariant is true before the $i$-th iteration and Observation 5.22 imply that $z_i z$ is not a monomial in $g$ and hence $z$ is not present in $\frac{\partial g}{\partial z}$. On the other hand, for all $z \in \{z_{i+1}, \ldots, z_m\}$, Observation 5.22 implies that in Step 23 $\alpha$ is exactly the coefficient of $z$ in $\ell^{(3)}_{x_i}(A'_4\mathbf{x})$. Hence, after that step is executed and $A'_4$ updated to map the sole $\mathbf{u}$ variable $u_i$ in $\ell^{(3)}_{x_i}(A'_4\mathbf{x})$ to $u_i - \alpha z$, $\ell^{(3)}_{x_i}(A'_4\mathbf{x})$ does not contain $z$. Notice that this also implies that the monomial $z_i z$ is no longer present in $f(A'_4\mathbf{x} + \mathbf{b})$. Also, in the $i$-th iteration, $A'_4$ does not act on any variable other than $u_i$. Hence, the invariant is also true after the execution of this iteration. □

Let $\ell^{(4)}_x$ be the linear part of the affine form replacing $x$ in $f(A\mathbf{x} + \mathbf{b})$. For all $k \in [s']$ and $x \in \mathbf{x}_k$, $\ell^{(4)}_x$ is now a linear form in $\mathbf{z}_k \uplus \mathbf{y}_k = \mathbf{x}_k$. Also, Claim 5.19 and the fact that $A'_4$ acts as

identity on $\mathbf{y}$ implies that $\sum_{k\in s_2+1}^{s'} \widehat{T}_k(A\mathbf{x}+\mathbf{b}) = \sum_{k\in s_2+1}^{s'} (y_{k,1}+c_{k,1})(y_{k,2}+c_{k,2})$. Now, as seen in the proof of Lemma 5.3, for every $x \in \mathbf{x}$ connected to a $\times$ gate computing a polynomial of degree at least three, $\ell_x^{(1)}$ is a constant multiple of a $\mathbf{z}$-variable. As $A_1' \cdots A_4'$ acts as identity on $\mathbf{z}$, so is $\ell_x^{(4)}$. Moreover, if $\mathtt{C}$ has a top dangling variable $u_0 = x_n$, then from Claim 5.20, the only $\mathbf{u}$ variable in $\ell_{u_0}^{(2)}$ was $u_0$. As $A_3'$ merely translates $u_0$ by constant multiples of $\mathbf{y}$-variables and $A_4'$ acts as identity on $u_0$, the only $\mathbf{u}$ variable in $\ell_{u_0}^{(4)}$ is $u_0$. Also, $u_0 \notin \text{var}\left(\ell_x^{(4)}\right)$ for any $x \neq u_0$. □

#### 5.5.3.3 Discovering the top linear form

We begin by stating the following useful claim.

**Claim 5.23 (Learning variable sets)** *After Step 14 of Algorithm 3 is executed,* $\mathbf{z}_k = \text{var}\left(\widehat{T}_k(A\mathbf{x}+\mathbf{b})\right)$ *for all* $k \in [s_2]$.[1]

**Proof:** There are two cases.

**Case 1:** $k \in [s_1]$, say $T_k = Q_{k,1}\cdots Q_{k,m_k}$, $m_k \geq 2$ or neither $Q_{k,1}$ nor $Q_{k,2}$ is linear, $\widehat{T}_k = \widehat{Q}_{k,1}\cdots\widehat{Q}_{k,m_k}$, and $\widehat{Q}_{k,l} = Q_{k,l}(B\mathbf{x}+\mathbf{d})$ for all $l \in [m_k]$. Then from Claim 5.7,

$$\widehat{Q}_{k,1}(RA_0\mathbf{x}), \ldots, \widehat{Q}_{k,m_k}(RA_0\mathbf{x})$$

are irreducible factors of $h(A_0\mathbf{x})$ where $R$, $A_0$, and $h$ are as just before Step 9 of Algorithm 4 is executed. Because $\widehat{\mathbf{z}}_1, \ldots, \widehat{\mathbf{z}}_m$ are returned by Make-Factors-Var-Disjoint($h(A_0\mathbf{x})$), from Claim 5.3, for every $l \in [m_k]$, there exists an $i \in [m]$ such that $\text{var}\left(\widehat{Q}_{k,l}(RA_0\mathbf{x})\right) \subseteq \widehat{\mathbf{z}}_i$, where $A_0$ is as after Step 9 of Algorithm 4 has been executed. It follows from Observation 2.8 that every variable in $\text{var}(Q_{k,l})$ is mapped to a linear from in $\widehat{\mathbf{z}}_i$ by $BRA_0$. As $C$ only maps some of these linear forms to constant multiples of variables in $\widehat{\mathbf{z}}_i$, even after $A_0$ is updated to be $RA_0C$ in Step 15 of Algorithm 4, $\text{var}\left(\widehat{Q}_{k,l}(A_0\mathbf{x}+\mathbf{b})\right) = \text{var}\left(\widehat{Q}_{k,l}(A_0\mathbf{x})\right) \subseteq \widehat{\mathbf{z}}_i$. Because in Algorithms 5 and 6, $A_1', \ldots, A_4'$ act as identity on $\mathbf{z}$, $\text{var}\left(\widehat{Q}_{k,l}(A\mathbf{x}+\mathbf{b})\right) \subseteq \widehat{\mathbf{z}}_i$ where $A$ and $\mathbf{b}$ are as after Step 4 of Algorithm 3.

If $\text{var}\left(\widehat{Q}_{k,1}(A\mathbf{x}+\mathbf{b})\right) \cup \cdots \cup \text{var}\left(\widehat{Q}_{k,m_k}(A\mathbf{x}+\mathbf{b})\right) \subseteq \widehat{\mathbf{z}}_i$, then $\text{var}\left(\widehat{T}_k(A\mathbf{x}+\mathbf{b})\right)$ is clearly contained in a single connected component of $G$. So suppose that there exist disjoint sets

[1] Here we are overloading the notation. Now $\mathbf{z}_k = \text{var}\left(\widehat{T}_k(A\mathbf{x}+\mathbf{b})\right)$. but in Sections 5.5.3.1 and 5.5.3.2 it was a set of essential variables of $\det(H_{T_k})$ evaluated at $BRA\mathbf{x}+\mathbf{d}$. The new $\mathbf{z}_k$ is the union of the old $\mathbf{z}_k$ and $\mathbf{z}_y$.

$I_1, I_2 \subseteq [m_k]$ and $i \neq j \in [m]$ such that $\cup_{l \in I_1} \text{var}\left(\widehat{Q}_{k,l}(A\mathbf{x}+\mathbf{b})\right) \subseteq \widehat{\mathbf{z}}_i$, $\cup_{l \in I_2} \text{var}\left(\widehat{Q}_{k,l}(A\mathbf{x}+\mathbf{b})\right) \subseteq \widehat{\mathbf{z}}_j$, and $\cup_{l \notin I_1 \uplus I_2} \text{var}\left(\widehat{Q}_{k,l}(A\mathbf{x}+\mathbf{b})\right) \cap (\widehat{\mathbf{z}}_i \uplus \widehat{\mathbf{z}}_j) = \emptyset$. Then, for any $z_1 \in \widehat{\mathbf{z}}_i$ and $z_2 \in \widehat{\mathbf{z}}_j$,

$$\begin{aligned}\frac{\partial^2 f(A\mathbf{x}+\mathbf{b})}{\partial z_1 \partial z_2} &= \frac{\partial^2 \widehat{T}_k(A\mathbf{x}+\mathbf{b})}{\partial z_1 \partial z_2} \\ &= \frac{\partial}{\partial z_1}\left(\prod_{l \in I_1} \widehat{Q}_{k,l}(A\mathbf{x}+\mathbf{b})\right) \cdot \frac{\partial}{\partial z_2}\left(\prod_{l \in I_2} \widehat{Q}_{k,l}(A\mathbf{x}+\mathbf{b})\right) \cdot \left(\prod_{l \notin I_1 \uplus I_2} \widehat{Q}_{k,l}(A\mathbf{x}+\mathbf{b})\right) \\ &\neq 0\end{aligned}$$

So the edge $\{\widehat{\mathbf{z}}_i, \widehat{\mathbf{z}}_j\}$ is added to $G$, and $\text{var}\left(\widehat{T}_k(A\mathbf{x}+\mathbf{b})\right)$ is in a single connected component of $G$.

**Case 2:** $k \in \{s_1+1, \ldots, s_2\}$, say $\widehat{T}_k = \widehat{Q}_{k,1}\widehat{Q}_{k,2}$, where $\widehat{Q}_{k,1}$ is in the orbit of a variable. If there exists an $i$ such that $\text{var}\left(\widehat{Q}_{k,1}(A\mathbf{x}+\mathbf{b})\right) \cup \text{var}\left(\widehat{Q}_{k,2}(A\mathbf{x}+\mathbf{b})\right) \subseteq \widehat{\mathbf{z}}_i$, then $\text{var}\left(\widehat{T}_k(A\mathbf{x}+\mathbf{b})\right)$ is clearly contained in a single connected component of $G$. Otherwise, it follows from Lemma 5.4 that $\widehat{Q}_{k,1}(A\mathbf{x}+\mathbf{b})$ is a constant multiple of a variable, say $z_1 \in \widehat{\mathbf{z}}_i$. Let $x$ be any variable in $\widehat{Q}_{k,2}(A\mathbf{x}+\mathbf{b})$. First suppose that $x \in \widehat{\mathbf{z}}_j$ for some $j \neq i$ and $x = z_2$. Then $\frac{\partial^2 f(A\mathbf{x}+\mathbf{b})}{\partial z_1 \partial z_2} = \frac{\partial^2 \widehat{T}_k(A\mathbf{x}+\mathbf{b})}{\partial z_1 \partial z_2} = \frac{\partial}{\partial z_1}\left(\widehat{Q}_{k,1}(A\mathbf{x}+\mathbf{b}) \cdot \frac{\partial \widehat{Q}_{k,2}(A\mathbf{x}+\mathbf{b})}{\partial z_2}\right)$. As, $\frac{\partial \widehat{Q}_{k,2}(A\mathbf{x}+\mathbf{b})}{\partial z_2} \neq 0$, $z_1 \in \text{var}\left(\widehat{Q}_{k,1}(A\mathbf{x}+\mathbf{b}) \cdot \frac{\partial \widehat{Q}_{k,2}(A\mathbf{x}+\mathbf{b})}{\partial z_2}\right)$. Thus, $\frac{\partial^2 f(A\mathbf{x}+\mathbf{b})}{\partial z_1 \partial z_2} \neq 0$ and the edge $\{\widehat{\mathbf{z}}_i, \widehat{\mathbf{z}}_j\}$ is added to $G$. Now, if $x \in \mathbf{y}$ and $x = y$, even then using the same argument as above, $\frac{\partial^2 f(A\mathbf{x}+\mathbf{b})}{\partial z_1 \partial y} \neq 0$ and the edge $\{\widehat{z}, y\}$ is added to $G$. Thus $\text{var}\left(\widehat{T}_k(A\mathbf{x}+\mathbf{b})\right)$ is contained in a single connected component of $G$.

So for all $k \in [s_2]$, $\text{var}\left(\widehat{T}_k(A\mathbf{x}+\mathbf{b})\right)$ is contained in a single connected component of $G$. Also, for $k \neq k' \in [s']$ and any $z_1 \in \widehat{T}_k(A\mathbf{x}+\mathbf{b})$, $z_2 \in \widehat{T}_{k'}(A\mathbf{x}+\mathbf{b})$, $\frac{\partial^2 f(A\mathbf{x}+\mathbf{b})}{\partial z_1 \partial z_2} = 0$. Thus, $\text{var}\left(\widehat{T}_k(A\mathbf{x}+\mathbf{b})\right)$ corresponds to a connected component in $G$. Further for any $y_1, y_2 \in \text{var}\left(\widehat{T}_{s_2+1}(A\mathbf{x}+\mathbf{b})\right) \uplus \ldots \uplus \text{var}\left(\widehat{T}_{s'}(A\mathbf{x}+\mathbf{b})\right)$ observe that $\frac{\partial^2 f(A\mathbf{x})}{\partial y_1 \partial y_2}$ is never computed, hence they are in distinct connected components of $G$ of size 1 each. $u_0$ is also in a connected component containing just itself. Hence the only connected components of $G$ with more then 1 variable correspond to $\text{var}\left(\widehat{T}_1(A\mathbf{x}+\mathbf{b})\right), \ldots, \text{var}\left(\widehat{T}_{s_2}(A\mathbf{x}+\mathbf{b})\right)$. □

Because of Step 17 of Algorithm 3, the following algorithm will only be called if $\mathsf{C}$ has a top dangling variable. It finds an affine form $\ell'$ such that when we map $u_0$ to $u_0 - \ell'$ in

$f(A\mathbf{x}+\mathbf{b})$, all its terms become variable disjoint and $\ell(A\mathbf{x}+\mathbf{b})$ becomes $u_0 + c$ for some $c \in \mathbb{F}$ (recall that $\ell$ is the affine form that the top dangling variable is mapped to by $B\mathbf{x}+\mathbf{d}$). This is done in $s_2$ iterations. In the $k$-th iteration it finds $\ell'$ restricted to $\mathbf{z}_k$ variables, denoted by $\ell_k$.

**Algorithm 7** Find-Top-Linear-Form($f'$)

**Input:** $f' = f(A\mathbf{x}+\mathbf{b})$, where $A$ and $\mathbf{b}$ as after Step 4 of Algorithm 3.

**Output:** An affine form $\ell'$ such that all terms in $f'\left(\mathbf{x} \setminus \{u_0\}, u_0 = u_0 - \ell'\right)$ are variable disjoint.

1: **for** $k \in [s_2]$ **do**

2: $\widehat{T} \leftarrow f'(\mathbf{z}_k, \mathbf{x} \setminus \mathbf{z}_k = \mathbf{0})$. $h' \leftarrow$ the Hessian determinant of $\widehat{T}$ with respect to $\mathbf{z}_k$-variables. $N \leftarrow$ the set of irreducible factors of $h'$. $F \leftarrow$ a subset of $\mathbb{F}$ of size at least $n^5$.

3: **for** $\widehat{Q} \in N$ **do**

4: **if** $\widehat{Q}$ is not linear **then**

5:

6: /* $\widehat{Q}$ is non-linear. */

7: $\mathbf{a}_1, \ldots, \mathbf{a}_{|\mathbf{z}_k|} \leftarrow$ vectors of size $|\mathbf{z}_k|$ containing random elements from $F$. $t \leftarrow$ a fresh variable.

8: $\forall i \in [|\mathbf{z}_k|]$, interpolate $\widehat{Q}(t\mathbf{a}_i)$ and $\widehat{T}(t\mathbf{a}_i)$. Discover $\widehat{Q}'_i(t)$ and $\beta_{i,0},\ \beta_{i,1} \in \mathbb{F}$ such that $\widehat{Q}(t\mathbf{a}_i) \cdot \widehat{Q}'_i(t) + \beta_{i,1} \cdot t + \beta_{i,0} = \widehat{T}(t\mathbf{a}_i)$ by solving a system of linear equations in the coefficients of $\widehat{Q}'_i(t)$ and $\beta_{i,0},\ \beta_{i,1}$.

9: Interpolate $\sum_{z \in \mathbf{z}_k} \alpha_z z$ using $\beta_{i,1}, \ldots, \beta_{|\mathbf{z}_k|,1}$. If $\widehat{T} - \sum_{z \in \mathbf{z}_k} \alpha_z z - \beta_{1,0}$ is reducible, $\ell_k \leftarrow \sum_{z \in \mathbf{z}_k} \alpha_z z - \beta_{1,0}$ and $\widehat{T} \leftarrow \widehat{T} - \ell_k$. Break.

10:

11: **else**

12: /* $\widehat{Q}$ is linear. */

13: Suppose $\widehat{Q} = z$; if not, move to the next iteration. $\mathbf{a}_i, \ldots, \mathbf{a}_{|\mathbf{z}_k|-1} \leftarrow$ vectors of size $|\mathbf{z}_k| - 1$ containing random elements from $F$. $t \leftarrow$ a fresh variable.

14: $\forall i \in [|\mathbf{z}_k| - 1]$, interpolate $\widehat{T}(z, \mathbf{z}_k \setminus \{z\} = t\mathbf{a}_i)$. Find $\widehat{Q}'_i(z,t)$ and $\beta_{i,0},\ \beta_{i,1},\ \beta_{i,2} \in \mathbb{F}$ such that $z \cdot \widehat{Q}'_i(z,t) + \beta_{i,2} \cdot z + \beta_{i,1} \cdot t + \beta_{i,0} = \widehat{T}(z, \mathbf{z}_k \setminus \{z\} = t\mathbf{a}_i)$ by solving a system of linear equations in the coefficients of $\widehat{Q}'_i(z,t)$ and $\beta_{i,0},\ \beta_{i,1},\ \beta_{i,2}$.

15: Interpolate $\sum_{z' \in \mathbf{z}_k \setminus \{z\}} \alpha_{z'} z'$ using $\beta_{i,1}, \ldots, \beta_{|\mathbf{z}_k|-1,1}$. If $\widehat{T} - \sum_{z' \in \mathbf{z}_k \setminus \{z\}} \alpha_{z'} z' - \beta_{1,0}$ is reducible, $\ell_k \leftarrow \sum_{z' \in \mathbf{z}_k \setminus \{z\}} \alpha_{z'} z' - \beta_{1,0}$ and $\widehat{T} \leftarrow \widehat{T} - \ell_k$. Break.

16: **end if**

17: **end for**

18: **end for**
19: $\ell' \leftarrow \sum_{k\in[s_2]} \ell_k$. Return $\ell'$.

---

We now prove the following lemma.

**Lemma 5.5 (Correctness of Algorithm 7)** *Find-Top-Linear-Form($f(A\mathbf{x}+\mathbf{b})$), where $A$ and $\mathbf{b}$ are as after Step 4 of Algorithm 3, finds an affine form $\ell'$ such that when $u_0$ is mapped to $u_0 - \ell'$ in $f(A\mathbf{x}+\mathbf{b})$, all its terms become variable disjoint and $\ell(A\mathbf{x}+\mathbf{b})$ becomes $u_0 + \alpha$ for some $\alpha \in \mathbb{F}$.*

**Proof:** Because of Step 17 of Algorithm 3, this algorithm will only be called if $\mathbb{C}$ has a top dangling variable. Recall that the variable sets of $\widehat{T}_1(A\mathbf{x}+\mathbf{b}), \ldots, \widehat{T}_{s_2}(A\mathbf{x}+\mathbf{b})$, and $\sum_{k=s_2+1}^{s'} \widehat{T}_k(A\mathbf{x}+\mathbf{b})$ are $\mathbf{z}_1, \ldots, \mathbf{z}_{s_2}$, and $\mathbf{y} \setminus \{u_0\}$ respectively. From Observation 5.18 the coefficient of $u_0$ in $\ell(A\mathbf{x}+\mathbf{b})$ is 1, and from Claim 5.20 no $y \in \mathbf{y} \setminus \{u_0\}$ appears in $\ell(A\mathbf{x}+\mathbf{b})$. Let $\ell(A\mathbf{x}+\mathbf{b}) = u_0 + \sum_{z\in\mathbf{z}} c_z \cdot z + c_0$; recall that $\mathbf{z} = \mathbf{z}_1 \uplus \cdots \uplus \mathbf{z}_{s_2}$. Fix a $k \in [s_2]$. We now show that in $k$-th iteration of the loop of lines 1-18 (the outer loop), the algorithm finds $\ell_k$ which is $\ell'$ restricted to $\mathbf{z}_k$ variables. Towards this, we first show that in the $k$-th iteration of the outer loop, $\widehat{T}$ is reducible after the execution of the for loop of lines 3-17 (the inner loop).

**Claim 5.24** *For any $k \in [s_2]$, after the execution of the inner loop during the $k$-th iteration of the outer loop, $\widehat{T}$ is reducible.*

**Proof:** At the beginning of the $k$-th iteration, $\widehat{T} = \widehat{T}_k(A\mathbf{x}+\mathbf{b}) + [\ell(A\mathbf{x}+\mathbf{b})]_{\mathbf{z}_k} + \gamma'$, for some $\gamma' \in \mathbb{F}$. In the algorithm $N$ is the set of irreducible factors of $h'$ which is the Hessian determinant of $\widehat{T}$ with respect to the $\mathbf{z}_k$-variables. Let $\widehat{T}_k = \widehat{Q}_{k,1} \cdots \widehat{Q}_{k,m_k}$. It follows from Corollary 5.1 and Fact 2.4, that at least one of the $\widehat{Q}_{k,1}(A\mathbf{x}+\mathbf{b}), \ldots, \widehat{Q}_{k,m_k}(A\mathbf{x}+\mathbf{b})$ is in an irreducible factor of $h'$. Hence, a constant multiple of at least one of the $\widehat{Q}_{k,1}(A\mathbf{x}+\mathbf{b}), \ldots, \widehat{Q}_{k,m_k}(A\mathbf{x}+\mathbf{b})$ is present in $N$ along with some other 'bad' factors. Fix a $\widehat{Q} \in N$. Then it is either a 'good' non-linear factor, 'good' linear factor, or a bad factor. In the following two claims we show that in the first two cases, $\widehat{T}$ is made reducible.

**Claim 5.25** *If $\widehat{Q}$ is a constant multiple of one of the $\widehat{Q}_{k,1}(A\mathbf{x}+\mathbf{b}), \ldots, \widehat{Q}_{k,m_k}(A\mathbf{x}+\mathbf{b})$ and is non-linear, then after the execution of the first inner loop, $\widehat{T}$ is reducible.*

**Proof:** Fix an $i \in [|\mathbf{z}_k|]$. As $\mathbf{a}_i$ is chosen randomly, $\deg(\widehat{Q}(t\mathbf{a}_i)) \geq 2$ with high probability. Notice that there exist $\widehat{Q}'_i, \beta_{i,0}, \beta_{i,1}$ such that $\widehat{Q}(t\mathbf{a}_i) \cdot \widehat{Q}'_i(t) + \beta_{i,1} \cdot t + \beta_{i,0} = \widehat{T}(t\mathbf{a}_i)$ is satisfied; one solution is $\widehat{Q}'_i = \left(\widehat{T}_k(A\mathbf{x}+\mathbf{b})/\widehat{Q}\right)(t\mathbf{a}_i)$, $\beta_{i,1} = (\sum_{z\in\mathbf{z}_k} c_z \cdot z)(\mathbf{a}_i)$, and $\beta_{i,0} = \gamma'$. We claim

that this solution is unique with high probability. Suppose that there existed two solutions $\widehat{Q}'_i, \beta_{i,0}, \beta_{i,1}$ and $\widehat{Q}''_i, \beta'_{i,0}, \beta'_{i,1}$. Then $\widehat{Q}(t\mathbf{a}_i) \cdot (\widehat{Q}'_i(t) - \widehat{Q}''_i(t)) = (\beta'_{i,1} - \beta_{i,1}) \cdot t + \beta'_{i,0} - \beta_{i,0}$. As $\deg(\widehat{Q}(t\mathbf{a}_i)) \geq 2$ with high probability, this is only possible if $\widehat{Q}'_i(t) = \widehat{Q}''_i(t)$, $\beta_{i,1} = \beta'_{i,1}$ and $\beta_{i,0} = \beta'_{i,1}$. In particular $\beta_{i,1} = (\sum_{z \in \mathbf{z}_k} c_z \cdot z)(\mathbf{a}_i)$ and $\beta_{i,0} = \gamma'$. Thus, after we interpolate $\sum_{z \in \mathbf{z}_k} \alpha_z z$ using $\beta_{i,1}, \ldots, \beta_{|\mathbf{z}_k|,1}$ and set $\ell_k = \sum_{z \in \mathbf{z}_k} \alpha_z z + \beta_{1,0}$, $\widehat{T} = \widehat{T}_k(A\mathbf{x} + \mathbf{b}) + [\ell(A\mathbf{x} + \mathbf{b})]_{\mathbf{z}_k} + \gamma' - \ell_k = \widehat{T}_k(A\mathbf{x} + \mathbf{b})$ is reducible. □

**Claim 5.26** *If $\widehat{Q}$ is a constant multiple of one of the $\widehat{Q}_{k,1}(A\mathbf{x} + \mathbf{b}), \ldots, \widehat{Q}_{k,m_k}(A\mathbf{x} + \mathbf{b})$ and is linear, after the execution of the first inner loop, $\widehat{T}$ is reducible.*

**Proof:** Fix an $i \in [|\mathbf{z}_k| - 1]$. Because $\deg(T_k) \geq 3$, Lemma 5.4 implies that $\widehat{Q}$ is a variable. Notice that there exist $\widehat{Q}'_i(z,t), \beta_{i,2}, \beta_{i,1}, \beta_{i,0}$ satisfying $z \cdot \widehat{Q}'_i(z,t) + \beta_{i,2} \cdot z + \beta_{i,1} \cdot t + \beta_{i,0} = \widehat{T}(z, \mathbf{z}_k \setminus \{z\} = \mathbf{a}_i \cdot t)$; one such solution is $\widehat{Q}'_i = \left(\widehat{T}_k(A\mathbf{x} + \mathbf{b})/\widehat{Q}\right)(z, t\mathbf{a}_i)$, $\beta_{i,2} = c_z$, $\beta_{i,1} = (\sum_{z' \in \mathbf{z}_k \setminus \{z\}} c_{z'} \cdot z')(\mathbf{a}_i)$ and $\beta_{i,0} = \gamma'$. We now show that $\beta_{i,1}$, and $\beta_{i,0}$ are unique with high probability. Suppose there are two solutions $\widehat{Q}'_i(z,t), \beta_{i,2}, \beta_{i,1}, \beta_{i,0}$ and $\widehat{Q}''_i(z,t), \beta'_{i,2}, \beta'_{i,1}, \beta'_{i,0}$. Then, $z \cdot (\widehat{Q}'_i(z,t) - \widehat{Q}''_i(z,t)) + (\beta_{i,2} - \beta'_{i,2}) \cdot z = (\beta'_{i,1} - \beta_{i,1}) \cdot t + (\beta'_{i,0} - \beta_{i,0})$. By putting $z = 0$ it can be seen that $\beta_{i,1} = \beta'_{i,1} = (\sum_{z' \in \mathbf{z}_k \setminus \{z\}} c_{z'} \cdot z')(\mathbf{a}_i)$ and $\beta_{i,0} = \beta'_{i,0} = \gamma'$. Thus, after we interpolate $\sum_{z' \in \mathbf{z}_k \setminus \{z\}} \alpha_{z'} z'$ using $\beta_{i,1}, \ldots, \beta_{|\mathbf{z}_k|,1}$ and set $\ell_k = \sum_{z' \in \mathbf{z}_k \setminus \{z\}} \alpha_{z'} z' + \beta_{1,0}$, $\widehat{T} = \widehat{T}_k(A\mathbf{x} + \mathbf{b}) + [\ell(A\mathbf{x} + \mathbf{b})]_{\mathbf{z}_k} + \gamma' - \ell_k = \widehat{T}_k(A\mathbf{x} + \mathbf{b})$ is reducible. □

Consider an iteration of the inner loop for a bad $\widehat{Q}$. If in this iteration $\widehat{T}$ is made reducible, then there is nothing to prove. Otherwise it follows from the above two observations that for all previous iterations of the inner loop, $\widehat{Q}$ must have been a bad factor. It follows from Corollary 5.1, that at least one of $\widehat{Q}_{k,1}(A\mathbf{x} + \mathbf{b}), \ldots, \widehat{Q}_{k,m_k}(A\mathbf{x} + \mathbf{b})$ is an irreducible factor of $h'$. This means that the next iteration of this loop will be executed and this will continue to happen until for an iteration $\widehat{Q}$ is a constant multiple of $\widehat{Q}_{k,1}(A\mathbf{x} + \mathbf{b}), \ldots, \widehat{Q}_{k,m_k}(A\mathbf{x} + \mathbf{b})$. The claim follows from Claims 5.25 and 5.26. □

We now prove a structural result.

**Claim 5.27** *Suppose that $Q_1 \cdots Q_m + u$ is a canonical ROF. Let $T = Q_1 \cdots Q_m + \ell(\mathbf{z})$, where $\ell$ is a non-zero affine form and $Q_1 \cdots Q_m$ is not a quadratic polynomial. Then $T$ is reducible if and only if for some $l \in [m]$, $Q_l$ is an affine form in a single variable and $\ell$ is a constant multiple of $Q_l$.*

**Proof:** One direction is simple. If $\ell = c \cdot \widehat{Q}_l$ for some $l \in [m]$, then it is clear that $T$ is reducible.

For the other direction, pick a $z \in \text{var}(\ell)$ and let its coefficient be $c$. Note that there must exist an $l \in [m]$ such that $z \in \text{var}(Q_l)$, for otherwise $T$ is in the orbit of the $+$-rooted ROF $Q_1 \cdots Q_m + z$ and therefore irreducible. So assume without loss of generality that $z \in \text{var}(Q_1)$. Let $T = r_1 \cdot r_2$ where $r_1$ is an irreducible factor of $T$ containing $z$ and $r_2$ is not necessarily irreducible. Note that as $T$ is multilinear $r_1$ and $r_2$ are variable disjoint. Now,

$$r_1 r_2 = Q_1 Q + c \cdot z + \ell'$$

where $Q = Q_2 \cdots Q_m$ and $\ell = c \cdot z + \ell'$. Let $Q_1 = g_1 \cdot z + g_2$ and $r_1 = w_1 \cdot z + w_2$ where $g_1, g_2, w_1$, and $w_2$ are $z$-free polynomials. Then,

$$(w_1 \cdot z + w_2) \cdot r_2 = (g_1 \cdot z + g_2) \cdot Q + c \cdot z + \ell'. \tag{5.9}$$

**Observation 5.23** $w_1 \in \mathbb{F}^\times$.

**Proof:** Taking derivatives with respect to $z$ on both sides of Equation (5.9), we see that $w_1 \cdot r_2 = g_1 \cdot Q + c$. If $g_1$ is a non-constant polynomial, then the latter is a $+$ rooted ROF and therefore from Fact 2.1, irreducible. Hence, $r_2$ is irreducible and $g_1 \in \mathbb{F}$.

If $g_1$ is a constant and $g_2$ is also a constant, then as $Q_1 Q + u$ is a canonical ROF, either $Q$ must be a product of at least two $+$ rooted ROFs or if it is just a single $+$ rooted ROF, then it can not have a constant attached to its top-most $+$ gate. In either of these cases $g_1 \cdot Q + c$ is irreducible and hence we again get that $r_2$ is irreducible and $g_1 \in \mathbb{F}$.

Suppose that $g_1$ is a constant, but $g_2$ is not. As $Q$ is non-constant (since $m \geq 2$), there exists a $z_2 \in \text{var}(Q)$. Then, for any $z_1 \in \text{var}(g_2)$, as $zz_1$ does not appear on the right side of Equation (5.9), $z_1 \notin \text{var}(w_1)$ or $\text{var}(r_2)$. Also, for any $z_2' \in \text{var}(Q)$, $z_2' \notin \text{var}(w_1)$. This is so because, for any $z_1' \in \text{var}(g_2)$, $z_1' z_2'$ appears on the right side of Equation (5.9). If $z_1'$ were to be in $\text{var}(w_1)$, then $z_1' z_2'$ can not appear on the left side of (5.9) since $z_1' \notin \text{var}(w_1)$ or $\text{var}(r_2)$. Thus, no variable in $\text{var}(g_2)$ or $\text{var}(Q)$ can be present in $\text{var}(w_1)$ forcing $w_1$ to be a constant. □

Assume without loss of generality that $w_1 = 1$. Thus,

$$(z + w_2) \cdot r_2 = (g_1 \cdot z + g_2) \cdot Q + c \cdot z + \ell',$$

which implies that $r_2 = g_1 \cdot Q + c$. Hence,

$$(z + w_2) \cdot (g_1 \cdot Q + c) = (g_1 \cdot z + g_2) \cdot Q + c \cdot z + \ell'.$$

**Observation 5.24** $g_1 \in \mathbb{F} \setminus \{0\}$.

**Proof:** By contradiction. Suppose that $g_1$ is a non-constant polynomial. Then we have

$$\begin{aligned}(z + w_2) \cdot (g_1 \cdot Q + c) &= (g_1 \cdot z + g_2) \cdot Q + c \cdot z + \ell' \\ \implies (w_2 \cdot g_1 - g_2)Q &= -c \cdot w_2 + \ell'.\end{aligned}$$

Suppose that $w_2 \cdot g_1 - g_2 \neq 0$. Then $Q$ and $w_2$ must be variable disjoint, we get

1. $Q$ is linear and so as it is a canonical ROF is an affine form in a single variable,
2. $w_2 \cdot g_1 - g_2 \in \mathbb{F}^\times$, say it is $c'$, and
3. $w_2$ is an affine form.

Hence, $Q_1 = g_1 \cdot z + g_2 = g_1 \cdot z + g_1 \cdot w_2 - c'$. We claim that $c'$ must be 0. It given that $Q_1 Q + u$ is a canonical ROF and $Q$ is an affine form in a single variable. Thus it follows from property 6 of the definition of a canonical ROF that $Q_1$ can not have a constant attached to its top-most $+$ gate; hence $c' = 0$. However this contradicts our assumption that $c' = w_2 \cdot g_1 - g_2 \neq 0$. Hence, $w_2 \cdot g_1 - g_2 = 0$. So, $g_2 = w_2 \cdot w_1$. This implies that $Q_1 = g_1(z + w_2)$, which implies that $g_1 \in \mathbb{F}^\times$ because $Q_1$ being a $+$-rooted ROF is irreducible. □

Assume without loss of generality that $g_1 = 1$; if it is not, then replace $Q$ by $g_1 \cdot Q$ and $g_2$ by $g_1^{-1} \cdot g_2$. Then,

$$\begin{aligned}(z + w_2)(Q + c) &= (z + g_2) \cdot Q + c \cdot z + \ell' \\ \implies (w_2 - g_2)Q &= -c \cdot w_2 + \ell'.\end{aligned}$$

Then, just as before, $Q$ and $w_2$ are variable disjoint. Thus,

1. $Q$ is linear and so as it is a canonical ROF is an affine form in a single variable,
2. $w_2 - g_2 \in \mathbb{F}^\times$, say it is $c''$, and
3. $w_2$ is an affine form.

Suppose $w_2 - g_2 \neq 0$. Hence, $Q_1 = z + w_2 - c''$ and as $Q_1$ is canonical, $w_2$ has to be a constant. But then, $Q_1$ and $Q$ are both affine which contradicts the hypothesis that $Q_1 \cdots Q_m$ is not a quadratic polynomial. Thus, $w_2 = g_2$. Then,

$$\begin{aligned}(z + w_2)(Q + c) &= (z + w_2) \cdot Q + \ell \\ \implies (z + w_2) \cdot c &= \ell.\end{aligned}$$

Now, as $Q_1 = (z + w_2)$ is a canonical ROF, $w_2 \in \mathbb{F}$. Hence, $\ell = c \cdot Q_1$. □

We complete the proof of the lemma by combining Claims 5.24 and 5.27. As the $k$-th iteration of the outer loop only works with $\mathbf{z}_k$, we can analyse each iteration in isolation. Claim 5.24 implies that after the execution of the inner loop, $\widehat{T}$ is reducible. Initially, $\widehat{T} = \widehat{T}_k(A\mathbf{x}+\mathbf{b}) + [\ell(A\mathbf{x}+\mathbf{b})]_{\mathbf{z}_k} + \gamma'$ for some $\gamma' \in \mathbb{F}$. The inner loop only subtracts an affine form from $\widehat{T}$. So after the execution of the inner loop, $\widehat{T} = \widehat{T}_k(A\mathbf{x}+\mathbf{b}) + \widetilde{\ell}$ for some affine form $\widetilde{\ell}$. Notice that $\widehat{T} \in \text{orb}(T_k + \widetilde{\ell}')$ for $\widetilde{\ell}' := \widetilde{\ell}(A^{-1}B^{-1}(\mathbf{x} - B\mathbf{b} - \mathbf{d}))$. Let $T_k = Q_{k,1} \dots Q_{k,m_k}$, $\widehat{T}_k = \widehat{Q}_{k,1} \dots \widehat{Q}_{k,m_k}$, and $\widehat{Q}_{k,l} = Q_{k,l}(B\mathbf{x}+\mathbf{d})$ for all $l \in [m_k]$. Claim 5.24 implies that $\widehat{T}$ is reducible. Then, if none of the factors $\widehat{Q}_{k,1}, \dots, \widehat{Q}_{k,m_k}$ are linear, Claim 5.27 implies that $\widetilde{\ell} = 0$. On the other hand, if one of the factors, say $\widehat{Q}_{k,1}$ is linear, then Claim 5.27 implies that $\widetilde{\ell} = c'_k \cdot \widehat{Q}_{k,1}$ for some $c'_k \in \mathbb{F}$ and $\widehat{T} = \widehat{Q}_{k,1}\left(\widehat{Q}_{k,2} \dots \widehat{Q}_{k,m_k} + c'_k\right)$. In the first case, $\ell_k$ must be $[\ell(A\mathbf{x})]_{\mathbf{z}_k} + \gamma'$. Because $Q_{k,1}$ is a variable connected to a $\times$ gate computing a polynomial of degree at least 3, Lemma 5.4 implies that $\widehat{Q}_{k,1}(A\mathbf{x}+\mathbf{b})$ is a constant multiple of a variable, say $z$. Thus, in this case, $\ell_k$ and $[\ell]_{\mathbf{z}_k}$ must agree on the coefficients of all $z' \in \mathbf{z}_k$ except perhaps that of $z$. For every $k \in \{s_2+1, \dots, s'\}$, every $k \in [s_2]$ such that $T_k$ is in the first case, and every $k \in [s_2]$ in the second case that looks like $xQ$, where $Q$ has a top-dangling $y$, let $T'_k = T_k$. For every other $k$, let $T'_k = Q_{k,1}\left(Q_{k,2} \dots Q_{k,m_k} + c'_k\right)$. We also redefine $\mathbf{d}$ as follows: For every $k \in [s_2]$ such that $T_k$ is in the second case, it looks like $zQ$, and the top dangling variable of $Q$ is $y$, we add $c'_k$ to the $y$-th entry of $\mathbf{d}$; all other entries remain unchanged. If we redefine $\widehat{T}(A\mathbf{x}+\mathbf{b}) = T'_k(BA\mathbf{x} + B\mathbf{b} + \mathbf{d})$, and $\mathtt{C}' = T'_1 + \dots + T'_s + \gamma$, then $f(A\mathbf{x}+\mathbf{b}) = \widehat{T}_1 + \dots + \widehat{T}_s + \gamma = \mathtt{C}'(BA\mathbf{x} + B\mathbf{b} + \mathbf{d})$. Now, when we map $u_0$ to $u_0 - \ell'$ in $f(A\mathbf{x}+\mathbf{b})$, all its terms are variable disjoint and $\ell(A\mathbf{x}+\mathbf{b})$ becomes $u_0 + \alpha$ for some $\alpha \in \mathbb{F}$. □

*Remark.* Notice that $\mathtt{C}'$ need not be a canonical ROF. However, for all $k \in [s_2]$, all the factors of $T'_k$ are still canonical. As we only recursively perform equivalence test on the factors of $\widehat{T}_k(A\mathbf{x})$, $\mathtt{C}'$ not being canonical is not a problem.

#### 5.5.3.4 Obtaining efficient black-box access to a term

The next algorithm is used to obtain black-box access to a term $\widehat{T}_k(A\mathbf{x})$ using *a single* query to $f$.

**Algorithm 8** Compute-Term-Black-Box($g$)

**Input:** Black-box access to a term of $f(A\mathbf{x})$ plus an unknown constant.

**Output**: Black-box access to the term using just one query to the black-box of $f$.

1: $F \leftarrow$ a subset of $\mathbb{F}$ of size at least $n^5$.
2: Obtain black-box access to $\det(H_g)$ with respect to $\text{var}(g)$ and factorize it using the algorithm in [KT90]. $N \leftarrow$ set of black-boxes of the irreducible factors.
3: **for** $r \in N$ **do**
4: $\quad$ $\mathbf{a} \leftarrow$ a vector of size $|\text{var}(g)|$ containing random elements from $F$. For a fresh variable $t$, interpolate $r(t\mathbf{a})$ and $g(t\mathbf{a})$.
5: $\quad$ Discover $r'(t)$ and $\beta \in \mathbb{F}$ such that $r(t\mathbf{a})r'(t) + \beta = g(t\mathbf{a})$ by solving a system of linear equations in the coefficients of $r'$ and $\beta$.
6: $\quad$ If $g - \beta$ is reducible, then return black-box access to $g - \beta$.
7: **end for**

The following lemma establishes the correctness of the above algorithm.

**Lemma 5.6 (Correctness of Algorithm 8)** *Compute-Term-Black-Box*$(f\,(A\,(\mathbf{z}_k, \mathbf{x} \setminus \mathbf{z}_k = \mathbf{0})))$ *gives black-box access to* $\widehat{T}_k(A\mathbf{x})$ *with high probability. Also, one query to* $\widehat{T}_k(A\mathbf{x})$ *needs just one query to* $f$.

**Proof:** As $f(A\mathbf{x}) = \widehat{T}_1(A\mathbf{x}) + \cdots + \widehat{T}_s(A\mathbf{x}) + \gamma$, $\widehat{T}_1(A\mathbf{x}), \ldots, \widehat{T}_s(A\mathbf{x})$ are variable disjoint, and for all $k \in [s_2]$ $\mathbf{z}_k = \text{var}\left(\widehat{T}_k(A\mathbf{x})\right)$, $f\,(A\,(\mathbf{z}_k, \mathbf{x} \setminus \mathbf{z}_k = \mathbf{0})) = \widehat{T}_k(A\mathbf{x}) + \gamma'$ for some $\gamma' \in \mathbb{F}$. So all that needs to be done to obtain black-box access to $\widehat{T}_k(A\mathbf{x})$ is to find $\gamma'$ and subtract it from $g = f\,(A\,(\mathbf{z}_k, \mathbf{x} \setminus \mathbf{z}_k = \mathbf{0}))$. We show that this is exactly what the algorithm does.

In the algorithm, $N$ is the set of irreducible factors of $\det(H_g)$. Suppose that $\widehat{T}_k = \widehat{Q}_{k,1} \cdots \widehat{Q}_{k,m_k}$. It follows from Claim 5.7 and Fact 2.4, that a non-zero constant multiple of at least one of the factors $\widehat{Q}_{k,1}(A\mathbf{x}), \ldots, \widehat{Q}_{k,m_k}(A\mathbf{x})$ is in $N$ along with some other 'bad' factors. First let us analyse the behaviour of the for loop of lines 3-7 when $r$ is a constant multiple of one of the $\widehat{Q}_{k,1}(A\mathbf{x}), \ldots, \widehat{Q}_{k,m_k}(A\mathbf{x})$. In this case, there exist $r'(t)$ and $\beta \in \mathbb{F}$ such that $r(t\mathbf{a})r'(t) + \beta = g(t\mathbf{a})$; one solution is $r'(t) = \left(\widehat{T}_k(A\mathbf{x})/r\right)(A(t\mathbf{a}))$ and $\beta = \gamma'$. $r'(t)$ and $\beta$ can be discovered as follows: first interpolate $r(t\mathbf{a})$ and $g(t\mathbf{a})$ which are univariate polynomials in $t$. Treat the coefficients of $r'(t)$ and $\beta$ as unknowns. Then, by equating the coefficients of monomials on both sides of $r(t\mathbf{a})r'(t) + \beta = g(t\mathbf{a})$, we get a system of linear equations in these unknowns which the algorithm solves. As mentioned before, this system has a solution, we now show it is unique.

Suppose that there existed two solutions $r'_1(t), \beta_1$ and $r'_2(t), \beta_2$. Then, $r(t\mathbf{a})\,(r'_1(t) - r'_2(t)) = \beta_2 - \beta_1$. Because $\mathbf{a}$ is chosen randomly, with high probability $r(t\mathbf{a})$ is a non-constant polynomial in $t$. Thus, this is only possible when $r'_1(t) = r'_2(t)$ and $\beta_1 = \beta_2 = \gamma'$. Hence, if $r$ is a

constant multiple of one of the $\widehat{Q}_{k,1}(A\mathbf{x}), \ldots, \widehat{Q}_{k,m_k}(A\mathbf{x})$, then $\beta = \gamma'$ and $g - \beta$ is reducible. Thus, the algorithm returns a black-box of $g - \beta = \widehat{T}_k(A\mathbf{x})$.

On the other hand, when $r$ is one of the bad factors, there are two cases: $\beta = \gamma'$ and $\beta \neq \gamma'$. In the first case there is noting to prove. On the other hand, if $\beta \neq \gamma'$, then notice that $g + \gamma' - \beta$ is in the orbit of $T_k + \gamma' - \beta$. As the latter is a $+$-rooted ROF, Fact 2.1 implies that it is irreducible. Hence $g - \beta$ is also irreducible. Now, the fact that the for loop was executed for this bad factor implies that in all of the previous iterations, $r$ must have been a bad factor, for otherwise as seen above, the loop would have terminated. This along with the fact that $N$ contains a constant multiple of at least one of the $\widehat{Q}_{k,1}(A\mathbf{x}), \ldots, \widehat{Q}_{k,m_k}(A\mathbf{x})$ implies that in this case, the next iteration of the loop will be executed. This will continue to happen until the loop is executed for an $r$ which is a non-zero constant multiple of one of the $\widehat{Q}_{k,1}(A\mathbf{x}), \ldots, \widehat{Q}_{k,m_k}(A\mathbf{x})$ and $\gamma'$ is discovered.

Notice that $\widehat{T}_k(A\mathbf{x}) = f\left(A\left(\mathbf{z}_k, \mathbf{x} \setminus \mathbf{z}_k = \mathbf{0}\right)\right) - \beta$. Hence, to query $\widehat{T}_k(A\mathbf{x})$ at $\mathbf{z}_k = \mathbf{a}$, for some $\mathbf{a} \in \mathbb{F}^{|\mathbf{z}_k|}$, $f(A\mathbf{x})$ just needs to be queried at the point $(\mathbf{z}_k = \mathbf{a}, \mathbf{x} \setminus \mathbf{z}_k = \mathbf{0})$ and then $\beta$, which is a fixed, known constant, subtracted from the result. Now as $A$ is known to us, to query $f(A\mathbf{x})$ at any point, we just need to query $f$ at one point. □

#### 5.5.3.5 Proof of Lemma 5.2

By induction on the product-depth $\Delta$ of $\mathtt{C}$. If $\Delta = 0$, as $\mathtt{C}$ is a canonical ROF, $\mathtt{C} = x_1$ and $f$ is an affine form. Since all variables in $f$ are essential, $n = 1$ and $f = \alpha_1 x_1 + \alpha_0$ for some $\alpha_0, \alpha_1 \in \mathbb{F}$, $\alpha_1 \neq 0$. Then, $f(I_{n\times n}\mathbf{x}) \in \text{PS-orb}(\mathtt{C})$ and the algorithm works correctly for product-depth 0 ROFs.

Assume that the algorithm works correctly for all polynomials in the orbit of a canonical ROF of product-depth $\Delta \geq 0$ and let $\mathtt{C}$ be a canonical ROF of product-depth $\Delta + 1$. Recall that the algorithm is given black-box access to an $f \in \mathbb{F}[\mathbf{x}]$ such that there exist a $B \in \text{GL}(n, \mathbb{F})$ and a $\mathbf{d} \in \mathbb{F}^n$ satisfying $f = \mathtt{C}(B\mathbf{x} + \mathbf{d})$. Also, there are no redundant variables in $f$. Further $\mathtt{C} = T_1 + \cdots + T_s + \gamma$, where $T_1, \cdots, T_s$ are $\times$-rooted canonical ROFs and $\gamma \in \mathbb{F}$. Also, $f = \widehat{T}_1 + \cdots + \widehat{T}_s + \gamma$, where for all $k \in [s]$, $\widehat{T}_k = T_k(B\mathbf{x} + \mathbf{d})$. $T_1, \ldots, T_{s_1}$ are the good terms of $\mathtt{C}$, while, $\widehat{T}_1, \ldots, \widehat{T}_{s_1}$ are the good terms of $f$. Similarly, $T_{s_1+1}, \ldots, T_{s_2}$ are the bad terms of $\mathtt{C}$, while $\widehat{T}_{s_1+1}, \ldots, \widehat{T}_{s_2}$ are the bad terms of $f$. If $\mathtt{C}$ has a top dangling variable, it is $T_s = x_n$, $s' := s - 1$, and $T_{s_2+1} + \cdots + T_{s-1}$ is the top quadratic form. Otherwise, $T_{s_2+1} + \cdots + T_s$ is the top quadratic form and $s' := s$. If $\mathtt{C}$ has a top dangling variable, then $\ell := B \circ x_n + d_n$, where $d_n$ is the $n$-th coordinate of $\mathbf{d}$.

It follows from Lemma 5.4 that after Step 4 is executed, $\widehat{T}_1(A\mathbf{x} + \mathbf{b}), \ldots, \widehat{T}_{s'}(A\mathbf{x} + \mathbf{b})$, and

hence $\widehat{T}_1(A\mathbf{x}), \ldots, \widehat{T}_{s'}(A\mathbf{x})$ are variable disjoint while $\sum_{k=s_2+1}^{s'} \widehat{T}_k(A\mathbf{x}) = (y_1 + c_1)(y_2 + c_2) + \cdots + (y_{2m-1} + c_{2m-1})(y_{2m} + c_{2m})$, where $c_1, \ldots, c_{2m} \in \mathbb{F}$. Then, Claim 5.23 implies that after Step 14, $\mathbf{z}_1, \ldots, \mathbf{z}_{s_2}$ are variable sets of $\widehat{T}_1(A\mathbf{x}), \ldots, \widehat{T}_{s_2}(A\mathbf{x})$, respectively. Further, if there is a dangling variable, then Lemma 5.5 implies that $\ell(A\mathbf{x}) = u_0 + c$ for some $c \in \mathbb{F}$. However, now $f \in \text{orb}(\mathtt{C}')$, where $\mathtt{C}'$ is as defined in the proof of Lemma 5.5. If $\mathtt{C}$ does not have a top-dangling variable, then let $\mathtt{C}' = \mathtt{C}$. We first show that after Step 35 is executed, $f(A\mathbf{x}) \in \text{PS-orb}(\mathtt{C}')$. As there are no redundant variables in $\widehat{T}_1(A\mathbf{x}), \ldots, \widehat{T}_{s'}(A\mathbf{x})$, for all $k \in [s_2]$, $|\mathbf{z}_k| = \left|\text{var}\left(T'_k\right)\right|$ and $|\mathbf{y}| = \left|\text{var}\left(\sum_{k=s_2+1}^{s} T'_k\right)\right|$. So there exists a permutation matrix $P_0 \in M(n, \mathbb{F})$ (that maps $u_0$ to $u_0$) such that for all $k \in [s_2]$, $\text{var}(T'_k(P_0\mathbf{x})) = \mathbf{z}_k$ and $\text{var}\left(\sum_{k=s_2+1}^{s} T'_k(P_0\mathbf{x})\right) = \mathbf{y}$. There exists a $B' \in \text{GL}(n, \mathbb{F})$ and $\mathbf{d}' \in \mathbb{F}^n$ such that $f(A\mathbf{x}) = \mathtt{C}'(P_0(B'\mathbf{x} + \mathbf{d}'))$. Notice that it suffices to prove that $f(A\mathbf{x}) \in \text{PS-orb}(\mathtt{C}'(P_0\mathbf{x}))$. We now analyse the for loop of lines 21-34. For any $k \in [s_2]$, as the $k$-th iteration of the loop only works on $\widehat{T}_k(A\mathbf{x})$ and $\mathbf{z}_k$, we can look at it in isolation.

**Claim 5.28** *For any $k \in [s_2]$, after the execution of the $k$-th iteration of the for loop of lines 21-34 there exists a permutation matrix $P_k \in M(|\mathbf{z}_k|, \mathbb{F})$, an invertible scaling matrix $S_k \in M(|\mathbf{z}_k|, \mathbb{F})$, and a $\mathbf{b}_k \in \mathbb{F}^{|\mathbf{z}_k|}$ such that $\widehat{T}_k\left(A\left(A_k\mathbf{z}_k, \mathbf{x} \setminus \mathbf{z}_k\right)\right) = T'_k(P_0\left(P_kS_k\mathbf{z}_k + \mathbf{b}_k, \mathbf{x} \setminus \mathbf{z}_k\right))$.*

**Proof:** Fix a $k \in [s_2]$. The hypothesis of Claim 5.6 is satisfied. Hence after Step 22 is executed, $\widehat{T}$ is the black-box of $\widehat{T}_k(A\mathbf{x})$. Suppose that $\widehat{T}_k(A\mathbf{x}) = \widehat{Q}_{k,1}(A\mathbf{x}) \cdots \widehat{Q}_{k,m_k}(A\mathbf{x})$, the corresponding term $T'_k(P_0\mathbf{x})$ of $\mathtt{C}'(P_0\mathbf{x})$ is a product of $+$-rooted canonical sub-ROFs $Q_{k,1}, \ldots, Q_{k,m_k}$ and for all $l \in [m_k]$, $\widehat{Q}_{k,l}(A\mathbf{x}) = Q_{k,l}(B'\mathbf{x} + \mathbf{d}')$. The factors $\widehat{Q}_1, \ldots, \widehat{Q}_{m_k}$ of $\widehat{T}$ computed in Step 25 are non-zero constant multiples of $\widehat{Q}_{k,1}(A\mathbf{x}), \ldots, \widehat{Q}_{k,m_k}(A\mathbf{x})$, respectively, say they are $c_1\widehat{Q}_{k,1}(A\mathbf{x}), \ldots, c_{m_k}\widehat{Q}_{k,m_k}(A\mathbf{x})$; here $\prod_{l\in[m_k]} c_l = 1$. Now, as $Q_{k,1}, \ldots, Q_{k,m_k}$ are variable disjoint ROFs, $N_{ess}\left(Q_{k,1} \cdots Q_{k,m_k}\right) = N_{ess}\left(Q_{k,1}\right) + \cdots + N_{ess}\left(Q_{m_k}\right)$. Also, for all $l \in [m_k]$ $N_{ess}\left(\widehat{Q}_l\right) = N_{ess}\left(\widehat{Q}_{k,l}(A\mathbf{x})\right) = N_{ess}\left(Q_{k,l}\right)$ and similarly $N_{ess}\left(\widehat{Q}_1 \cdots \widehat{Q}_{m_k}\right) = N_{ess}\left(Q_{k,1} \cdots Q_{k,m_k}\right)$. This means that $N_{ess}\left(\widehat{Q}_1 \cdots \widehat{Q}_{m_k}\right) = N_{ess}\left(\widehat{Q}_1\right) + \cdots + N_{ess}\left(\widehat{Q}_{m_k}\right)$. So from Claim 5.2, there exists an $A_{k,0} \in \text{GL}(|\mathbf{z}_k|, \mathbb{F})$ such that $\widehat{Q}_1(A_{k,0}\mathbf{z}_k), \ldots, \widehat{Q}_{m_k}(A_{k,0}\mathbf{z}_k)$ are variable disjoint. It also implies that $\widehat{Q}_1(A_{k,0}\mathbf{z}_k), \ldots, \widehat{Q}_{m_k}(A_{k,0}\mathbf{z}_k)$ do not contain any redundant variables. In Step 26, $\widehat{Q}_l$ has been updated to be $\widehat{Q}_l(A_{k,0}\mathbf{z}_k)$ for all $l \in [m_k]$. Now $\widehat{Q}_l(A_{k,0}\mathbf{z}_k)$, $|\text{var}\left(Q_{k,l}\right)| = |\mathbf{z}_{k,l}|$, where $\mathbf{z}_{k,l} = \text{var}\left(\widehat{Q}_l(\mathbf{z}_k)\right)$. So, there exists a permutation matrix $P_{k,0} \in M(|\mathbf{z}_k|, \mathbb{F})$ such that for all $l \in [m_k]$, $\text{var}\left(Q_{k,l}(P_{k,0}\mathbf{z}_k)\right) = \mathbf{z}_{k,l}$.

Much like the outer loop, the $l$-th iteration of the inner loop of lines 29-32, also only works with $\widehat{Q}_l(\mathbf{z}_k)$ and $\mathbf{z}_{k,l}$; so we can also look at an iteration of this loop in isolation. We now analyse the $l$-th iteration of this loop for some $l \in [m_k]$. $\mathbf{a}$ is a random vector of size $|\mathbf{z}_k|$ and $\mathbf{a}'$ is

$\mathbf{a}$ restricted to entries corresponding to $\mathbf{z}_k \setminus \mathbf{z}_{k,l}$. Also $\beta_l = \prod_{l' \in [m_k] \setminus \{l\}} \widehat{Q}_{l'}(\mathbf{z}_{k,l}, \mathbf{z}_k \setminus \mathbf{z}_{k,l} = \mathbf{a}')$. Because $\mathbf{a}$ is random, $\beta_l \neq 0$ with high probability. Thus $\widehat{Q}_l = \beta_l^{-1} \widehat{T}\left(A_{k,0}\left(\mathbf{z}_{k,l}, \mathbf{z}_k \setminus \mathbf{z}_{k,l} = \mathbf{a}'\right)\right) = c_l \widehat{Q}_{k,l}(A(A_{k,0}\mathbf{z}_k, \mathbf{x} \setminus \mathbf{z}_k))$. Now consider a product-depth $\Delta$ canonical ROF $Q_l$ obtained by multiplying $Q_{k,l}(P_{k,0}\mathbf{z}_k)$ by $c_l$, pushing it down to the leaves, and removing it from any non-constant leaf. Let $B'' = {P'_{k,0}}^{-1} B' A'_{k,0}$, where $P'_{k,0} \in M(n, \mathbb{F})$ maps every $z \in \mathbf{z}_k$ to $P_{k,0} \circ z$ and every other variable to itself, while $A'_{k,0} \in \mathrm{GL}(n, \mathbb{F})$ maps every $z \in \mathbf{z}_k$ to $A_{k,0} \circ z$ and every other variable to itself. Also, let $\mathbf{d}'' = {P'_{k,0}}^{-1} \mathbf{d}'$. It can be verified that $\widehat{Q}_l = Q_l\left(B''\mathbf{x} + \mathbf{d}''\right)$. To recursively perform equivalence test on $\widehat{Q}_l$ we shall show that there exists a $B_l \in \mathrm{GL}(|\mathbf{z}_{k,l}|, \mathbb{F})$ and a $\mathbf{d}_l \in \mathbb{F}^{|\mathbf{z}_{k,l}|}$ such that $\widehat{Q}_l(\mathbf{z}_{k,l}) = Q_l\left(B_l \mathbf{z}_{k,l} + \mathbf{d}_l\right)$.

Because $\mathrm{var}\left(Q_l\right) = \mathbf{z}_{k,l}$, $\widehat{Q}_l(\mathbf{z}_{k,l}) = Q_l\left(\left[B''\right]_{\mathbf{z}_{k,l}} \mathbf{x} + \left[\mathbf{d}''\right]_{\mathbf{z}_{k,l}}\right)$, where $\left[B''\right]_{\mathbf{z}_{k,l}}$ and $\left[\mathbf{d}''\right]_{\mathbf{z}_{k,l}}$ are $B''$ and $\mathbf{d}''$ restricted to the rows corresponding to $\mathbf{z}_{k,l}$. Also, since $\mathrm{var}\left(\widehat{Q}_l\right) = \mathbf{z}_{k,l}$, $\widehat{Q}_l(\mathbf{z}_{k,l}) = Q_l\left(\left[B''\right]_{\mathbf{z}_{k,l} \times \mathbf{z}_{k,l}} \mathbf{z}_{k,l} + \left[\mathbf{d}''\right]_{\mathbf{z}_{k,l}}\right)$, where $\left[B''\right]_{\mathbf{z}_{k,l} \times \mathbf{z}_{k,l}}$ is $B''$ restricted to the rows and columns corresponding to $\mathbf{z}_{k,l}$. It follows from Observation 2.8 that $\left[B''\right]_{\mathbf{z}_{k,l} \times \mathbf{z}_{k,l}}$ is invertible. So we can set $B_l = \left[B''\right]_{\mathbf{z}_{k,l} \times \mathbf{z}_{k,l}}$ and $\mathbf{d}_l = \left[\mathbf{d}''\right]_{\mathbf{z}_{k,l}}$.

Thus, by the induction hypothesis, $A_{k,l}$ computed in Step 31, is such that there exist a permutation matrix $P_{k,l} \in M(|\mathbf{z}_{k,l}|, \mathbb{F})$, a scaling matrix $S_{k,l} \in M(|\mathbf{z}_{k,l}|, \mathbb{F})$ and a $\mathbf{b}_{k,l} \in \mathbb{F}^{|\mathbf{z}_{k,l}|}$ satisfying $\widehat{Q}_l(A_{k,l}\mathbf{z}_{k,l}) = Q_l(P_{k,l}S_{k,l}\mathbf{z}_{k,l} + \mathbf{b}_{k,l})$. Since this is true for all $l \in [m_k]$, after the execution of the for loop of lines 29-32 and Step 33, for all $l \in [m_k]$, $c_l \widehat{Q}_{k,l}(A(A_k\mathbf{z}_k, \mathbf{x} \setminus \mathbf{z}_k)) = c_l Q_{k,l}(P_k S_k \mathbf{z}_k + \mathbf{b}_k)$, where for all $l \in [m_k]$ and $z \in \mathbf{z}_{k,l}$, $P_k$ maps $z$ to $P_{k,l} \circ \mathbf{z}_{k,l}$, $S_k$ maps $z$ to $S_{k,l} \circ \mathbf{z}_{k,l}$ and the $z$-th coordinate of $\mathbf{d}_k$ is the same as that of $\left[P_{k,0}\right]_{\mathbf{z}_{k,l} \times \mathbf{z}_{k,l}} \mathbf{d}_{k,l}$. Because $\prod_{l \in [m_k]} c_l = 1$, $\widehat{T}_k(A(A_k\mathbf{z}_k, \mathbf{x} \setminus \mathbf{z}_k)) = T'_k(P_0(P_k S_k \mathbf{z}_k + \mathbf{b}_k, \mathbf{x} \setminus \mathbf{z}_k))$, proving the claim. □

Now we finish the proof of the lemma assuming the claim. After Step 19, there already exists a permutation matrix $P_{s_2+1} \in M(|\mathbf{y}|, \mathbb{F})$, an invertible scaling matrix $S_{s_2+1} \in M(|\mathbf{y}|, \mathbb{F})$ (such that $P_{s_2+1}S_{s_2+1} \circ u_0 = u_0$) and a $\mathbf{b}_{s_2+1} \in \mathbb{F}^{|\mathbf{y}|}$ such that

$$\sum_{k=s_2+1}^{s} \widehat{T}_k\left(A\mathbf{x}\right) = \sum_{k=s_2+1}^{s} T'_k(P_0\left(P_{s_2+1}S_{s_2+1}\mathbf{y} + \mathbf{b}_{s_2+1}, \mathbf{z}\right)).$$

Let $P \in M(|\mathbf{z}|, \mathbb{F})$ be a permutation matrix that maps every $z \in \mathbf{z}_k$ to $P_k \circ z$ for all $k \in [s_2]$ and every $y \in \mathbf{y}$ to $P_{s_2+1} \circ y$. Similarly, let $S \in M(|\mathbf{z}|, \mathbb{F})$ be a scaling matrix that maps every $z \in \mathbf{z}_k$ to $S_k \circ z$ for all $k \in [s_2]$ and and every $y \in \mathbf{y}$ to $S_{s_2+1} \circ y$. Also, let $\mathbf{b} \in \mathbb{F}^n$ be such that for all $k \in [s_2]$, its coordinates corresponding to $\mathbf{z}_k$ are $\mathbf{b}_k$ and those corresponding to $\mathbf{y}$ are $\mathbf{b}_{s_2+1}$. As $A'_0$ maps every $z \in \mathbf{z}_k$ to $A_k \circ z$ , $\forall k \in [s_2]$ and maps every $y \in \mathbf{y}$ to itself, after $A$ is set to $AA'$ we have that $\widehat{T}_k(A\mathbf{x}) = T'(P_0(PS\mathbf{x} + \mathbf{b}))$ yielding $f(A\mathbf{x}) = \mathtt{C}'(P_0\left(PS\mathbf{x} + \mathbf{b}\right)) \in \text{PS-orb}\left(\mathtt{C}'\right)$.

If $\mathtt{C}' = \mathtt{C}$, then we are done. Otherwise, in Step 38 $f(A\mathbf{x})$ is reconstructed. From Lemma 5.7, we have that the terms of the reconstructed ROF $f'$ are constant multiples of $T'_1(P_0(PS\mathbf{x}+\mathbf{b})), \ldots, T'_s(P_0(PS\mathbf{x}+\mathbf{b}))$. In fact, as $T'_1(P_0(PS\mathbf{x}+\mathbf{b})), \ldots, T'_s(P_0(PS\mathbf{x}+\mathbf{b}))$ are variable disjoint and hence linearly independent, the terms are exactly

$$T'_1(P_0(PS\mathbf{x}+\mathbf{b})), \ldots, T'_s(P_0(PS\mathbf{x}+\mathbf{b})).$$

Fix any $k$ such that $T_k \neq T'_k$. Recall that in this case, $T_k = zQ_{k,2}\cdots Q_{k,m_k}$ and

$$T'_k = z\left(Q_{k,2}\cdots Q_{k,m_k} + c_k\right).$$

From Lemma 5.7, the corresponding term of $f'$ is

$$(c\alpha'_1 x + c\alpha'_0)\left(c^{-1}\left(Q_{k,2}\cdots Q_{k,m_k}\right)(P_0(PS\mathbf{x}+\mathbf{b})) + c^{-1}c_k\right),$$

where $\alpha'_1 x = P_0 PS \circ z$, $\alpha'_0$ is the $z$-th entry of $P_0\mathbf{b}$, and $c \neq 0$. Hence in Step 38, $\alpha_1 = c\alpha'_1$ and $\beta = c^{-1}c_k$. Notice that $P_0 PS \circ u_0 = u_0$. So after $A$ is updated to map $u_0$ to $u_0 - \alpha_1\beta x$, $\widehat{T}_k(A\mathbf{x}) = T_k(P_0(PS\mathbf{x}+\mathbf{b}))$ yielding $f(A\mathbf{x}) \in \text{PS-orb}(\mathtt{C})$.

#### 5.5.3.6 Running time of Algorithm 3

Notice that whenever a recursive call is made to Find-Equivalence(), it is for a polynomial in the orbit of a distinct $+$-rooted sub-ROF or variable of the original ROF $\mathtt{C}$. As there are at most $n$ many $+$-rooted sub-ROFs and $n$ variables, there are at most $2n$ many recursive calls. Thus to prove that Find-Equivalence() runs $\text{poly}(n)$ time we only need to argue that the time required by each recursive call (not counting the time spent in any sub-calls) is $\text{poly}(n)$. We divide this time into three parts: the time required to query the black-box of the input polynomial, time required to prepare black-boxes for sub-calls, and the time required to do everything else. The last of these is $\text{poly}(n)$ because the Algorithms 4, 5, 8, 7, and Algorithm 9 run in time $\text{poly}(n)$. This is so as all the operations that they perform like sparse polynomial interpolation, computing partial derivatives of order at most two, computing determinants of symbolic matrices, and factoring polynomials can be done efficiently in black-box fashion.

We now analyze how much time is required to query the black-box of the input polynomial and prepare black-boxes for sub-calls. To do this, let us understand how the black-boxes for the factors of the terms $\widehat{T}_1(A\mathbf{x}), \ldots, \widehat{T}_{s_2}(A\mathbf{x})$ are prepared in the for loop of lines 21-34 in first call to Find-Equivalence(), i.e., the call for $f$. Observe that for any $k \in [s_2]$, Compute-

Term-Black-Box() obtains black-box access to $\widehat{T}_k(A\mathbf{x})$ by setting all variables other than those in $\mathbf{z}_k$ to 0 in $f(A\mathbf{x})$ and subtracting a known constant $\beta$ from the resulting polynomial. Thus black-box access to $\widehat{T}_k(A\mathbf{x})$ is obtained by evaluating $f$ at known affine forms $\ell_1, \ldots, \ell_n$ (obtained from $A$ by setting variables not in $\mathbf{z}_k$ to 0) and subtracting a known constant $\beta$ from $f(\ell_1, \ldots, \ell_n)$.

For any $l \in [m_k]$, to obtain black-box access to the factors of $\widehat{T}_k(A\mathbf{x})$, we first compute a matrix $A_{k,0} \in \mathrm{GL}(|\mathbf{z}_k|, \mathbb{F})$ such that the factors $\widehat{Q}_1, \ldots, \widehat{Q}_{m_k}$ of $\widehat{T}_k(A(A_{k,0}\mathbf{z}_k, \mathbf{x} \setminus \mathbf{z}_k))$ are variable disjoint. To obtain black-box access to $\widehat{Q}_l$ for some $l \in [m_k]$, we first set the variables in $\mathbf{z}_k \setminus \mathbf{z}_{k,l}$ to random field elements $\mathbf{a}'$ and compute the constant $\beta_l$ from the (possibly inefficient) black-boxes of $\widehat{Q}_1, \ldots, \widehat{Q}_{m_k}$ obtained from $\widehat{T}_k(A(A_{k,0}\mathbf{z}, \mathbf{x} \setminus \mathbf{z}))$ using the black-box factorisation algorithm in [KT90]. We then compute $\beta_l^{-1} \cdot \widehat{T}\left(A_{k,0}\left(\mathbf{z}_{k,l}, \mathbf{z}_k \setminus \mathbf{z}_{k,l} = \mathbf{a}'\right)\right)$. Notice that this is the same as evaluating $f$ at known affine forms $\ell'_1, \ldots, \ell'_n$ (obtained from $\ell_1, \ldots, \ell_n$ by setting $\mathbf{z}_k \setminus \mathbf{z}_{k,l} = \mathbf{a}'$), multiplying it by a known constant $\beta_l^{-1}$ and subtracting a known constant $\beta_l^{-1}\beta$ from it.

In any recursive call to Find-Equivalence(), the black-boxes for sub-calls are prepared in the same way. Thus the discussion in the above paragraph implies that no matter the recursive depth for a recursive call for a polynomial $f'$, the black-box for $f'$ would look like $\alpha f(\ell_1, \ldots, \ell_n) - \beta$, where $\alpha, \beta$ are known constants and $\ell_1, \ldots, \ell_n$ known affine forms in $\mathrm{var}(f')$. Thus the time to query the black-box of $f'$ is $\mathrm{poly}(n)$; not $\mathrm{poly}(|\mathrm{var}(f')|)$, but still independent of the recursive depth. Similarly the time required to prepare black-boxes for sub-calls is also $\mathrm{poly}(n)$ and independent of the recursion depth as all that needs to be done is to compute appropriate affine forms $\ell'_1, \ldots, \ell'_n$ and constants $\alpha'$ and $\beta'$. Thus the algorithm runs in time $\mathrm{poly}(n)$.

## 5.6 ROF reconstruction

We present an algorithm that reconstructs an ROF in the PS-orb of a canonical ROF.[1] In this section, we slightly abuse the terminology and call an ROF that satisfies Properties 1-5 of Definition 2.23, but does not necessarily satisfy Property 6, a canonical ROF. We also give an accompanying algorithm to recover a translation vector and a scaling matrix that convert the reconstructed ROF to a canonical ROF. While randomized [HH91, BHH95] and deterministic [SV14, MV18] polynomial-time ROF reconstruction algorithms are known, we provide a randomized algorithm here as we need some special properties of this algorithm

[1] The algorithm can be easily adapted to work for a general ROF. However, since we only need to reconstruct ROFs in the PS-orb of a canonical ROF, we present the algorithm and its analysis just for ROFs in this form.

in Chapter 6.

### 5.6.1 The algorithm

Before formally describing the algorithm, let us see a high-level description of it.

**The idea.** Suppose that we have black-box access to an ROF $\mathtt{C} = T_1 + \cdots + T_s + \gamma$ in the PS-orb of a canonical ROF, where $T_k = Q_{k,1} \cdots Q_{k,s_k}$ for all $k \in [s]$, $Q_{k,l}$ is either a variable or a +-rooted sub-ROF of $\mathtt{C}$ for every $l \in [s_k]$, and $\gamma \in \mathbb{F}$. We first use the second-order derivatives of $\mathtt{C}$ to learn $\mathrm{var}(T_1), \ldots, \mathrm{var}(T_s)$. Then, we obtain black-box access to $T_1, \ldots, T_s$ as follows: As $\mathtt{C}$ is in the PS-orb of a canonical ROF, at most one of the $T_k$, say $T_s$, is a scalar multiple of a variable $x_i$. As $\frac{\partial \mathtt{C}}{\partial x_i x_j} = 0$ for all $j \neq i \in [n]$, we can find out $x_i$. On the other hand, for $k \in [s-1]$ and for a $x_i \in \mathrm{var}(Q_{k,l})$,

$$\frac{\partial \mathtt{C}}{\partial x_i} = r_1 \cdots r_m \cdot \prod_{l' \in [s_k] \setminus \{l\}} Q_{k,l'},$$

where $r_1, \cdots, r_m$ are pairwise variable disjoint, and every $r_i$ is either a variable or a $+$-rooted sub-ROF of $Q_{k,l}$. As $s_k \geq 2$ and $Q_{k,1}, \ldots, Q_{k,s_k}$ are non-constant polynomials, for every $Q_{k,l}$, there exists $x_j \in \mathrm{var}(T_k)$ such that $Q_{k,l}$ is an irreducible factor of $\frac{\partial \mathtt{C}}{\partial x_j}$ (because of Fact 2.1). Thus, by obtaining black-box access to the derivatives of $\mathtt{C}$ with respect to the variables in $\mathrm{var}(T_k)$, factoring them, collecting all the factors and then discarding a factor $r$ if there exists another factor $r'$ such that $\mathrm{var}(r) \subset \mathrm{var}(r')$, we get black-box access to $Q_{k,1}, \ldots, Q_{k,s_k}$ up to constant multiples. However, notice that if we want to query the black-box of a $Q_{k,l}$ at one point, we need to query the black-box of $\mathtt{C}$ at $\mathrm{poly}(n)$ points. So, if we try to recursively learn $Q_{k,1}, \ldots, Q_{k,s_k}$, the running time of the algorithm would be exponential in the depth of $\mathtt{C}$.

We need to be able to get black-box access to $Q_{k,1}, \ldots, Q_{k,s_k}$ in such a way that to obtain the value of $Q_{k,l}$ at one point, we only need to query the black-box of $\mathtt{C}$ at one point. We do this by first learning $\mathrm{var}(Q_{k,1}), \ldots, \mathrm{var}(Q_{k,s_k})$ and some roots of $Q_{k,1}, \ldots, Q_{k,s_k}$ using the black-boxes of $Q_{k,1}, \ldots, Q_{k,s_k}$ that we obtained above. Then we set all variables in $\mathbf{x} \setminus \mathrm{var}(Q_{k,l})$ to random field elements. This gives us black-box access to $c_{k,l} Q_{k,l} + c'_{k,l}$, for some unknown $c_{k,l} \neq 0, c'_{k,l} \in \mathbb{F}$. Plugging in the root of $Q_{k,l}$ into this black-box we learn $c'_{k,l}$. Subtracting this from $c_{k,l} Q_{k,l} + c'_{k,l}$ gives us black-box access to $c_{k,l} Q_{k,l}$, where $c_{k,l}$ is unknown. Notice that now we only need to make one query to the black-box of $\mathtt{C}$ to learn the value of $c_{k,l} Q_{k,l}$.

We learn $\gamma$ by finding a common root $\mathbf{a} = (a_1, \ldots, a_n)$ of $Q_{k,l}$ for all $k \in [s], l \in [s_k]$ and setting $\gamma = \mathtt{C}(\mathbf{a})$. Then, we find out $c_1 \ldots, c_s \in \mathbb{F}$ such that $\sum_{k=1}^{s} c_k \cdot \prod_{l=1}^{s_k} c_{k,l} \cdot Q_{k,l} = \mathtt{C} - \gamma$

and multiply $Q_{1,1}, \ldots, Q_{s,1}$ by $c_1, \ldots, c_s$, respectively. After that, we learn $T_k$ by recursively learning $c_k c_{k,1} Q_{k,1}, c_{k,2} Q_{k,2}, \ldots, c_{k,s_k} Q_{k,s_k}$ and multiplying them. We now describe the algorithm formally.

---

**Algorithm 9** Reconstruct-ROF($f(\mathbf{x})$)

---

**Input:** Black-box access to an $n$-variate ROF $f(\mathbf{x})$ in the PS-orb of a canonical ROF.

**Output:** An ROF $\mathsf{C}$ in the PS-orb of a canonical ROF computing $f$.

/* Learning $\text{var}(T_1), \ldots, \text{var}(T_s)$. */

Let $E \leftarrow \emptyset$, and $G \leftarrow (\mathbf{x}, E)$ be an undirected graph.

**for** $i, j \in [|\mathbf{x}|]$ **do**

  If $\frac{\partial^2 f}{\partial x_i \partial x_j} \neq 0$, add edge $\{x_i, x_j\}$ to $E$.

**end for**

Let $\mathcal{C} \leftarrow \{\mathbf{x}_1, \ldots, \mathbf{x}_s\}$ be the set of connected components of $G$, where $s \leftarrow |\mathcal{C}|$.

/* Discovering factors of $T_1, \ldots, T_s$. */

**if** $\exists k \in [s]$ such that $|\mathbf{x}_k| = 1$ **then**

  If $\mathbf{x}_k = \{x_i\}$, $T_k \leftarrow x_i$, $N_k \leftarrow \{x_i\}$.

**end if**

**for** $k \in [s]$ such that $|\mathbf{x}_k| \geq 2$ **do**

  $N_k \leftarrow \emptyset$.

  **for** $i$ such that $x_i \in \mathbf{x}_k$ **do**

    Compute black-box access to $\frac{\partial f}{\partial x_i}$ and then obtain black-box access to all its irreducible factors. Add all the irreducible factors to $N_k$.

  **end for**

  **for** $r_1, r_2 \in N_k$ **do**

    If $\text{var}(r_1) \subseteq \text{var}(r_2)$, $N_k \leftarrow N_k \setminus \{r_1\}$. Else, if $\text{var}(r_2) \subseteq \text{var}(r_1)$, $N_k \leftarrow N_k \setminus \{r_2\}$.

  **end for**

**end for**

/* Obtaining efficient black-box access to the factors of $T_1, \ldots, T_s$. */

**for** $k \in [s]$ such that $|\mathbf{x}_k| \geq 1$ **do**

  **for** $r \in N_k$ **do**

    $\mathbf{a}_r \leftarrow$ a vector of size $|\text{var}(r)|$ which is a root of $r$. $\mathbf{a}'_r \leftarrow$ a random vector of size $n - |\text{var}(r)|$.

    $\beta_r \leftarrow f(\text{var}(r) = \mathbf{a}_r, \mathbf{x} \setminus \text{var}(r) = \mathbf{a}'_r)$. $r \leftarrow f(\text{var}(r), \mathbf{x} \setminus \text{var}(r) = \mathbf{a}'_r) - \beta_r$.

  **end for**

27: **end for**

28: /* Learning $\gamma$ and $c_1, \ldots, c_s$. */

29: Construct $\mathbf{a} = (a_1, \ldots, a_n)$, a common root of $\prod_{r \in N_1} r, \ldots, \prod_{r \in N_s} r$. $\gamma \leftarrow f(\mathbf{a})$.

30: Solve for $c_1, \ldots, c_s$ such that $f - \gamma = c_1 \cdot \prod_{r \in N_1} r + \cdots + c_s \cdot \prod_{r \in N_s} r$.

31: For all $k \in [s]$, replace an arbitrary $r \in N_k$ by $c_k \cdot r$. If $\exists k \in [s]$ such that $|\mathbf{x}_k| = 1$, $T_k \leftarrow c_k T_k$.

32: /* Reconstructing $T_1, \ldots, T_s$. */

33: **for** $k \in [s]$ such that $|\mathbf{x}_k| \geq 2$ **do**

34: $\quad T_k \leftarrow 1$.

35: $\quad$ **for** $r \in N_k$ **do**

36: $\quad\quad \mathbf{y} \leftarrow \text{var}(r)$. $T_k \leftarrow T_k \times$Reconstruct-ROF$(r(\mathbf{y}))$.

37: $\quad$ **end for**

38: **end for**

39: $\mathtt{C} \leftarrow T_1 + \cdots + T_s + \gamma$. Return $\mathtt{C}$.

---

### 5.6.2 Analysis of the algorithm

We will assume, without loss of generality, that an ROF has no edge labels and every leaf node is either a constant or a constant multiple of a variable.

**Lemma 5.7** *If $f(\mathbf{x})$ computed by an ROF $\mathtt{C}'$ in the* PS-orb *of a canonical ROF, then the ROF $\mathtt{C}$ returned by the algorithm is equal to $\mathtt{C}'$ up to scaling of the leaves with high probability. Moreover, there is a one-to-one correspondence between the gates of $\mathtt{C}$ and the gates of $\mathtt{C}'$ with a gate of $\mathtt{C}$ computing a non-zero constant multiple of the polynomial computed by the corresponding gate of $\mathtt{C}'$.*

**Proof:** We induct on the product-depth $\Delta$ of $\mathtt{C}'$. If $\Delta = 0$, then as $\mathtt{C}'$ is in the PS-orb of a canonical ROF, $\mathtt{C}' = c_i x_i + \gamma$, where $c_i \neq 0, \gamma \in \mathbb{F}$. In this case $C$ has only one connected component $\mathbf{x}_1 = \{x_i\}$. In Step 10, $T_1 = x_i$. Step 29 can be implemented by simply setting $\mathbf{a}$ to be the all zero vector and Step 30 by computing $\frac{\partial f}{\partial x_i}$. Thus, $\mathtt{C} = \mathtt{C}'$ and for $\Delta = 0$, the lemma is true.

Now, suppose that the lemma is true for all ROFs of product-depth at most $\Delta \geq 0$ and let $\mathtt{C}'$ be a product-depth $\Delta + 1$ ROF. Let $\mathtt{C}' = T'_1 + \cdots + T'_s + \gamma'$. There exists at most one $k' \in [s]$, such that $|\text{var}(T'_{k'})| = 1$ and for every $k \in [s] \setminus \{k'\}$, let $T'_k = Q'_{k,1} \cdots Q'_{k,s_k}$, where $s_k \geq 2$ and for every $l \in [s_k]$, $Q'_{k,l}$ is either a variable or a +-rooted sub-ROF of $\mathtt{C}'$.

**Claim 5.29** *There is a bijection $\pi : [s] \to [s]$ s.t. the connected component $\mathbf{x}_{\pi(k)} = \mathrm{var}(T'_k)$ for all $k \in [s]$.*

**Proof:** Fix any $k \in [s]$. If $|\mathrm{var}(T'_k)| = 1$, say $T'_k = c \cdot x_i$ for some $c \in \mathbb{F}^\times$, then $\frac{\partial^2 f}{\partial x_i \partial x_j} = 0$ for all $j \in [s] \setminus \{i\}$, and so, $\{x_i\}$ is a connected component in $\mathcal{C}$. If $|\mathrm{var}(T'_k)| \geq 2$, then as $\mathbb{C}'$ is in the PS-orb of a canonical ROF, $T'_k = Q'_{k,1} \cdots Q'_{k,s_k}$, where $s_k \geq 2$. If $x_i, x_j \in \mathrm{var}(T'_k)$ are such that $x_i \in \mathrm{var}(Q'_{k,l})$ and $x_j \in \mathrm{var}(Q'_{k,l'})$, for $l \neq l'$, then as $\frac{\partial^2 f}{\partial x_i \partial x_j} \neq 0$, $\{x_i, x_j\} \in E$. On the other hand, if $l = l'$, then as $s_k \geq 2$ and $Q'_{k,1}, \ldots, Q'_{k,s_k}$ are non-constant, there exists a $x_m \in \mathrm{var}(Q'_{k,l''})$, $l'' \neq l$ such that $\frac{\partial^2 f}{\partial x_i \partial x_m} \neq 0$ and $\frac{\partial^2 f}{\partial x_j \partial x_m} \neq 0$. Thus, $\{x_i, x_m\}, \{x_j, x_m\} \in E$ and so $x_i$ and $x_j$ are in the same connected component. Moreover, as for any $x_i \in \mathrm{var}(T'_k)$ and $x_j \in \mathrm{var}(T'_1) \uplus \cdots \uplus \mathrm{var}(T'_{k-1}) \uplus \mathrm{var}(T'_{k+1}) \uplus \cdots \uplus \mathrm{var}(T'_s)$, $\frac{\partial^2 f}{\partial x_i \partial x_j} = 0$, this connected component is exactly $\mathrm{var}(T'_k)$, proving the claim. □

**Claim 5.30** *After Steps 9-11 and the for loop of lines 22-27 have been executed, for all $k \in [s]$, with high probability, $N_{\pi(k)} = \{Q_{k,1}, ..., Q_{k,s_k}\}$ where $Q_{k,l}$ is a non-zero constant multiple of $Q'_{k,l}$ for all $l \in [s_k]$ and $\pi$ is the bijection given in Claim 5.29.*

**Proof:** Fix any $k \in [s]$. If $|\mathrm{var}(T'_k)| = 1$, say $\mathrm{var}(T'_k) = \{x_i\}$, then Claim 5.29 immediately implies that after Steps 9-11 have been executed, $N_{\pi(k)} = \{x_i\}$. On the other hand if $|\mathrm{var}(T'_k)| \geq 2$, then observe that for any $x_i \in \mathrm{var}(Q'_{k,l})$,

$$\frac{\partial \mathbb{C}'}{\partial x_i} = r_1 \cdots r_m \cdot \prod_{l' \in [s_k] \setminus \{l\}} Q'_{k,l'},$$

where $r_1, \cdots, r_m$ are pairwise variable disjoint and every $r_i$ is a variable or a $+$-rooted sub-ROF of $Q'_{k,l}$. Hence, after the for loop 14-16 has been executed, $N_{\pi(k)}$ will contain two types of factors:

- constant multiples of $Q'_{k,1}, \ldots, Q'_{k,s_k}$ and
- constant multiples of $+$-rooted sub-ROFs of $Q'_{k,1}, ..., Q'_{k,s_k}$.

The first kind of factors are present because $s_k \geq 2$, $Q'_{k,1}, ..., Q'_{k,s_k}$ are non-constant polynomials and being $+$-rooted sub-ROFs are irreducible (see Fact 2.1). As all variables appearing in any $+$-rooted sub-ROF of $Q'_{k,l}$ are also variables of $Q'_{k,l}$ , the second kind of factors are removed from $N_{\pi(k)}$ by the for loop of lines 17-19. Moreover, as $Q'_{k,l}$ and $Q'_{k,l'}$ are variable disjoint for $l \neq l'$, the first kind of factors are not removed. This means that after the for

loop of lines 12-20 has been executed, for all $k \in [s]$, $N_{\pi(k)} = \{Q_{k,1}, ..., Q_{k,s_k}\}$ where $Q_{k,l}$ is a non-zero constant multiple of $Q'_{k,l}$ for all $l \in [s_k]$. Thus, a root of $Q_{k,l}$ is also a root of $Q'_{k,l}$.

Now, inside the loop of lines 22-27, for $r = Q_{k,l}$, $f\left(\text{var}(r),\ \mathbf{x} \setminus \text{var}(r) = \mathbf{a}'_r\right) = c_{k,l}Q'_{k,l} + c'_{k,l}$ for some $c_{k,l}, c'_{k,l} \in \mathbb{F}$. As every coordinate of $\mathbf{a}'_r$ is chosen randomly (say, from a subset of $\mathbb{F}$ of size $n^4$), with high probability $c_{k,l} \neq 0$. As $\mathbf{a}_r$ is a root of $Q'_{k,l}$ ,

$$\beta_r = f\left(\text{var}(r) = \mathbf{a}_r,\ \mathbf{x} \setminus \text{var}(r) = \mathbf{a}'_r\right) = c'_{k,l}.$$

Hence, after this loop has been executed $r = c_{k,l}Q'_{k,l}$ , proving the claim. □

Thus, for all $k \in [s]$, $T'_k$ is a non-zero constant multiple of the product of the polynomials in $N_{\pi(k)}$. So, for some $c_1, \ldots, c_s \in \mathbb{F}^\times$, $\mathtt{C}' = T'_1 + \cdots T'_s + \gamma' = \sum_{k\in[s]} c_k \cdot \prod_{l\in[s_k]} Q_{k,l} + \gamma$. Since in Step 29, $\mathbf{a}$ is a common root of $Q_{k,l}$, for all $k \in [s]$ and $l \in [s_k]$, $\gamma' = f(\mathbf{a}) = \gamma$. As the polynomials in $\left\{\prod_{l\in[s_k]} Q_{k,l} : k \in [s]\right\}$ are linearly independent, $c_1, \ldots, c_s$ are unique. Once $c_1, \ldots, c_s$ have been learnt in Step 30 and $N_1, \ldots, N_s$ updated in Step 31, we have for all $k \in [s]$, $T'_k$ is equal to the product of the polynomials in $N_{\pi(k)}$. For all $k \in [s]$ and $l \in [s_k]$, as $Q_{k,l}$ is a product-depth $\Delta$ ROF in the PS-orb of a canonical ROF, from the induction hypothesis, the output of Reconstruct-ROF$(Q_{k,l})$ is a ROF in the PS-orb of a canonical ROF and is equal to $Q_{k,l}$ up to scaling of the leaves. After the loop 33-38 has been executed, for all $k \in [s]$, $T_{\pi(k)}$ is an ROF in the PS-orb of a canonical ROF and is equal to $T'_k$ up to scaling of the leaves. Hence, $\mathtt{C}' = T'_1 + \cdots + T'_s + \gamma' = T_1 + \cdots + T_s + \gamma = \mathtt{C}$.

For the "moreover" part of the lemma, notice that from the induction hypothesis, we have the desired one-to-one correspondence between gates of the ROF output by Reconstruct-ROF$(Q_{k,l})$ and the gates of $Q_{k,l}$ for all $k \in [s]$ and $l \in [s_k]$. Then, as $T_{\pi(k)} = T'_k$ , this yields the desired one-to-one correspondence between the gates of $\mathtt{C}'$ and $\mathtt{C}$. □

**Running time of the algorithm**

We will show that the algorithm runs in time poly$(n)$. From black-box access to $f$, a black-box access to $\frac{\partial^2 f}{\partial x_i \partial x_j}$, for any $i, j \in [n]$, can be computed in poly$(n)$ time (Fact 5.1). Whether $\frac{\partial^2 f}{\partial x_i \partial x_j}$ is zero or not can be determined in poly$(n)$ time using the Schwartz-Zippel test. Hence $G$ can be constructed in time poly$(n)$. The connected components of $G$ can also be computed in poly$(n)$ time. Clearly, lines 9-11 run in poly$(n)$ time. Now we analyse the runtime of the loop of lines 12-20.

A black-box access to $\frac{\partial f}{\partial x_i}$ can be obtained in poly$(n)$ time from black-box access to $f$. Once we have black-box access to $\frac{\partial f}{\partial x_i}$, black-box access to its irreducible factors can be computed

in poly$(n)$ time using the algorithm in [KT90]. Hence, the for loop of lines 14-16 executes in poly$(n)$ time. For Step 18, var$(r_1)$ and var$(r_2)$ can be determined by obtaining black-box access to the derivatives of $r_1$ and $r_2$ with respect to all $\mathbf{x}$ variables and checking using the Schwartz-Zippel lemma which of them are non-zero. Thus this step, and hence, the for loop of lines 17-19 executes in poly$(n)$ time.

Once we have black-box access to all polynomials in $N_1, \ldots, N_s$, for all $k \in [s]$ and all $r \in N_k$, $\mathbf{a}'_r$ and $\mathbf{a}_r$ can be computed in poly$(n)$ time. To construct $\mathbf{a}_r$, first set all but one variable appearing in $r$ to random values. After doing this, $r$ becomes an affine form whose root can be computed easily. Notice that on line 25, we obtain black-box access to $Q_{k,l}$ using *only one* query to $f$.

The vector $\mathbf{a}$ can be constructed in poly$(n)$ time by just combining all $\mathbf{a}_r$'s constructed in the loop of lines 22-27. To compute $c_1, \ldots, c_s$ in Step 30, we can simply evaluate $f$ and $\prod_{l \in s_k} Q_{k,l}$ for all $k \in [s]$ at $s$ many random points $\mathbf{b}_1, \ldots \mathbf{b}_s$ and solve the linear system of equations

$$\left\{ f(\mathbf{b}_i) - \gamma = \sum_{k \in [s]} c_k \cdot \prod_{l \in s_k} Q_{k,l}(\mathbf{b}_i) : k \in [s] \right\}$$

for $c_1, \ldots, c_s$. As $\left\{ \prod_{l \in [s_k]} Q_{k,l} : k \in [s] \right\}$ are linearly independent, with high probability, the coefficient matrix of this system will be invertible.

So far we have shown that for each call to the algorithm, the time spent outside the recursive calls on line 36 is poly$(n)$. Now, given input $f$, the total number of recursive calls is at most poly$(n)$. This is because each leaf of the recursion tree corresponds to a distinct variable in $\mathbf{x}$ and whenever Reconstruct-ROF$(r)$ is called from inside Reconstruct-ROF$(r')$, var$(r) \subsetneq$ var$(r')$. Thus, the runtime of the algorithm is poly$(n)$, as a black-box query to $r$ amounts to only one query to $f$.

### 5.6.3 Canonization: Recovering scaling and translation

We shall slightly abuse the terminology in this section and say that a leaf of an ROF is a variable if it is a constant multiple of a variable. This is consistent with our assumption in the previous section that any leaf of an ROF is either a constant multiple of a variable or a constant. To recover the scaling matrix and the translation vector, we use the following algorithm.

**Algorithm 10** Canonize$(\mathbb{C})$

**Input:** An ROF $\mathtt{C}$ in the PS-orb of a canonical ROF.
**Output:** A scaling matrix $S \in \mathrm{GL}(|\mathbf{x}|, \mathbb{F})$ and a vector $\mathbf{b}$ such that $\mathtt{C}(S\mathbf{x}+\mathbf{b})$ is a canonical ROF.

1: $N_1 \leftarrow$ set of all $+$-gates in $\mathtt{C}$ directly connected to a variable. $N_2 \leftarrow$ set of all variable leaves connected to $\times$-gates. Initialize $S \leftarrow I_{n\times n}$ and $\mathbf{b} = (b_1, \ldots, b_n) = \mathbf{0}$.
2: **for** $v \in N_1 \uplus N_2$ **do**
3: If $v \in N_1$ and the variable and constant children of $v$ are $\alpha_i x_i$ and $\beta_i$ respectively, where $\alpha_i \in \mathbb{F}^\times$, then $b_i \leftarrow \frac{-\beta_i}{\alpha_i}$ and $S \leftarrow \mathrm{diag}\left(0, \ldots, 0, \alpha_i^{-1}, 0, \ldots, 0\right) \cdot S$.
4: Else, if $v \in N_2$ and $v = \alpha_i x_i$, $\alpha_i \in \mathbb{F}^\times$, $S \leftarrow \mathrm{diag}\left(0, \ldots, 0, \alpha_i^{-1}, 0, \ldots, 0\right) \cdot S$ and $b_i \leftarrow 0$.
5: **end for**
6: Return $S$, $\mathbf{b}$.

Clearly, the algorithm runs in $\mathrm{poly}(n)$ time; its correctness follows from the next observation.

**Observation 5.25** *Let $S, \mathbf{b} = \mathit{Canonize}(\mathtt{C})$. Then, $\mathtt{C}(S\mathbf{x}+\mathbf{b})$ is a canonical ROF.*

**Proof:** Let $v \in N_1$ and let the variable and constant children of $v$ be $\alpha_i x_i$ and $\beta_i$, respectively. As $b_i = \frac{-\beta_i}{\alpha_i}$ and $S$ is updated as $\mathrm{diag}\left(0, \ldots, 0, \alpha_i^{-1}, 0, \ldots, 0\right) \cdot S$, after the execution of the loop of lines 2-5, $\alpha_i x_i + \beta_i$ from $\mathtt{C}$ becomes $x_i$ in $\mathtt{C}(S\mathbf{x}+\mathbf{b})$. Similarly, if $v \in N_2$ and $v = \alpha_i x_i$, because $S$ is updated as $\mathrm{diag}\left(0, \ldots, 0, \alpha_i^{-1}, 0, \ldots, 0\right) \cdot S$, after the execution of the loop of lines 2-5, $\alpha_i x_i$ becomes $x_i$ in $\mathtt{C}(S\mathbf{x}+\mathbf{b})$. Since the only difference between a canonical ROF and an ROF in its PS-orb is that in the latter a variable can be scaled and translated, this proves the observation. □

We now show that not only does Algorithm 10 recover the translation vector but also that this vector is recovered uniquely. The following claim comes in handy in Section 6.3.

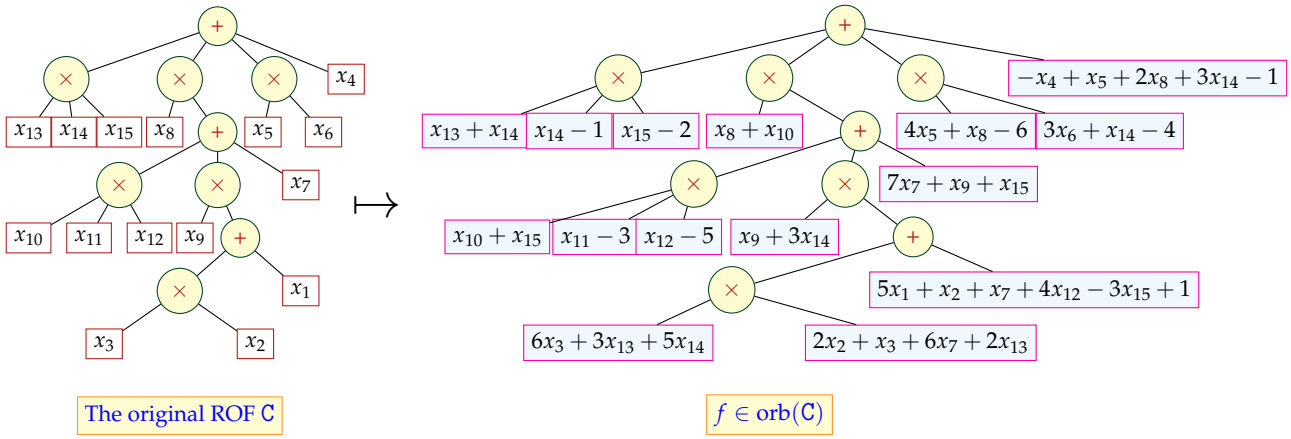
**Claim 5.31** *Let $\mathtt{C}'$ be an ROF in the* PS-orb *of a canonical ROF, $S'$ be a scaling matrix and $\mathbf{b}'$ a translation vector such that $\mathtt{C}'(S'\mathbf{x}+\mathbf{b}')$ is a canonical ROF. Also, let $\mathtt{C} = \mathit{Reconstruct\text{-}ROF}(\mathtt{C}'(\mathbf{x}))$ and $S, \mathbf{b} = \mathit{Canonize}(\mathtt{C})$. Then, $\mathbf{b} = \mathbf{b}'$.*

**Proof:** Let $v \in N_1 \uplus N_2$. From Lemma 5.7, the corresponding gate $v'$ in $\mathtt{C}'$ is such that $v = c \cdot v'$ for some $c \neq 0$. So, if $v \in N_1$, then $v'$ must be a $+$ gate with variable and constant children. If the variable and constant children of $v'$ are $\alpha_i' x_i$ and $\beta_i'$, respectively, then the variable and the constant children of $v$ are $c\alpha_i' x_i$ and $c\beta_i'$, respectively. Observe that $b_i' = \frac{-\beta_i'}{\alpha_i'}$.

Thus, $b_i = \frac{-c\beta'_i}{c\alpha'_i} = \frac{-\beta'_i}{\alpha'_i} = b'_i$. Similarly, if $v \in N_2$, then $v'$ is also a variable leaf. Thus, $b_i = b'_i = 0$. □

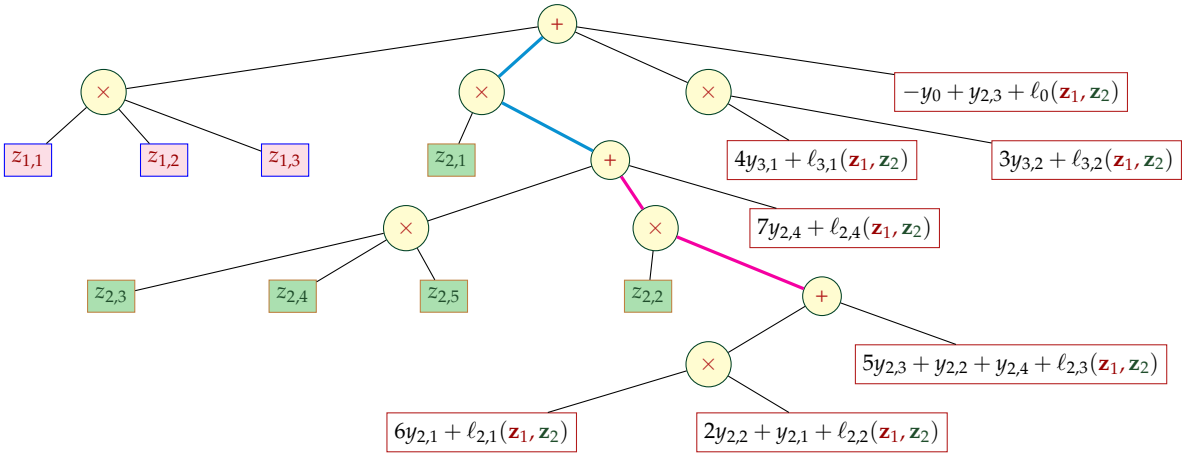
## 5.7 A pictorial overview of Algorithm 3

We consider the following simple example to show the working of Phase 1 of the algorithm mentioned in Section 5.5.1.




The original ROF $\mathbb{C}$ $\quad$ $f \in \text{orb}(\mathbb{C})$

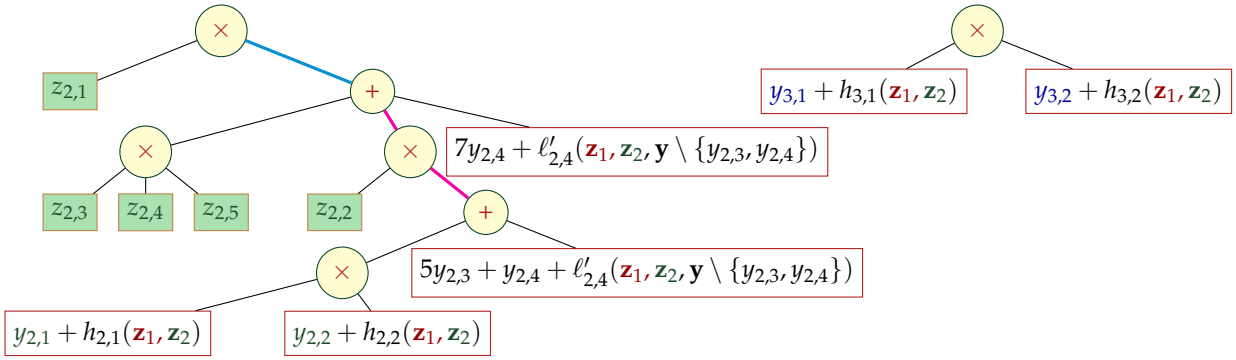
In the following figures, $\ell_i, \ell_{i,j}, \ell'_{i,j}, h_i, h'_i, h_{i,j}, h'_{i,j}$ etc. denote affine forms.




Step 1

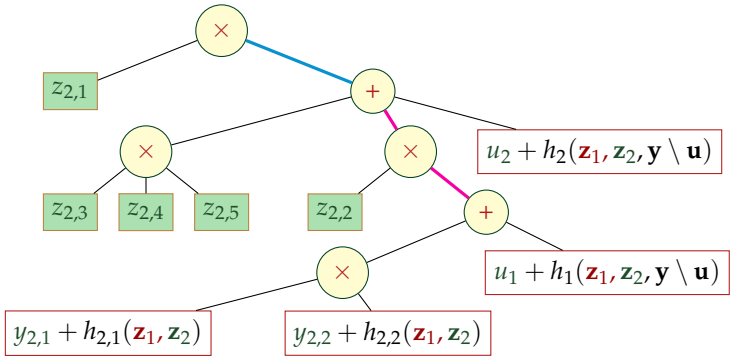
In Step 1, all the good terms of $f$ are made variable disjoint. Furthermore, every "good" linear factor - affine form connected to a $\times$ gate computing a polynomial of degree at least 3

in $f$ - gets mapped to (a constant multiple) of a distinct variable. In our example, the good term has three good linear factors, which have been mapped to $z_{1,1}, z_{1,2}, z_{1,3}$. Also, there are five good linear factors in the bad term; these have been mapped to $z_{2,1}, \ldots, z_{2,5}$. Let $\mathbf{z}_1 := \{z_{1,1}, z_{1,2}, z_{1,3}\}$, $\mathbf{z}_2 := \{z_{2,1}, \ldots, z_{2,5}\}$, $\mathbf{z} := \mathbf{z}_1 \uplus \mathbf{z}_2$, and $\mathbf{y} := \mathbf{x} \setminus \mathbf{z}$. Step 2 extensively uses skewed paths. In the above figure, there are two skewed paths identified by the "marker monomials" $z_{2,1}$ and $z_{2,1}z_{2,2}$.




Step 2.1

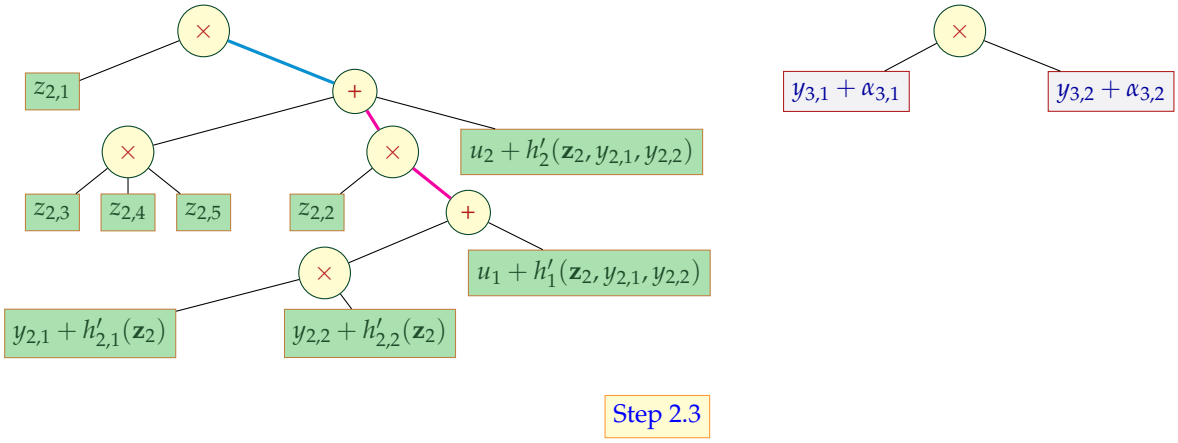
In Step 2.1, every affine form corresponding to a variable in the top quadratic form or a quadratic form along a skewed path which is redundant for $\det(H_C)$ is mapped to an affine form of the type $y_{i,j} + h_{i,j}(\mathbf{z})$. In the above figure, the $\mathbf{y}$-variables corresponding to affine forms in the top quadratic form and in the quadratic form along the skewed path $z_{2,1}z_{2,2}$ are $y_{3,1}, y_{3,2}$ and $y_{2,1}, y_{2,2}$, respectively. We shall refer to all the remaining $\mathbf{y}$-variables, i.e., $y_0, y_{2,3}, y_{2,4}$ as $\mathbf{u}$-variables.



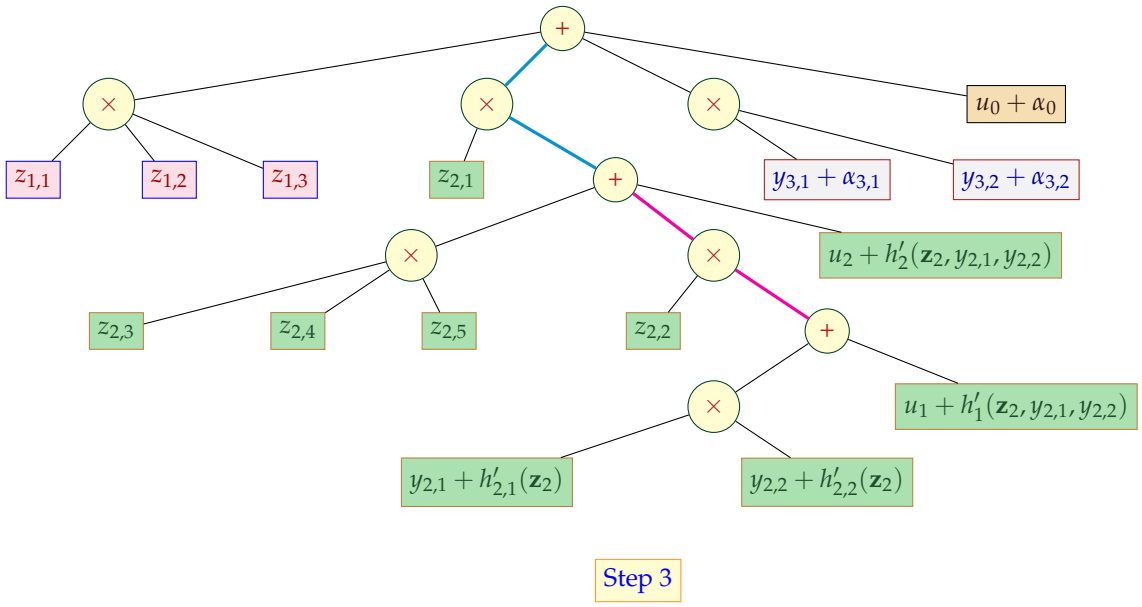

Step 2.2

In Step 2.2, every affine form corresponding to a dangling variable along a skewed path which is redundant for $\det(H_C)$ is mapped to an affine form of the type $u_i + h_i(\mathbf{z}, \mathbf{y} \setminus \mathbf{u})$.

There might be dangling variables along skewed paths that are present in a set of essential variables of $\det(H_C)$. In our simple example, such variables are not present. In the general case, such variables can be handled by picking a basis of an appropriate vector space. This space is spanned by the $\mathbf{u}$-parts of the affine forms corresponding to the dangling variables along skewed paths and the top dangling variable (see Section 5.5.1 for more details). A word of caution: the affine form corresponding to the top dangling variable has not been handled; it will be fixed in Step 3. Let $\mathbf{y}_2 = \{u_1, u_2, y_{2,1}, y_{2,2}\}$. In the above figure, the variables in $\mathbf{z}_1$ and $\mathbf{y} \setminus \mathbf{y}_2$ are external for the bad term and all variables in $\mathbf{x} \setminus \{y_{3,1}, y_{3,2}\}$ are external for the top quadratic form. These will be removed in Step 2.3.




In Step 2.3, external variables are removed from the affine forms in the top quadratic form, quadratic forms along skewed paths and corresponding to dangling variables along skewed paths. In the above figure, $h'_{i,j}$ and $h'_i$ are obtained after removing external variables from $h_{i,j}$ and $h_i$, respectively.




In Step 3, the affine form corresponding to the top-most dangling variable is mapped to $u_0 + \alpha_0, \alpha_0 \in \mathbb{F}$. Now, all the terms of $f$ are variable disjoint. After this, we recursively call Algorithm 3 on the factors of the good and the bad terms to map them to variable disjoint ROFs.

# Chapter 6

# Polynomial equivalence for orbits of read-once arithmetic formulas

This chapter gives polynomial equivalence algorithms for orbits of read-once arithmetic formulas in various special cases. The contents of this chapter are from a joint work with Nikhil Gupta and Chandan Saha [GST23].

## 6.1 Introduction

The polynomial equivalence problem for a circuit class $\mathcal{C}$ is as follows: given $f, g \in \mathbb{F}[\mathbf{x}]$ computable by circuits in $\mathcal{C}$, check if $g \sim f$. If yes, find $A \in \mathrm{GL}(|\mathbf{x}|, \mathbb{F})$ and $\mathbf{b} \in \mathbb{F}^{|\mathbf{x}|}$ such that $g(\mathbf{x}) = f(A\mathbf{x} + \mathbf{b})$. This section is devoted to designing and analysing randomised polynomial time algorithms for various special cases of the polynomial equivalence problem for the orbit of ROFs (see Definition 2.4). In particular, we prove the following theorems.

**Theorem 1.7 (PE for orbits of additive-constant-free ROFs)** *Let $n \in \mathbb{N}$, $\mathrm{char}(\mathbb{F}) = 0$ or $\geq n^2$, and $|\mathbb{F}| \geq n^{13}$. There is a $\mathrm{poly}(n)$ time randomized algorithm (with oracle access to QFE over $\mathbb{F}$) that takes input black-box access to two $n$-variate polynomials $f_1, f_2 \in \mathbb{F}[\mathbf{x}]$, which are in the orbits of two unknown additive-constant-free canonical ROFs, and checks if $f_1 \in \mathrm{orb}(f_2)$. Furthermore, if $f_1 \in \mathrm{orb}(f_2)$, then the algorithm outputs (with high probability) an $A \in \mathrm{GL}(n, \mathbb{F})$ and a $\mathbf{b} \in \mathbb{F}^n$ such that $f_1 = f_2(A\mathbf{x} + \mathbf{b})$.*

**Theorem 1.8 (PE for orbits of product-depth 2 ROFs)** *Let $n \in \mathbb{N}$, $\mathrm{char}(\mathbb{F}) = 0$ or $\geq n^2$, and $|\mathbb{F}| \geq n^{13}$. There is a $\mathrm{poly}(n)$ time randomized algorithm (with oracle access to QFE over $\mathbb{F}$) that takes input black-box access to two $n$-variate polynomials $f_1, f_2 \in \mathbb{F}[\mathbf{x}]$, which are in the orbits of two unknown canonical ROFs with product-depth 2, and checks if $f_1 \in \mathrm{orb}(f_2)$. Furthermore, if*

$f_1 \in \text{orb}(f_2)$, *then the algorithm outputs (with high probability) an* $A \in \text{GL}(n, \mathbb{F})$ *and a* $\mathbf{b} \in \mathbb{F}^n$ *such that* $f_1 = f_2(A\mathbf{x} + \mathbf{b})$.

Recall that Quadratic Form Equivalence can be solved efficiently over $\mathbb{C}, \mathbb{R}, \mathbb{F}_q$ and also over $\mathbb{Q}$ with oracle access to integer factoring (see Fact 5.3). Hence, PE for the orbits of additive constant free ROFs as well as for the orbit of depth 5 ROFs can also be solved efficiently over these fields.

Theorem 1.7 is proved in the following section and Theorem 1.8 is proved in Section 6.3.

## 6.2 Polynomial Equivalence for orbits of additive constant free ROFs

In this section, we shall prove Theorem 1.7. Let $\text{ROF}_0$ be the class of all additive-constant-free canonical ROFs. We make use of an efficient algorithm for rooted tree isomorphism in our proof, so we first define rooted tree isomorphism.

### 6.2.1 Rooted tree isomorphism

**Definition 6.1 (Tree isomorphism)** *Two rooted trees* $G_1 = (V_1, E_1, v_1)$ *and* $G_2 = (V_2, E_2, v_2)$ *are* isomorphic, *if there is a bijection* $\pi : V_1 \to V_2$ *s.t.* $(v, v') \in E_1 \Leftrightarrow (\pi(v), \pi(v')) \in E_2$ *and* $\pi(v_1) = v_2$.

**Fact 6.1 (Efficient tree isomorphism [AHU83])** *There is an algorithm that takes input two rooted trees* $G_1 = (V_1, E_1, v_1)$ *and* $G_2 = (V_2, E_2, v_2)$ *and decides if* $G_1$ *and* $G_2$ *are isomorphic. If the answer is yes, it also outputs an isomorphism. The running time of the algorithm is* $\text{poly}(|V_1|, |V_2|)$.

The following observation is easy to show.

**Observation 6.1** *Let* $G_1 = (V_1, E_1, v_1)$ *and* $G_2 = (V_2, E_2, v_2)$ *be rooted trees and* $\pi$ *an isomorphism from* $G_1$ *to* $G_2$ *s.t.* $\pi(v_1) = v_2$. *Then,* $\pi$ *is completely determined by its restriction to the leaves of* $G_1$.

### 6.2.2 The algorithm

The following algorithm decides whether $f_1(\mathbf{x}), f_2(\mathbf{x}) \in \text{orb}(\text{ROF}_0)$ are equivalent or not.

**Algorithm 11** Equivalence-Test$(f_1(\mathbf{x}), f_2(\mathbf{x}))$

**Input:** Black-box access to $f_1(\mathbf{x}), f_2(\mathbf{x}) \in \text{orb}(\text{ROF}_0)$.

**Output:** Whether or not $f_1$ and $f_2$ are equivalent. If they are equivalent, then $A \in \text{GL}(n, \mathbb{F})$ and $\mathbf{b} \in \mathbb{F}^n$ such that $f_1(\mathbf{x}) = f_2(A\mathbf{x} + \mathbf{b})$.

1: /* Reconstructing canonical ROFs equivalent to $f_1$ and $f_2$. */

2: **for** $i \in [2]$ **do**

3: $A_i \leftarrow$ Find-Equivalence$(f_i(\mathbf{x}))$ (Algorithm 3).

4: $\mathtt{C}'_i \leftarrow$ Reconstruct-ROF$(f_i(A_i\mathbf{x}))$ (Algorithm 9).

5: $S_i, \mathbf{b}_i \leftarrow$ Canonize$(\mathtt{C}'_i)$ (Algorithm 10), where $S_i \in \mathrm{GL}(n, \mathbb{F})$ is a scaling matrix, $\mathbf{b}_i \in \mathbb{F}^n$.

6: $\mathtt{C}_i \leftarrow \mathtt{C}'_i(S_i\mathbf{x} + \mathbf{b}_i)$, $G_i \leftarrow$ the underlying tree of $\mathtt{C}_i$ wherein all internal nodes are unlabelled and the leaves are labelled by variables.

7: **end for**

8:

9: /* Checking if $\mathtt{C}_1$ and $\mathtt{C}_2$ are equivalent */

10: **if** $G_1$ and $G_2$ are isomorphic as rooted trees **then**

11: If $\sigma$ is the permutation such that $\sigma(G_2) = G_1$, construct a permutation matrix $P$ that maps $x_i$ to $\sigma(x_i)$ $\forall i \in [n]$ using Fact 6.1. $A \leftarrow A_2S_2PS_1^{-1}A_1^{-1}$, $\mathbf{b} \leftarrow A_2\mathbf{b}_2 - A_2S_2PS_1^{-1}\mathbf{b}_1$.

12: Use the Schwartz-Zippel Lemma to check if $f_1(\mathbf{x}) = f_2(A\mathbf{x} + \mathbf{b})$. If yes, return EQUIVALENT, $A$ and $\mathbf{b}$. Else, return NOT EQUIVALENT.

13: **else**

14: Return NOT EQUIVALENT.

15: **end if**

### 6.2.3 Analysis of the algorithm

We establish the correctness of the above algorithm by proving the following lemma.

**Lemma 6.1 (Correctness of Algorithm 11)** *Given black-box access to two $n$-variate polynomials $f_1(\mathbf{x})$, $f_2(\mathbf{x}) \in \mathrm{orb}(ROF_0)$, Algorithm 11 correctly determines with high probability whether they are equivalent or not provided that $\mathrm{char}(\mathbb{F}) = 0$ or $\geq n^2$ and $|\mathbb{F}| \geq n^{13}$. Moreover, if they are equivalent, it returns an $A \in \mathrm{GL}(n, \mathbb{F})$ and a $\mathbf{b} \in \mathbb{F}^n$ such that $f_1(\mathbf{x}) = f_2(A\mathbf{x} + \mathbf{b})$.*

**Proof:** If $f_1 \notin \mathrm{orb}(f_2)$, then Step 12 ensures that the algorithm returns NOT EQUIVALENT with high probability. So suppose that $f_1 \in \mathrm{orb}(f_2)$. In this case, there exists a $\mathtt{C} \in \mathrm{ROF}_0$ such that $f_1, f_2 \in \mathrm{orb}(\mathtt{C})$. Then, $f_1(A_1\mathbf{x}), f_2(A_2\mathbf{x}) \in \text{PS-orb}(\mathtt{C})$ (from Lemma 5.2), and so the only non-zero additive-constants in them are translations, i.e. constants attached to $+$ gates which have a variable as a child. As, from Lemma 5.7, $\mathtt{C}'_1$ and $f_1(A_1\mathbf{x})$, $\mathtt{C}'_2$ and $f_2(A_2\mathbf{x})$ are equal up to scaling of the leaves, the only non-zero additive-constants in $\mathtt{C}'_1$ and $\mathtt{C}'_2$ are

also translations. We now use this fact to show that $\mathtt{C}_1$ and $\mathtt{C}_2$ are the same as $\mathtt{C}$ up to a permutation of variables.

As mentioned above, $f_1(A_1\mathbf{x}), f_2(A_2\mathbf{x}) \in$ PS-orb$(\mathtt{C})$ and $\mathtt{C}'_1, \mathtt{C}'_2$ are same as $f_1(A_1\mathbf{x}), f_2(A_2\mathbf{x})$ up to scaling of leaves. This means that $\mathtt{C}'_1, \mathtt{C}'_2$ can be obtained from $\mathtt{C}$ by permuting, scaling, and translating the variables and scaling the additive-constants (as they are also leaves of $\mathtt{C}'_1, \mathtt{C}'_2$). However as the only non-zero additive constants in $\mathtt{C}'_1$ and $\mathtt{C}'_2$ are translations, these two ROFs differ from $\mathtt{C}$ by just permutation, scaling and translation of the variables. From Observation 5.25, $\mathtt{C}_1$ and $\mathtt{C}_2$ are constant-free regular ROFs obtained from $\mathtt{C}'_1$ and $\mathtt{C}'_2$ by recovering the scaling and translation of variables. Hence, they must be equal to $\mathtt{C}$ up to a permutation of variables.

As $\mathtt{C}_1$ and $\mathtt{C}_2$ are the same up to a permutation of variables, their underlying trees $G_1$ and $G_2$ are isomorphic as rooted trees. So, a permutation $\sigma$ such that $\sigma(G_2) = G_1$ exists. As $\sigma$ is completely determined by its restriction to the leaves of $G_2$, if $P$ is as defined in Step 11, then $\mathtt{C}_1(\mathbf{x}) = \mathtt{C}_2(P\mathbf{x})$. A simple calculation shows that this implies $f_1(\mathbf{x}) = f_2(A\mathbf{x} + \mathbf{b})$ for $A$ and $\mathbf{b}$ defined in Step 11. □

**Running time of the algorithm.** Find-Equivalence(), Reconstruct-ROF(), and Canonize() run in time polynomial in $n$. Also as mentioned in Fact 6.1, a polynomial time algorithm exists for the rooted tree isomorphism problem. Moreover, the Schwartz-Zippel lemma also yields a polynomial time algorithm for checking if $f_1(\mathbf{x}) = f_2(A\mathbf{x} + \mathbf{b})$ in Step 12. Thus, Algorithm 11 runs in time poly$(n)$. This along with Lemma 6.1 proves Theorem 1.7.

## 6.3 Polynomial equivalence for orbits of product depth-2 ROFs

In this section, we gave an algorithm for PE for orbits of additive-constant-free canonical ROFs. Here we show how to solve PE for product-depth 2 canonical ROFs with additive-constants.

### 6.3.1 Overview of the algorithm

**The issue with additive-constants.** Let $f_1$ and $f_2$ be two $n$-variate polynomials in the orbits of ROFs and suppose that they are equivalent. Then there exists a canonical ROF $\mathtt{C}$ such that $f_1, f_2 \in$ orb$(\mathtt{C})$. If $A_1, A_2 \in \mathrm{GL}(n, \mathbb{F})$ are matrices obtained by invoking the Find-Equivalence() algorithm (Algorithm 3) on $f_1$ and $f_2$, respectively, then $f_1(A_1\mathbf{x})$, $f_2(A_2\mathbf{x}) \in$ PS-orb$(\mathtt{C})$. In particular, the additive-constants other than translations in $f_1(A_1\mathbf{x})$ and $f_2(A_2\mathbf{x})$ are the same. However, when we reconstruct $f_1(A_1\mathbf{x})$ and $f_2(A_2\mathbf{x})$ using the Reconstruct-ROF() algorithm (Algorithm 9) and recover the translation of variables using the Canonize()

algorithm (Algorithm 10), the outputs $\mathtt{C}'_1$ and $\mathtt{C}'_2$ are equal to $f_1(A_1\mathbf{x})$ and $f_2(A_2\mathbf{x})$ up to scaling of the leaves. This means that the additive-constants in $\mathtt{C}'_1$ and $\mathtt{C}'_2$ might not be the same. Thus, if we were to construct a permutation matrix $P$ from the isomorphism that maps the underlying tree of $\mathtt{C}'_1$ to that of $\mathtt{C}'_2$, $\mathtt{C}'_1$ need not be equal to $\mathtt{C}'_2(P\mathbf{x})$. So the strategy used in the previous section does not work in a straightforward manner. In this section, we show how to overcome this issue for the case of orbits of product-depth 2 ROFs.

**The idea.** Suppose $f_1 \in \text{orb}(f_2)$; then $\mathtt{C}_1 = \mathtt{C}_2 = \mathtt{C}$, where $\mathtt{C}$ is a product-depth 2 canonical ROF and thus, from Theorem 1.6 $f_1(A_1\mathbf{x}) \in \text{PS-orb}(f_2(A_2\mathbf{x}))$. We reconstruct $f_1(A_1\mathbf{x})$ and $f_2(A_2\mathbf{x})$ to obtain $\mathtt{C}'_1$ and $\mathtt{C}'_2$, recover, and remove the translations of variables in both $\mathtt{C}'_1$ and $\mathtt{C}'_2$. We then show that there is a way to transform $\mathtt{C}'_1$ and $\mathtt{C}'_2$ such that all the non-zero additive-constants in them are 1 and that as ROFs, they only differ by permutation and scaling of variables; we exploit this to give an equivalence test. We then recover the scaling of variables in $\mathtt{C}'_1$ and $\mathtt{C}'_2$. After that we check if for every term $T_1$ in $\mathtt{C}'_1$ there exists a term $T_2$ in $\mathtt{C}'_2$ such that the number of factors of both having 1 as additive-constant and having 0 as additive-constant is the same. Furthermore, we check that for every factor of $T_1$ having 1 (respectively, 0) as additive-constant, there exists a factor of $T_2$ also having 1 (respectively, 0) as additive-constant such that their underlying trees are isomorphic. If $f_1 \in \text{orb}(f_2)$, this must be true for all terms of $\mathtt{C}'_1$ and $\mathtt{C}'_2$ and we are thus able to check for equivalence. Note that here we do not need to worry about their additive-constants as they are the same.

**Transforming $\mathtt{C}'_1$ and $\mathtt{C}'_2$.** Let $f(\mathbf{x}) = f_1(\mathbf{x}), A = A_1, \mathtt{C}'(\mathbf{x}) = \mathtt{C}'_1(\mathbf{x})$ (or $f(\mathbf{x}) = f_2(\mathbf{x}), A = A_2, \mathtt{C}'(\mathbf{x}) = \mathtt{C}'_2(\mathbf{x})$). Suppose that $f(A\mathbf{x}) = T_1 + \cdots + T_s + \gamma$. Then from Lemma 5.7 as $\mathtt{C}'$ and $f(A\mathbf{x})$ are equal up to scaling of leaves and each gate in $\mathtt{C}'$ computes a non-zero constant multiple of the corresponding gate in $f(A\mathbf{x})$, if $\mathtt{C}'(\mathbf{x}) = T'_1 + \cdots + T'_s + \gamma'$, then $T'_i = c_i T_i$ for all $i \in [s]$ and $\gamma' = c_0\gamma$, where $c_0, \ldots, c_s$ are non-zero constants.

**Observation 6.2** $c_0, \ldots, c_s = 1$.

**Proof:** Since $f(A\mathbf{x})$ and $\mathtt{C}'(\mathbf{x})$ are the same polynomials, we get $0 = \mathtt{C}'(\mathbf{x}) - f(A\mathbf{x}) = (c_1 - 1)T_1 + \cdots + (c_s - 1)T_s + (c_0 - 1)\gamma$. Now each of the terms $T_1, \ldots, T_s$ contains a variable not contained in any other term or in $\gamma$. This means $T_1, \ldots, T_s, \gamma$ are linearly independent forcing $c_0 = \cdots = c_s = 1$. □

Let $\mathbf{b}$ be the translation vector output by Canonize($\mathtt{C}'$). Then, from Claim 5.31, $\mathbf{b}$ is also the translation vector of $f(A\mathbf{x})$. So, $f(A\mathbf{x} + \mathbf{b})$ is free of translations and is the same

as $\mathtt{C}'(\mathbf{x}+\mathbf{b})$ up to scaling of the leaves. We shall transform the terms of $\mathtt{C}'(\mathbf{x}+\mathbf{b})$. Let $T = Q_1 \cdots Q_m$ be a term of $f(A\mathbf{x}+\mathbf{b})$. Then, from Lemma 5.7 the corresponding term of $\mathtt{C}'$, $T' = Q'_1 \cdots Q'_m$ will be such that $Q'_i = \beta_i \cdot Q_i,\ \beta_i \neq 0$ for all $i \in [m]$. There are two kinds of $T'$:

- Kind 1: The additive-constant of at least one of the factors $Q'_1, \ldots, Q'_m$ is 0.
- Kind 2: The additive-constant of all the factors $Q'_1, \ldots, Q'_m$ are non-zero.

In the following claims we see how to transform each of the above two kinds of terms.

**Claim 6.1** *Let $T'$ be a term of kind 1 such that for some $k < m$, the additive-constants of $Q'_{k+1}, \ldots, Q'_m$ are zero, while the additive-constants $\alpha'_1, \ldots, \alpha'_k$ of $Q'_1, \ldots, Q'_k$ are non-zero. Also, let $\alpha_1, \ldots, \alpha_m$ be the additive-constants in $T$. Then, if we transform $T'$ by bringing out $\alpha'_1, \ldots, \alpha'_k$ and absorb the product $\alpha' = \prod_{i\in[k]} \alpha'_i$ in $Q'_m$, we recover $T'$ as $\frac{Q_1}{\alpha_1} \cdots \frac{Q_k}{\alpha_k} \cdot (\beta_{k+1}Q_{k+1}) \cdots (\beta_{m-1}Q_{m-1}) \cdot (\beta \cdot \alpha \cdot \beta_m Q_m)$, where $\beta = \prod_{i\in[k]} \beta_i$ and $\alpha = \prod_{i\in[k]} \alpha_i$. Also, the only non-zero additive-constants in $T'$ are all 1.*

**Proof:** As $Q'_i = \beta_i Q_i$, $\alpha'_i = \beta_i \alpha_i$ for all $i \in [m]$. Thus, when we bring out $\alpha'_1, \ldots, \alpha'_k$ from $Q'_1, \ldots, Q'_k$ we recover $T'$ as $T' = (\beta_1\alpha_1) \cdots (\beta_k\alpha_k) \cdot \frac{\beta_1 Q_1}{\beta_1\alpha_1} \cdots \frac{\beta_k Q_k}{\beta_k\alpha_k} \cdot (\beta_{k+1}Q_{k+1}) \cdots (\beta_m Q_m)$, which implies $T' = (\beta_1\alpha_1) \cdots (\beta_k\alpha_k) \cdot \frac{Q_1}{\alpha_1} \cdots \frac{Q_k}{\alpha_k} \cdot (\beta_{k+1}Q_{k+1}) \cdots (\beta_m Q_m)$.

So, after absorbing $\alpha'_1 \cdots \alpha'_k = (\beta_1\alpha_1) \cdots (\beta_k\alpha_k)$ in $Q_m$, we get $T'$ in the desired form. Also, the only non-zero additive-constants are those in $\frac{Q_1}{\alpha_1}, \ldots, \frac{Q_k}{\alpha_k}$ and they are 1 by the definition of $\alpha_1, \ldots, \alpha_k$. □

**Claim 6.2** *Let $T'$ be a term of kind 2 and the additive-constants of $Q'_1, \ldots, Q'_m$ be $\alpha'_1, \ldots, \alpha'_m$. Also, let $\alpha_1, \ldots, \alpha_m$ be the additive-constants of $T$. Then, if we transform $T'$ by bringing out $\alpha'_1, \ldots, \alpha'_m$, we recover $T'$ as $\alpha \cdot \frac{Q_1}{\alpha_1} \cdots \frac{Q_m}{\alpha_m}$, where $\alpha = \prod_{i\in[m]} \alpha_i$. Also, all additive-constants in $T'$ are 1.*

**Proof:** As $Q'_i = \beta_i Q_i$, $\alpha'_i = \beta_i \alpha_i$ for all $i \in [m]$. Thus, when we bring out $\alpha'_1, \ldots, \alpha'_m$ from $Q'_1, \ldots, Q'_m$ we recover $T'$ as $T' = (\beta_1\alpha_1) \cdots (\beta_m\alpha_m) \cdot \frac{\beta_1 Q_1}{\beta_1\alpha_1} \cdots \frac{\beta_m Q_m}{\beta_m\alpha_m}$. Observation 6.2 implies $\beta_1 \cdots \beta_m = 1$ and we get $T' = \alpha \cdot \frac{Q_1}{\alpha_1} \cdots \frac{Q_m}{\alpha_m}$. By the definition of $\alpha_1, \ldots, \alpha_m$ all additive-constants in $T'$ are 1. □

Now, suppose that $f_1 \in \text{orb}(f_2)$ and $\mathbf{b}_1, \mathbf{b}_2$ are translation vectors of $\mathtt{C}'_1$ and $\mathtt{C}'_2$ recovered using the Canonise algorithm. Then from Claim 5.31, as they are also translation vectors of $f_1(A_1\mathbf{x})$ and $f_2(A_2\mathbf{x})$, $f_1(A_1\mathbf{x}+\mathbf{b}_1)$ and $f_2(A_2\mathbf{x}+\mathbf{b}_2)$ are free of translations. Moreover,

as ROFs they only differ by permutation and scaling of variables. Notice that, in this case, the above two claims imply that the same relationship also holds between $\mathtt{C}'_1(\mathbf{x}+\mathbf{b}_1)$ and $\mathtt{C}'_2(\mathbf{x}+\mathbf{b}_2)$. As we exploit this property to give an equivalence test for $f_1$ and $f_2$, we record it as an observation.

**Observation 6.3** *If $f_1 \in \mathrm{orb}(f_2)$, then after modifying $\mathtt{C}'_1(\mathbf{x}+\mathbf{b}_1)$ and $\mathtt{C}'_2(\mathbf{x}+\mathbf{b}_2)$ according to Claims 6.1 and 6.2, $\mathtt{C}'_1(\mathbf{x}+\mathbf{b}_1)$ and $\mathtt{C}'_2(\mathbf{x}+\mathbf{b}_2)$ as ROFs only differ by permutation and scaling of variables.*

### 6.3.2 The Algorithm

**Algorithm 12** Product-Depth-2-Equivalence-Test$(f_1(\mathbf{x}), f_2(\mathbf{x}))$

**Input:** Black-box access to $f_1(\mathbf{x}), f_2(\mathbf{x})$ in the orbits of product-depth 2 canonical ROFs.

**Output:** Whether or not $f_1$ and $f_2$ are equivalent. If they are equivalent, then $A \in \mathrm{GL}(n,\mathbb{F})$ and $\mathbf{b} \in \mathbb{F}^n$ such that $f_1(\mathbf{x}) = f_2(A\mathbf{x}+\mathbf{b})$.

/* Reconstructing canonical ROFs equivalent to $f_1$ and $f_2$ and transforming their terms. */

**for** $i \in [2]$ **do**
  $A_i \leftarrow$ Find-Equivalence$(f_i(\mathbf{x}))$ (Algorithm 3).
  $\mathtt{C}'_i \leftarrow$ Reconstruct-ROF$(f_i(A_i\mathbf{x}))$ (Algorithm 9).
  $\mathbf{b}_i \leftarrow$ translation vector returned by Canonize$(\mathtt{C}'_i)$ (Algorithm 10). $\mathtt{C}'_i \leftarrow \mathtt{C}'_i(\mathbf{x}+\mathbf{b}_i)$.
  Transform all terms in $\mathtt{C}'_i$ according to Claims 6.1 and 6.2. $\mathtt{C}'_i \leftarrow$ the ROF obtained after the transformation and recovering scaling of variables, $S_i \leftarrow$ the scaling matrix.
**end for**

/* Checking if $\mathtt{C}'_1$ and $\mathtt{C}'_2$ are equivalent. */

**if** the additive-constants of $\mathtt{C}'_1$ and $\mathtt{C}'_2$ attached to the top $+$ gate are not equal **then**
  Return NOT EQUIVALENT.
**end if**
$N_1 \leftarrow$ set of terms of $\mathtt{C}'_1$, $N_2 \leftarrow$ set of terms of $\mathtt{C}'_2$, $P \leftarrow I_{n\times n}$, the permutation matrix mapping the variables of $\mathtt{C}'_1$ to the variables of $\mathtt{C}'_2$.
**for** $T'_1 \in N_1$ **do**
  If $T'_1$ is a term of kind 1 and $\exists\ T'_2 \in N_2$ also of kind 1 such that Check-Kind-1$(T'_1, T'_2)$ returns SUCCESS, then $N_2 \leftarrow N_2 \setminus \{T'_2\}$. Update $P$ so that it maps var$(T'_1)$ to var$(T'_2)$ appropriately.

16: If $T_1'$ is a term of kind 2 and $\exists\ T_2' \in N_2$ also of kind 2 such that Check-Kind-2$(T_1', T_2')$ returns SUCCESS, then $N_2 \leftarrow N_2 \setminus \{T_2'\}$. Update $P$ so that it maps var$(T_1')$ to var$(T_2')$ appropriately.
17: **end for**
18: **if** $N_2 = \emptyset$ **then**
19: $A \leftarrow A_2 S_2 P S_1^{-1} A_1^{-1}$, $\mathbf{b} \leftarrow A_2\mathbf{b}_2 - A_2 S_2 P S_1^{-1}\mathbf{b}_1$.
20: Use the Schwartz-Zippel Lemma to check if $f_1(\mathbf{x}) = f_2(A\mathbf{x} + \mathbf{b})$. If yes, return EQUIVALENT, $A$ and $\mathbf{b}$. Else, return NOT EQUIVALENT.
21: **else**
22: Return NOT EQUIVALENT.
23: **end if**

The checks on lines 15 and 16 are performed using the following algorithms.

**Algorithm 13** Check-Kind-1$(T_1', T_2')$

**Input:** Terms $T_1'$ of $\mathsf{C}_1'$ and $T_2'$ of $\mathsf{C}_2'$ of Kind 1.

**Output:** SUCCESS if they are equivalent, FAILURE otherwise.

1: Suppose $T_1' = Q_{1,1}' \cdots Q_{1,k_1}' \cdot Q_{1,k_1+1}' \cdots Q_{1,m_1}'$ and $T_2' = Q_{2,1}' \cdots Q_{2,k_2}' \cdot Q_{2,k_2+1}' \cdots Q_{2,m_2}'$, where $Q_{1,1}', \cdots, Q_{1,k_1}'$ and $Q_{2,1}', \cdots, Q_{2,k_2}'$ are the only factors with additive-constants.
2: **if** $k_1 \neq k_2$ or $m_1 \neq m_2$ **then**
3: Return FAILURE
4: **end if**
5: **if** there exists a bijection $\sigma : [m_1] \to [m_1]$ such that $\sigma([k_1]) = [k_1]$ and $\forall i \in [m_1]$, the rooted trees of $Q_{1,i}'$ and $Q_{2,\sigma(i)}'$ are isomorphic **then**
6: Return SUCCESS.
7: **else**
8: Return FAILURE.
9: **end if**

**Algorithm 14** Check-Kind-2$(T_1', T_2')$

**Input:** Terms $T_1'$ of $\mathsf{C}_1'$ and $T_2'$ of $\mathsf{C}_2'$ of kind 2.

**Output:** SUCCESS if they are equivalent, FAILURE otherwise.

1: Suppose $T_1' = \alpha_1' \cdot Q_{1,1}' \cdots Q_{1,m_1}'$ and $T_2' = \alpha_2' \cdot Q_{2,1}' \cdots Q_{2,m_2}'$.
2: **if** $\alpha_1' \neq \alpha_2'$ or $m_1 \neq m_2$ **then**
3: Return FAILURE
4: **end if**
5: **if** there exists a bijection $\sigma : [m_1] \to [m_1]$ such that $\forall i \in [m_1]$, the rooted trees of $Q_{1,i}'$ and $Q_{2,\sigma(i)}'$ are isomorphic **then**
6: Return SUCCESS.
7: **else**
8: Return FAILURE.
9: **end if**

### 6.3.3 Analysis of the algorithm

We establish the correctness of the above algorithm by proving the following lemma.

**Lemma 6.2 (Correctness of Algorithm 12)** *Given black-box access to two $n$-variate polynomials $f_1(\mathbf{x})$, $f_2(\mathbf{x})$ in the orbits of two unknown product-depth 2 canonical ROFs, Algorithm 12 correctly determines whether they are equivalent or not provided that* $\mathrm{char}(\mathbb{F}) = 0$ *or* $\geq n^2$ *and* $|\mathbb{F}| \geq n^{13}$. *Moreover, if they are equivalent, it returns an* $A \in \mathrm{GL}(n, \mathbb{F})$ *and a* $\mathbf{b} \in \mathbb{F}^n$ *such that* $f_1(\mathbf{x}) = f_2(A\mathbf{x} + \mathbf{b})$.

**Proof:** If $f_1 \notin \mathrm{orb}(f_2)$, then Step 20 ensures that the algorithm returns NOT EQUIVALENT with high probability. So suppose that $f_1 \in \mathrm{orb}(f_2)$. Then, from Observation 6.3, we have that after Step 6 of the algorithm, $\mathtt{C}_1'$ and $\mathtt{C}_2'$ as ROFs only differ by permutation of variables (because the scaling of variables has already been recovered). Thus, if the additive-constants attached to the top-most gates in $\mathtt{C}_1'$ and $\mathtt{C}_2'$ are not equal, $f_1 \notin \mathrm{orb}(f_2)$ and so Step 11 is correct.

Now for every term $T_1$ of $f_1$, there must exist a term $T_2$ of $f_2$ such that $T_1 \in \text{PS-orb}(T_2)$. Then, Observation 6.3 also implies that the corresponding terms $T_1'$ and $T_2'$ of $f_1'$ and $f_2'$ as ROFs must be same up to permutation of variables. It is easy to see that this is true if and only if, depending on the kind of these terms, either Check-Kind-1$(T_1', T_2')$ or Check-Kind-2$(T_1', T_2')$ succeeds. Hence the algorithm correctly determines whether $f_1$ and $f_2$ are equivalent or not. A simple calculation then shows that $f_1(\mathbf{x}) = f_2(A\mathbf{x} + \mathbf{b})$ for $A$ and $\mathbf{b}$ as defined in Step 20. □

**Running time of the algorithm.** Find-Equivalence(), Reconstruct-ROF(), and Canonize() run in time polynomial in $n$. Also as mentioned in Fact 6.1, a polynomial time algorithm exists for the rooted tree isomorphism problem. This implies that Check-Kind-1() and Check-Kind-2() run in time poly$(n)$. As $|N_1|, |N_2| \leq n$, this means that the for loop of lines 14-17 also runs in poly$(n)$ time. Moreover, the Schwartz-Zippel lemma also yields a polynomial time algorithm for checking if $f_1(\mathbf{x}) = f_2(A\mathbf{x} + \mathbf{b})$ in Step 20. Thus, Algorithm 12 runs in time poly$(n)$. This along with Lemma 6.2 proves Theorem 1.8.

# Chapter 7

# Lower bounds for constant depth arithmetic circuits

In a breakthrough paper [LST21], Limaye, Srinivasan, and Tavenas proved super-polynomial lower bounds against low depth arithmetic circuits. This chapter provides an alternate and a more direct proof of their result. The contents of this chapter are from a joint work with Prashanth Amireddy, Ankit Garg, Neeraj Kayal, and Chandan Saha [AGK+23].

## 7.1 Introduction

In a remarkable work, [LST21], Limaye, Srinivasan, and Tavenas solved the long-standing open problem of proving super-polynomial lower bounds for constant depth arithmetic circuits. They showed that any circuit with product-depth $\Delta$ (see Definition 2.1) computing the $n$ variate, degree $d$ $IMM$ polynomial (see Definition 2.15) must have size at least $d^{O(d)}n^{\Omega(d^{2^{1-2\Delta}})}$; this yields a super-polynomial lower bound as long as $d = O(\log n)$ and $\Delta = \frac{\log n}{\log\log n}$. They do this by showing that if $IMM$ is computed by a size $s$, product-depth $\Delta$ circuit $C$, then it must also be computed by a homogeneous circuit $C'$ with product-depth $2\Delta$ and size $s \cdot d^{O(d)}$. They further show that $C'$ can be converted into a set-multilinear circuit $C''$ of product-depth $2\Delta$ and size $s \cdot d^{O(d)}$ and finally prove a lower bound for this $C''$ using the relative rank measure, which is a variant of the partial derivatives measure. In this section, we give a more direct proof of their result: we prove a lower bound for $C'$ directly using the shifted partials (SP) and the APP measures (see Definition 2.17). Specifically, we prove the following theorem.

**Theorem 1.9 (Lower bound for low-depth homogeneous formulas via shifted partials)** *Let* $\mathsf{C}$ *be a homogeneous formula of size* $s$ *and product-depth* $\Delta$ *that computes a polynomial of degree* $d$ *in*

*n variables. Then for appropriate values of k and $\ell$,*

$$\mathsf{SP}_{k,\ell}(\mathsf{C}) \leq \frac{s\, 2^{O(d)}}{n^{\Omega(d^{2^{1-\Delta}})}} \min\{M(n,k)M(n,\ell), M(n,d-k+\ell)\}.$$

*At the same time, there are homogeneous polynomials f of degree d in n variables (e.g., an appropriate projection of iterated matrix multiplication polynomial, Nisan-Wigderson design polynomial, etc.) such that*

$$\mathsf{SP}_{k,\ell}(f) \geq 2^{-O(d)} \min\{M(n,k)M(n,\ell), M(n,d-k+\ell)\}.$$

*This gives a lower bound of $\frac{n^{\Omega(d^{2^{1-\Delta}})}}{2^{O(d)}}$ on the size of homogeneous product-depth $\Delta$ formulas for f.*

In Section 7.2, we describe some notations and definitions specific to this chapter. In Section 7.3, we give a high-level overview of the techniques. Section 7.4 analyses the structure of a certain space of partial derivatives; the results proved in this section play a vital role in giving us a more direct proof in Section 7.5.

## 7.2 Preliminaries

Let $a, b, c$ be real numbers. Then we define the sets $[a..b] := \{x \in \mathbb{Z} : x \in [a,b]\}$ and $[a] := [1..a]$. For a constant $c \geq 1$ and $b \geq 0$, we say $a \approx_c b$ if $a \in [b/c, b]$. We write $a \approx b$ if $a \approx_c b$ for some (unspecified) constant $c$. All logarithms have base 2 unless specified otherwise. We denote the fractional part of $a$ by $\{a\} := a - \lfloor a \rfloor$ and the nearest integer of $a$ by $\lfloor a \rceil$. The following quantity will be crucially used in the proofs of our lower bounds. Here, we think of $d_1, \ldots, d_t$ as degrees of certain homogeneous polynomials, $d$ as the degree of the product of those polynomials, and $k$ is the order of partial derivatives used for the complexity measures.

**Definition 7.1 (**residue**)** *For non-negative integers $d_1, \ldots, d_t$ such that $d := \sum_{i=1}^{t} d_i \geq 1$ and $k \in [0..(d-1)]$, we define* $\mathsf{residue}_k(d_1, \ldots, d_t) := \frac{1}{2} \cdot \min_{k_1,\ldots,k_t \in \mathbb{Z}} \sum_{i=1}^{t} \left| k_i - \frac{k}{d} \cdot d_i \right|$.

The factor of half has been included in the definition just to make the statements of some of the lemmas in our analysis simple. It is easy to show that $\mathsf{residue}_k(d_1, \ldots, d_t) \leq \frac{k}{2}$. The minimum is attained when for all $i \in [t]$, $k_i = \left\lfloor \frac{k}{d} \cdot d_i \right\rceil$. When we use residue in the analysis of complexity measures, we would also have the following additional constraints that $k_i \geq 0$ and $k_i \leq d_i$, $k_1 + \cdots + k_n = k$, where $k$ shall be the order of derivatives. As the value of residue can not decrease when we impose these constraints, we omit them.

**Sets and functions.** When some sets $S_1, \ldots, S_t$ are pair-wise disjoint, we write their union as $S_1 \sqcup \cdots \sqcup S_t$. For a function $\mu : S \to T$ and a subset $A \subseteq S$, we define the multiset $\mu(A) := \{\mu(x) : x \in A\}$. Clearly $|\mu(A)| = |A|$[1]. We denote the power set of a set $S$ by $2^S$. We say that a function $\mu : S \to T$ extends $\kappa : A \to T$ if $A \subseteq S$ and for all $x \in A$, $\mu(x) = \kappa(x)$. Let $\mu : S \to T$ and $\kappa : A \to T$ be functions such that $S \cap A = \emptyset$. Then $\mu \sqcup \kappa : S \sqcup A \to T$ is defined by setting $(\mu \sqcup \kappa)(x) = \mu(x)$ for all $x \in S$ and $(\mu \sqcup \kappa)(x) = \kappa(x)$ for all $x \in A$.

**Binomial coefficients.** For non-negative integers $a, b$, we shall denote the quantity $\binom{a+b-1}{b}$ by $M(a, b)$. Note that $M(a, b)$ is the number of (monic) monomials of degree $b$ over $a$ many variables.

The following lemma is useful when dealing with binomial coefficients.

**Lemma 7.1 (Approximations for $M(a, b)$)** *For positive integers $a \geq b \geq c$ and $d$, we have*

1. $(a/b)^b \leq M(a, b) \leq (6a/b)^b$,
2. $(a/2b)^c \leq \frac{M(a,b+c)}{M(a,b)} \leq (2a/b)^c$,
3. $\frac{M(c,d)}{M(b,d)} \geq (c/b)^d$.

**Proof:**

1.

$$M(a, b) = \frac{(a+b-1)\cdots(a)}{b!} \geq \frac{a^b}{b^b}\text{, and}$$

$$M(a, b) \leq \frac{(a+b-1)^b}{\left(\frac{b}{e}\right)^b} \qquad \text{(using } b! \geq (b/e)^b\text{)}$$
$$\leq \frac{(2a)^b \cdot e^b}{b^b} \leq \left(\frac{6a}{b}\right)^b.$$

2.

$$\frac{M(a, b+c)}{M(a, b)} = \left(\frac{a+b+c-1}{b+c}\right) \cdots \left(\frac{a+b}{b+1}\right).$$

[1]For a multiset $B$, $|B|$ denotes its size, i.e. the number of elements in $B$ counted with their respective multiplicities.

The bounds follow from the fact that each of the above $c$ many fractions lies between $\frac{a}{2b}$ and $\frac{2a}{b}$.

3.

$$\frac{M(c,d)}{M(b,d)} = \left(\frac{c+d-1}{b+d-1}\right)\cdots\left(\frac{c}{b}\right).$$

The lower bound follows from the fact that each of the above $d$ many fractions is at least $\frac{c}{b}$.

□

## 7.3 Proof techniques

In this section, we explain the proof idea and compare it with that in [LST21]. A lot of lower bounds in arithmetic complexity follow the following outline.

**Step 1: Depth reduction.** One first shows that if $f(\mathbf{x})$ is computed by a *small* circuit from some restricted subclass of circuits, then there is a corresponding subclass of depth-4 circuits such that $f(\mathbf{x})$ is also computed by a *relatively small* circuit from this subclass[1]. The resulting subclass is of the form: $f(\mathbf{x}) = \sum_{i=1}^{s}\prod_{j=1}^{t_i} Q_{i,j}$. Usually there are simple restrictions on the degrees of $Q_{i,j}$'s. For example, they could be upper bounded by some number.

**Step 2: Employing a suitable set of linear maps.** Let $\mathbb{F}[\mathbf{x}]^{=d}$ be the space of homogeneous polynomials of degree $d$, $W$ be a suitable vector space, and $\mathrm{Lin}(\mathbb{F}[\mathbf{x}]^{=d}, W)$ be the space of linear maps from $\mathbb{F}[\mathbf{x}]^{=d}$ to $W$. We choose a suitable set of linear maps $\mathcal{L} \subseteq \mathrm{Lin}(\mathbb{F}[\mathbf{x}]^{=d}, W)$ that define a complexity measure $\mu_{\mathcal{L}}(f) := \dim(\mathcal{L}(f))$, where $\mathcal{L}(f) := \langle\{L(f) \,:\, L \in \mathcal{L}\}\rangle$.

We would like to choose $\mathcal{L}$ so that it identifies some weakness of the terms $\prod_{j=1}^{t} Q_j$ in the depth-4 circuit. That is, $\mu_{\mathcal{L}}\left(\prod_{j=1}^{t} Q_j\right)$ should be much smaller than $\mu_{\mathcal{L}}(f)$ for a generic $f$. For e.g., if $Q_j$'s are all linear polynomials, we can choose $\mathcal{L}$ to be the partial derivatives of order $k$, $\partial^k$. Then, $\mu_{\mathcal{L}}\left(\prod_{j=1}^{t} Q_j\right) \leq \binom{t}{k} \ll \binom{n+k-1}{k}$ which is the value for a generic $f$ (for $k \leq t/2$). This is the basis of the homogeneous depth-3 formula lower bound in [NW97].

For proving lower bounds for bounded bottom fan-in depth-4 circuits (i.e., when degree of $Q_j$'s is upper bounded by some number), [GKKS14, Kay12b] introduced the SP measure

[1]Some major results in the area such as [Raz03, LST21] did not originally proceed via a depth reduction but instead analysed formulas directly. These results can however be restated as first doing a depth reduction and then applying the appropriate measure.

and used the linear maps $\mathcal{L} = \mathbf{x}^{\ell} \cdot \partial^{k}$. The main insight in their proof was that if we apply a partial derivative of order $k$ on $\prod_{j=1}^{t} Q_j$ and use the product rule, then at least $t - k$ of the $Q_j$'s remain untouched. This structure can then be exploited by the shifts to get a lower bound. This intuition however completely breaks down for $k \geq t$. Due to this, progress remain stalled for higher depth arithmetic circuit lower bounds via SP.

In a major breakthrough, [LST21] gets around the above obstacle by working with set-multilinear circuits which entails working with polynomials over $d$ sets of variables $(\mathbf{x}_1, \ldots, \mathbf{x}_d)$, $|\mathbf{x}_i| = n$. Let us use the shorthand $\mathbf{x}_S = (\mathbf{x}_i)_{i \in S}$. The products they deal with are of the form $\prod_{j=1}^{t} Q_j(\mathbf{x}_{S_j})$, where $S_1, S_2, \ldots, S_t$ form a partition of $[d]$. The set of linear maps they use are $\mathcal{L} = \Pi \circ \partial_{\mathbf{x}_A}$ for a subset $A \subseteq [d]$. Here, $\Pi$ is a map that sets $n - n_0$ variables in each of the variable sets in $\mathbf{x}_{[d]\setminus A}$ to 0. They observe (for the appropriate choice of $n_0$) that $\mu_{\mathcal{L}}\left(\prod_{j=1}^{t} Q_j(\mathbf{x}_{S_j})\right) \leq \frac{n^{|A|}}{2^{\frac{1}{2}\sum_{j=1}^{t} \text{imbalance}_j}}$.

Here, $\text{imbalance}_j = ||A \cap S_j| \log(n) - |S_j \setminus A| \log(n_0)|$. For the appropriate choice of $n_0$, a generic set-multilinear $f$ satisfies $\mu_{\mathcal{L}}(f) = n^{|A|}$, so that lower bound (on the number of summands) obtained is exponential in the total imbalance $\sum_{j=1}^{t} \text{imbalance}_j$. [LST21] observe that this quantity is *somewhat large* for the depth-4 circuits that they consider.

The core of the above derivatives-based argument allows us to unravel some structure in partial derivatives of order $k$ applied on $\prod_{j=1}^{t} Q_j$ for values of $k \gg t$. We use this to derive a structure for the partial derivative space of a product $\prod_{j=1}^{t} Q_j(\mathbf{x})$. Consider a partial derivative operator of order $k$ indexed by a multiset $\alpha$ of size $k$. Using the chain rule,

$$\partial_{\alpha} \prod_{j=1}^{t} Q_j = \sum_{\alpha_1, \ldots, \alpha_t : \sum_{i=1}^{t} \alpha_i = \alpha} c^{\alpha}_{\alpha_1, \ldots, \alpha_t} \prod_{j=1}^{t} \partial_{\alpha_j} Q_j$$

for appropriate constants $c^{\alpha}_{\alpha_1, \ldots, \alpha_t}$'s. In the product $\prod_{j=1}^{t} \partial_{\alpha_j} Q_j$, we can try to club terms into two groups depending on if the size of $|\alpha_j|$ is small or large. It turns out that the right threshold for $|\alpha_j|$ is $k \deg(Q_j)/d$ (i.e., if we divide the order of the derivatives proportional to the degrees of the terms). Let $S := \{j : |\alpha_j| \leq k \deg(Q_j)/d\}$. Define $k_0 := \sum_{j \in S} |\alpha_j|$ and $\ell_0 := \sum_{j \in \overline{S}} (\deg(Q_j) - |\alpha_j|)$. Notice that we can write the product $\prod_{j=1}^{t} \partial_{\alpha_j} Q_j$ as $P \prod_{j \in S} \partial_{\alpha_j} Q_j$, for a degree $\ell_0$ polynomial $P$. Hence, $\partial_{\alpha} \prod_{j=1}^{t} Q_j$ is a sum of terms of this form. While it is not immediate (due to the condition on $\alpha_j$'s in $S$), with a bit more work, one can combine the product of partials into a single partial.

What can we say about $k_0$ and $\ell_0$? It turns out that the quantity that comes up in the calculations is $k_0 + \frac{k}{d-k}\ell_0$ and it satisfies $k_0 + \frac{k}{d-k}\ell_0 \leq k$. Note that $k_0$ is between 0 and $k$, and $\ell_0$ between 0 and $d - k$. So the normalization brings $\ell_0$ to the right 'scale'.

It turns out we can give a better bound in terms of a quantity we call *residue* defined as

$$\mathsf{residue}_k(d_1,\ldots,d_t) := \frac{1}{2}\cdot \min_{k_1,\ldots,k_t\in\mathbb{Z}} \sum_{j=1}^{t} \left| k_j - \frac{k}{d}\cdot d_j \right|.$$

and having the property that:

**Proposition 7.1** *Let $k_0$ and $\ell_0$ be defined as above. Then, $k_0 + \frac{k}{d-k}\ell_0 \le k - \mathsf{residue}_k(d_1,\ldots,d_t)$, where $d_j = deg(Q_j)$.*

We want to spread the derivatives equally among all terms but cannot due to *integrality issues*. The residue captures this quantitatively and as described below, is what gives us our lower bounds. While the proof in [LST21] also relies on an integrality issue, there it originates from an imbalance between the sizes of the variable sets involved in a set-multilinear partition (as the map $\Pi$ sets some variables in certain sets to 0). In contrast, we show that the integrality issue arising directly from the derivatives can be leveraged without involving set-multilinearity. In this sense, our approach is *conceptually* direct and simpler. Combined with the above discussion, we get the following structural lemma about the derivative space of $\prod_{j=1}^{t} Q_j$.

**Lemma 7.2**

$$\left\langle \partial^k \left(Q_1\cdots Q_t\right)\right\rangle \subseteq \sum_{\substack{S\subseteq[t],\, k_0\in[0..k],\, \ell_0\in[0..(d-k)],\\ k_0+\frac{k}{d-k}\cdot\ell_0 \le k-\mathsf{residue}_k(d_1,\ldots,d_t)}} \left\langle \mathbf{x}^{\ell_0}\cdot\partial^{k_0}\left(\prod_{j\in S} Q_j\right)\right\rangle.$$

Now we have the choice to utilize the above structure using an additional set of linear maps. Both shifts and projections give similar lower bounds, so let us explain shifts here. Note that there is an intriguing possibility of getting even better lower bounds (in terms of dependence on $d$) using other sets of linear maps! From the above structural result, we have

$$\left\langle \mathbf{x}^{\ell}\cdot\partial^k \left(Q_1\cdots Q_t\right)\right\rangle \subseteq \sum_{\substack{S\subseteq[t],\, k_0\in[0..k],\, \ell_0\in[0..(d-k)],\\ k_0+\frac{k}{d-k}\cdot\ell_0 \le k-\mathsf{residue}_k(d_1,\ldots,d_t)}} \left\langle \mathbf{x}^{\ell+\ell_0}\cdot\partial^{k_0}\left(\prod_{j\in S} Q_j\right)\right\rangle.$$

Thus we can upper bound,

$$\mathsf{SP}_{k,\ell}((Q_1 \cdots Q_t)) \leq 2^t \cdot d^2 \cdot \max_{\substack{k_0,\ell_0 \geq 0 \\ k_0 + \frac{k}{d-k}\cdot \ell_0 \leq k - \mathsf{residue}_k(d_1,\ldots,d_t)}} M(n,k_0) \cdot M(n,\ell_0+\ell)$$

$$\leq 2^t \cdot d^2 \frac{2^{O(d)}}{n^{\mathsf{residue}_k(d_1,\ldots,d_t)}} \min\{M(n,k)M(n,\ell), M(n,d-k+\ell)\},$$

where the second inequality follows from elementary calculations.

Now to upper bound the shifted partial dimension of polynomials computed by low-depth formulas, we give a decomposition for such formulas into sums of products of polynomials (Lemma 7.5) where the degree sequences are carefully chosen so that that the residues can be simultaneously lower bounded for all the terms (Lemma 7.6). While in a different context, these calculations do bear similarity with related calculations in [LST21].

**Step 3: Lower bounding $\dim(\mathcal{L}(f))$ for an explicit $f$.** As a last step, one shows that for some explicit candidate hard polynomial $\dim(\mathcal{L}(f))$ is large and thereby obtains a lower bound. This is another step where bypassing set-multilinearity helps as one is not constrained to pick a set-multilinear hard polynomial. Indeed, using a straightforward analysis we show that the APP measure is high for an explicit non-set-multilinear polynomial (see Remark 7.2). We also show that the measures are high for more standard polynomial families such as the iterated matrix multiplication polynomials and the Nisan-Wigderson design polynomials.

## 7.4 Structure of the space of partial derivatives of a product

In this section, we bound the partial derivative space of a product of homogeneous polynomials. In the following lemma, we show that the space of $k$-th order partial derivatives of a product of polynomials is contained in a sum of shifted partial spaces with shift $\ell_0$ and order of derivatives $k_0$ such that $k_0 + \frac{k}{d-k} \cdot \ell_0$ is 'small'. Using this lemma, we upper bound the SP and APP measures of a product of homogeneous polynomials. These bounds are then used in Section 7.5 for proving lower bounds for low-depth homogeneous formulas.

**Lemma 7.3 (Upper bounding the partials of a product)** *Let $n$ and $t$ be positive integers and $Q_1, \ldots, Q_t$ be non-constant, homogeneous polynomials in $\mathbb{F}[\mathbf{x}]$ with degrees $d_1, \ldots, d_t$ respectively.*

*Let* $d := \deg(Q_1 \cdots Q_t) = \sum_{i=1}^{t} d_i$ *and* $k < d$ *be a non-negative integer. Then,*

$$\left\langle \partial^k (Q_1 \cdots Q_t) \right\rangle \subseteq \sum_{\substack{S \subseteq [t],\, k_0 \in [0..k],\, \ell_0 \in [0..(d-k)], \\ k_0 + \frac{k}{d-k} \cdot \ell_0 \,\leq\, k - \mathsf{residue}_k(d_1,\ldots,d_t)}} \left\langle \mathbf{x}^{\ell_0} \cdot \partial^{k_0} \left( \prod_{i \in S} Q_i \right) \right\rangle .$$

**Proof:** We first give some intuition about the proof. Let $m$ be any multilinear monomial (i.e., no variable appears more than once) of degree $k$, and $X$ be the corresponding set of variables. Then, by the product rule $\partial_X(Q_1 \cdots Q_t)$ can be expressed as the sum

$$\sum_{\substack{(X_1,\ldots,X_t): \\ X_1 \sqcup \cdots \sqcup X_t = X}} \partial_{X_1} Q_1 \cdots \partial_{X_t} Q_t. \tag{7.1}$$

Note that since the sizes of $X_i$'s should sum to $k$ and the degrees of $Q_i$'s should sum to $d$ in each term of the above summation, some factors are differentiated 'a lot' while the others are differentiated only 'a little'. More specifically, if $|X_i| > \frac{k}{d} \cdot d_i$, we use the fact that $\partial_{X_i} Q_i \in \left\langle \mathbf{x}^{d_i - |X_i|} \right\rangle$ and otherwise we use $\partial_{X_i} Q_i \in \left\langle \partial^{|X_i|} Q_i \right\rangle$ to conclude that

$$\partial_{X_1} Q_1 \cdots \partial_{X_t} Q_t \in \left\langle \mathbf{x}^{\sum_{i \in \overline{S}} \ell_{0,i}} \cdot \prod_{i \in S} \partial^{k_{0,i}} Q_i \right\rangle,$$

where $S := \left\{ i \in [t] : |X_i| \leq \frac{k}{d} \cdot d_i \right\}$, $\overline{S} = [t] \setminus S$, $\ell_{0,i} = d_i - |X_i|$ for all $i \in \overline{S}$, and $k_{0,i} = |X_i|$ for all $i \in S$. By the nature of our construction, we can show that $k_0 + \frac{k}{d-k} \cdot \ell_0 \leq k - \mathsf{residue}_k(d_1, \ldots, d_t)$, where $k_0 := \sum_{i \in S} k_{0,i}$ and $\ell_0 := \sum_{i \in \overline{S}} \ell_{0,i}$ (see the calculations at the end of the proof). Now suppose it holds that $\prod_{i \in S} \partial^{k_{0,i}} Q_i = \partial^{k_0} \prod_{i \in S} Q_i$. In such a case, we would get the space required in the R.H.S. of the lemma statement, and we would be done. However, this assumption need not be true if $|S| \geq 1$. To get around this issue, we employ an inductive argument on the size of $S$ (see Claim 7.2). For this argument, it will be helpful to combine certain terms in the sum (7.1) depending on the set of factors that are differentiated a 'lot' (see Claim 7.1). We now present the proof in full detail. Since, in general, the variables in $m$ need not be distinct, it will be convenient to think of degree $k$ monomials over $\mathbf{x}$ as maps from $[k]$ to $\mathbf{x}$.

For a function $\mu : P \to \mathbf{x}$ and any $P' \subseteq P$, recall that $\mu(P')$ refers to the multiset of images of the elements of $P'$ under $\mu$. Thus $|\mu(P')| = |P'|$. Let $\mathcal{V}$ be the set of polynomials on the

R.H.S., i.e.,

$$\mathcal{V} := \sum_{\substack{S \subseteq [t],\ k_0 + \frac{k}{d-k} \cdot \ell_0 \\ \leq\ k - \mathsf{residue}_k(d_1,\ldots,d_t)}} \left\langle \mathbf{x}^{\ell_0} \cdot \boldsymbol{\partial}^{k_0} \left( \prod_{i \in S} Q_i \right) \right\rangle .$$

We now argue that for an arbitrary total function $\mu : [k] \to \mathbf{x}$, $\partial_{\mu([k])} \left( \prod_{i \in [t]} Q_i \right) \in \mathcal{V}$; the lemma then follows immediately. We use the following identity which is a direct consequence of the product rule for derivatives:

$$\partial_{\mu([k])} \left( \prod_{i \in [t]} Q_i \right) = \sum_{\substack{\kappa : [t] \to 2^{[k]} \text{ s.t.} \\ \bigsqcup_{i \in [t]} \kappa_i = [k]}} \prod_{i \in [t]} \partial_{\mu(\kappa_i)} Q_i .$$

In fact, the product rule yields something general: for any $P \subseteq [k]$, function $\mu : P \to \mathbf{x}$, and $S \subseteq [t]$,

$$\partial_{\mu(P)} \left( \prod_{i \in S} Q_i \right) = \sum_{\substack{\kappa : S \to 2^P \text{ s.t.} \\ \bigsqcup_{i \in S} \kappa_i = P}} \prod_{i \in S} \partial_{\mu(\kappa_i)} Q_i . \tag{7.2}$$

In the above identities we have used $\kappa_i$ as a shorthand for $\kappa(i)$; we shall also do so for the rest of this section.

For an arbitrary $S \subseteq [t]$, recall that we denote $\overline{S} = [t] \setminus S$. Let $\widetilde{\kappa} : \overline{S} \to 2^{[k]}$ be such that $|\widetilde{\kappa}_i| > \frac{k}{d} \cdot d_i$ for all $i \in \overline{S}$. Then we define a polynomial $R_{S,\widetilde{\kappa}}$ as

$$R_{S,\widetilde{\kappa}} := \sum_{\substack{\kappa : [t] \to 2^{[k]} \text{ s.t.} \\ \kappa \text{ extends } \widetilde{\kappa} \\ \bigsqcup_{i \in [t]} \kappa_i = [k] \\ \forall i \in S,\ |\kappa_i| \leq \frac{k}{d} \cdot d_i}} \prod_{i \in [t]} \partial_{\mu(\kappa_i)} Q_i . \tag{7.3}$$

The idea is to express any $k$-th order partial derivative of the product $Q_1 \cdots Q_t$ in terms of $R_{S,\widetilde{\kappa}}$. Indeed we have the following claim.

**Claim 7.1**

$$\partial_{\mu([k])}\left(\prod_{i\in[t]} Q_i\right) = \sum_{S\subseteq[t]} \sum_{\substack{\widetilde{\kappa}:\overline{S}\to 2^{[k]} \text{ s.t.} \\ \forall i\in\overline{S}, |\widetilde{\kappa}_i| > \frac{k}{d}\cdot d_i}} R_{S,\widetilde{\kappa}}.$$

**Proof:**

$$\begin{aligned}
\partial_{\mu([k])}\left(\prod_{i\in[t]} Q_i\right) &= \sum_{\substack{\kappa:[t]\to 2^{[k]} \text{ s.t.} \\ \sqcup_{i\in[t]}\kappa_i=[k]}} \prod_{i\in[t]} \partial_{\mu(\kappa_i)} Q_i \\
&= \sum_{S\subseteq[t]} \sum_{\substack{\kappa:[t]\to 2^{[k]} \text{ s.t.} \\ \sqcup_{i\in[t]}\kappa_i=[k] \\ \left\{i\in[t]:|\kappa_i|\le\frac{k}{d}.d_i\right\}=S}} \prod_{i\in[t]} \partial_{\mu(\kappa_i)} Q_i \\
&= \sum_{S\subseteq[t]} \sum_{\substack{\widetilde{\kappa}:\overline{S}\to 2^{[k]} \text{ s.t.} \\ \forall i\in\overline{S}, |\widetilde{\kappa}_i| > \frac{k}{d}.d_i}} \sum_{\substack{\kappa':S\to 2^{[k]} \text{ s.t.} \\ \kappa=\kappa'\sqcup\widetilde{\kappa} \\ \sqcup_{i\in[t]}\kappa_i=[k] \\ \forall i\in S, |\kappa'_i|\le\frac{k}{d}.d_i}} \prod_{i\in[t]} \partial_{\mu(\kappa_i)} Q_i \\
&= \sum_{S\subseteq[t]} \sum_{\substack{\widetilde{\kappa}:\overline{S}\to 2^{[k]} \text{ s.t.} \\ \forall i\in\overline{S}, |\widetilde{\kappa}_i| > \frac{k}{d}.d_i}} \sum_{\substack{\kappa:[t]\to 2^{[k]} \text{ s.t.} \\ \kappa \text{ extends } \widetilde{\kappa} \\ \sqcup_{i\in[t]}\kappa_i=[k] \\ \forall i\in S, |\kappa_i|\le\frac{k}{d}.d_i}} \prod_{i\in[t]} \partial_{\mu(\kappa_i)} Q_i \\
&= \sum_{S\subseteq[t]} \sum_{\substack{\widetilde{\kappa}:\overline{S}\to 2^{[k]} \text{ s.t.} \\ \forall i\in\overline{S}, |\widetilde{\kappa}_i| > \frac{k}{d}.d_i}} R_{S,\widetilde{\kappa}}. \qquad \text{(by the definition of } R_{S,\widetilde{\kappa}} \text{ in (7.3))}
\end{aligned}$$

□

Hence, to show that $\partial_{\mu([k])}(Q_1\cdots Q_t) \in \mathcal{V}$, it suffices to argue that the polynomials $R_{S,\widetilde{\kappa}}$ are in $\mathcal{V}$. We show this by induction on the size of $S$. In the base case of $|S| = 0$, there does not exist any function $\kappa : [t] \to 2^{[k]}$ that extends $\widetilde{\kappa}$ such that $\left\{i \in [t] : |\kappa_i| \le \frac{k}{d}\cdot d_i\right\} = S$ and $\sqcup_{i\in[t]}\kappa_i = [k]$. This is so because $|\kappa_i| = |\widetilde{\kappa}_i| > \frac{k}{d}\cdot d_i$ for all $i \in [t]$ implies that $\sum_{i\in[t]} |\kappa_i| > \sum_{i\in[t]} \frac{k}{d}\cdot d_i = k$, and hence $\sqcup_{i\in[t]}\kappa_i \neq [k]$. So by definition, $R_{S,\widetilde{\kappa}} = 0 \in \mathcal{V}$.

Suppose that $R_{T,\kappa'} \in \mathcal{V}$ for all $T \subseteq [n]$ such that $|T| < |S|$. Let $\widetilde{\kappa} : \overline{S} \to 2^{[k]}$ be any function

such that $|\widetilde{\kappa}_i| > \frac{k}{d}\cdot d_i$ for all $i \in \overline{S}$, and let $\kappa : [t] \to 2^{[k]}$ be a function that extends $\widetilde{\kappa}$ such that

$$\sqcup_{i\in[t]}\kappa_i = [k] \text{ and } \left\{ i \in [t] : |\kappa_i| \leq \frac{k}{d}\cdot d_i \right\} = S. \tag{7.4}$$

Denoting $\sqcup_{i\in\overline{S}}\kappa_i$ by $P_{\overline{S}}$ and $\sqcup_{i\in S}\kappa_i$ by $P_S$,

$$\begin{aligned}\partial_{\mu(\sqcup_{i\in S}\kappa_i)}\prod_{i\in S}Q_i &= \partial_{\mu(P_S)}\prod_{i\in S}Q_i \\ &= \sum_{\substack{\kappa':S\to 2^{P_S} \text{ s.t.} \\ \sqcup_{i\in S}\kappa'_i = P_S}} \prod_{i\in S}\partial_{\mu(\kappa'_i)}Q_i. \qquad \text{(from Equation (7.2))}\end{aligned}$$

For $U_{S,\kappa} \in \mathbb{F}[\mathbf{x}]$ defined as $U_{S,\kappa} := \left(\partial_{\mu(P_S)}\prod_{i\in S}Q_i\right)\cdot\prod_{i\in\overline{S}}\partial_{\mu(\kappa_i)}Q_i$, we have the following claim.

**Claim 7.2**

$$R_{S,\widetilde{\kappa}} = U_{S,\kappa} - \sum_{\substack{T\subsetneq S \text{ and } \kappa'':S\setminus T\to 2^{P_S} \text{ s.t.} \\ \forall i\in S\setminus T, |\kappa''_i| > \frac{k}{d}\cdot d_i}} R_{T,\kappa''\sqcup\widetilde{\kappa}}.$$

**Proof:**

$$\begin{aligned}U_{S,\kappa} &= \left(\partial_{\mu(P_S)}\prod_{i\in S}Q_i\right)\cdot\prod_{i\in\overline{S}}\partial_{\mu(\kappa_i)}Q_i \\ &= \sum_{\substack{\kappa':S\to 2^{P_S} \text{ s.t.} \\ \sqcup_{i\in S}\kappa'_i = P_S}} \prod_{i\in S}\partial_{\mu(\kappa'_i)}Q_i\cdot\prod_{i\in\overline{S}}\partial_{\mu(\kappa_i)}Q_i \\ &= \sum_{T\subseteq S}\sum_{\substack{\kappa':S\to 2^{P_S} \text{ s.t.} \\ \sqcup_{i\in S}\kappa'_i = P_S \\ \left\{i\in S:|\kappa'_i|\leq\frac{k}{d}.d_i\right\}=T}} \prod_{i\in S}\partial_{\mu(\kappa'_i)}Q_i\cdot\prod_{i\in\overline{S}}\partial_{\mu(\kappa_i)}Q_i \qquad \text{(reordering based on } T\text{)} \\ &= \sum_{T\subseteq S}\sum_{\substack{\kappa'':S\setminus T\to 2^{P_S} \text{ and } \kappa''':T\to 2^{P_S} \text{ s.t.} \\ \kappa'=\kappa''\sqcup\kappa''' \\ \sqcup_{i\in S}\kappa'_i = P_S \\ \forall i\in S\setminus T, |\kappa''_i| > \frac{k}{d}\cdot d_i \\ \forall i\in T, |\kappa'''_i| \leq \frac{k}{d}\cdot d_i}} \prod_{i\in S}\partial_{\mu(\kappa'_i)}Q_i\cdot\prod_{i\in\overline{S}}\partial_{\mu(\kappa_i)}Q_i \\ &= \sum_{T\subseteq S}\sum_{\substack{\kappa'':S\setminus T\to 2^{P_S} \text{ s.t.} \\ \forall i\in S\setminus T, |\kappa''_i| > \frac{k}{d}\cdot d_i}}\sum_{\substack{\kappa''':T\to 2^{P_S} \text{ s.t.} \\ \kappa^*=\kappa''\sqcup\kappa'''\sqcup\widetilde{\kappa} \\ \forall i\in T, |\kappa^*_i| \leq \frac{k}{d}\cdot d_i \\ \sqcup_{i\in T}\kappa'''_i\sqcup\sqcup_{i\in S\setminus T}\kappa''_i = P_S}} \prod_{i\in[t]}\partial_{\mu(\kappa^*_i)}Q_i\end{aligned}$$

(as $\kappa^*$ extends $\kappa' = \kappa'' \sqcup \kappa'''$ and $\kappa$ extends $\widetilde{\kappa}$ )

$$= \sum_{T \subseteq S} \sum_{\substack{\kappa'' : S \setminus T \to 2^{P_S} \text{ s.t.} \\ \forall i \in S \setminus T, |\kappa_i''| > \frac{k}{d} \cdot d_i}} \sum_{\substack{\kappa^* : [t] \to 2^{[k]} \text{ s.t.} \\ \kappa^* = \kappa'' \sqcup \kappa''' \sqcup \widetilde{\kappa} \\ \forall i \in T, |\kappa_i^*| \leq \frac{k}{d} \cdot d_i \\ \sqcup_{i \in [t]} \kappa_i^* = [k]}} \prod_{i \in [t]} \partial_{\mu(\kappa_i^*)} Q_i$$

(because $\sqcup_{i \in [t]} \kappa_i^* = \left( \sqcup_{i \in S \setminus T} \kappa_i'' \sqcup \sqcup_{i \in T} \kappa_i''' \right) \sqcup \sqcup_{i \in \overline{S}} \widetilde{\kappa}_i = P_S \sqcup \sqcup_{i \in \overline{S}} \kappa_i = \sqcup_{i \in [t]} \kappa_i = [k]$ from (7.4))

$$= \sum_{T \subseteq S} \sum_{\substack{\kappa'' : S \setminus T \to 2^{P_S} \text{ s.t.} \\ \forall i \in S \setminus T, |\kappa_i''| > \frac{k}{d} \cdot d_i}} R_{T, \kappa'' \sqcup \widetilde{\kappa}}$$

($R_{T,\kappa'' \sqcup \widetilde{\kappa}}$ is well-defined because $\kappa^*$ extends $\kappa'' \sqcup \widetilde{\kappa}$ and (7.4))

$$= R_{S,\widetilde{\kappa}} + \sum_{\substack{T \subsetneq S \text{ and } \kappa'' : S \setminus T \to 2^{P_S} \text{ s.t.} \\ \forall i \in S \setminus T, |\kappa_i''| > \frac{k}{d} . d_i}} R_{T, \kappa'' \sqcup \widetilde{\kappa}}. \qquad \text{(separating out the case } T = S\text{)}$$

□

When $T \subsetneq S$, by the induction hypothesis, all the terms $R_{T,\kappa'' \sqcup \widetilde{\kappa}}$ in the above expression are in $\mathcal{V}$. Therefore, to conclude that $R_{S,\widetilde{\kappa}} \in \mathcal{V}$, it suffices to show that $U_{S,\kappa} \in \mathcal{V}$. From its definition, note that $U_{S,\kappa} \in \left\langle \partial^{k_0} \left( \prod_{i \in S} Q_i \right) \cdot \mathbf{x}^{\ell_0} \right\rangle$ where $k_0 := |\mu(P_S)| = |P_S| = \sum_{i \in S} |\kappa_i|$ and $\ell_0 := \sum_{i \in \overline{S}} \deg(\partial_{\mu(\kappa_i)} Q_i) = \sum_{i \in \overline{S}} (d_i - |\kappa_i|)$. Also,

$$\begin{aligned} k - k_0 - \frac{k}{d-k} \cdot \ell_0 &= k - \sum_{i \in S} |\kappa_i| - \frac{k}{d-k} \cdot \sum_{i \in \overline{S}} (d_i - |\kappa_i|) \\ &= \sum_{i \in \overline{S}} |\kappa_i| - \frac{k}{d-k} \cdot \sum_{i \in \overline{S}} (d_i - |\kappa_i|) \\ &\qquad \text{(as from (7.4), } \kappa_1, \ldots, \kappa_t \text{ form a partition of } [k]\text{)} \\ &= \sum_{i \in \overline{S}} |\kappa_i| - \frac{k}{d-k} \cdot (d_i - |\kappa_i|) \end{aligned}$$

$$
\begin{aligned}
&= \sum_{i\in\overline{S}} \frac{d}{d-k}\cdot\left(|\kappa_i| - \frac{k}{d}\cdot d_i\right)\\
&\geq \sum_{i\in\overline{S}} |\kappa_i| - \frac{k}{d}.d_i && \text{(using } d \geq d-k \text{ and } |\kappa_i| > \tfrac{k}{d}\cdot d_i \text{ iff } i\in\overline{S}\text{)}\\
&= \frac{1}{2}\left(\sum_{i\in\overline{S}} |\kappa_i| - \frac{k}{d}\cdot d_i\right) + \frac{1}{2}\left(\sum_{i\in S} |\kappa_i| - \frac{k}{d}\cdot d_i\right)\\
&\quad + \frac{1}{2}\left(\sum_{i\in\overline{S}} |\kappa_i| - \frac{k}{d}\cdot d_i\right) - \frac{1}{2}\left(\sum_{i\in S} |\kappa_i| - \frac{k}{d}\cdot d_i\right)\\
&= \frac{1}{2}\left(\sum_{i\in[t]} |\kappa_i| - \frac{k}{d}\cdot d_i\right) + \frac{1}{2}\left(\sum_{i\in\overline{S}} |\kappa_i| - \frac{k}{d}\cdot d_i\right) - \frac{1}{2}\left(\sum_{i\in S} |\kappa_i| - \frac{k}{d}\cdot d_i\right)\\
&= \frac{1}{2}\left(k - \frac{k}{d}\cdot d\right) + \frac{1}{2}\left(\sum_{i\in\overline{S}} |\kappa_i| - \frac{k}{d}\cdot d_i\right) - \frac{1}{2}\left(\sum_{i\in S} |\kappa_i| - \frac{k}{d}\cdot d_i\right)\\
&&& \text{(since } |\kappa_i|\text{'s sum to } k \text{ and } d_i\text{'s sum to } d\text{)}\\
&= \frac{1}{2}\cdot\sum_{i\in[t]}\left||\kappa_i| - \frac{k}{d}\cdot d_i\right| && \text{(from (7.4))}\\
&\geq \mathsf{residue}_k(d_1,\ldots,d_t). && \text{(from definition of residue)}
\end{aligned}
$$

Hence, $U_{S,\kappa} \in \left\langle \mathbf{x}^{\ell_0}\cdot\boldsymbol{\partial}^{k_0}\left(\prod_{i\in S} Q_i\right)\right\rangle \subseteq \mathcal{V}$ as $k_0 + \frac{k}{d-k}\cdot\ell_0 \leq k - \mathsf{residue}_k(d_1,\ldots,d_t)$. □

We now use the above lemma to upper bound the shifted partials and affine projections of partials measures of a product of polynomials.

**Lemma 7.4 (Upper bounding SP and APP of a product)** *Let $Q = Q_1\cdots Q_t$ be a homogeneous polynomial in $\mathbb{F}[x_1,\ldots,x_n]$ of degree $d = d_1+\cdots+d_t \geq 1$, where $Q_i$ is homogeneous and $d_i := \deg(Q_i)$ for $i\in[t]$. Then, for any non-negative integers $k<d$, $\ell\geq 0$, and $n_0\leq n$,*

1. $$\mathsf{SP}_{k,\ell}(Q) \leq 2^t\cdot d^2\cdot \max_{\substack{k_0,\ell_0\geq 0\\ k_0+\frac{k}{d-k}\cdot\ell_0 \leq k-\mathsf{residue}_k(d_1,\ldots,d_t)}} M(n,k_0)\cdot M(n,\ell_0+\ell),$$

2. $$\mathsf{APP}_{k,n_0}(Q) \leq 2^t\cdot d^2\cdot \max_{\substack{k_0,\ell_0\geq 0\\ k_0+\frac{k}{d-k}\cdot\ell_0 \leq k-\mathsf{residue}_k(d_1,\ldots,d_t)}} M(n,k_0)\cdot M(n_0,\ell_0).$$

**Proof:** We will first upper bound the shifted partials measure. From Lemma 7.3, we know that

$$\left\langle \partial^k \left(Q_1 \cdots Q_t\right) \right\rangle \subseteq \sum_{\substack{S \subseteq [t];\ k_0, \ell_0 \geq 0 \\ k_0 + \frac{k}{d-k} \cdot \ell_0 \leq k - \mathsf{residue}_k(d_1, \ldots, d_t)}} \left\langle \mathbf{x}^{\ell_0} \cdot \partial^{k_0} \left( \prod_{i \in S} Q_i \right) \right\rangle.$$

Hence,

$$\left\langle \mathbf{x}^{\ell} \cdot \partial^k \left(Q_1 \cdots Q_t\right) \right\rangle \subseteq \sum_{\substack{S \subseteq [t];\ k_0, \ell_0 \geq 0 \\ k_0 + \frac{k}{d-k} \cdot \ell_0 \leq k - \mathsf{residue}_k(d_1, \ldots, d_t)}} \left\langle \mathbf{x}^{\ell_0 + \ell} \cdot \partial^{k_0} \left( \prod_{i \in S} Q_i \right) \right\rangle. \tag{7.5}$$

For a fixed $S \subseteq [t]$ and $k_0, \ell_0$, since $\left\langle \mathbf{x}^{\ell_0+\ell} \cdot \partial^{k_0} \left( \prod_{i \in S} Q_i \right) \right\rangle \subseteq \left\langle \mathbf{x}^{\ell_0+\ell} \right\rangle \cdot \left\langle \partial^{k_0} \cdot \left( \prod_{i \in S} Q_i \right) \right\rangle$, and $\dim \left\langle \mathbf{x}^{\ell_0+\ell} \right\rangle \leq \left| \mathbf{x}^{\ell_0+\ell} \right| = M(n, \ell_0 + \ell)$ and $\dim \left\langle \partial^{k_0} \left( \prod_{i \in S} Q_i \right) \right\rangle \leq \left| \partial^{k_0} \left( \prod_{i \in S} Q_i \right) \right| \leq \left| \mathbf{x}^{k_0} \right| = M(n, k_0)$, we have,

$$\dim \left\langle \mathbf{x}^{\ell_0+\ell} \cdot \partial^{k_0} \left( \prod_{i \in S} Q_i \right) \right\rangle \leq \dim \left\langle \mathbf{x}^{\ell_0+\ell} \right\rangle \cdot \dim \left\langle \partial^{k_0} \left( \prod_{i \in S} Q_i \right) \right\rangle \leq M(n, \ell_0 + \ell) \cdot M(n, k_0).$$

Adding up the above upper bound over all the $2^t \cdot d^2$ possible combinations of $S \subseteq [t]$, $k_0 \in [0..k]$, and $\ell_0 \in [0..(d-k)]$ in (7.5), we get,

$$\mathsf{SP}_{k,\ell}(Q) = \dim \left\langle \mathbf{x}^{\ell} \cdot \partial^k \left(Q_1 \cdots Q_t\right) \right\rangle \leq 2^t \cdot d^2 \cdot \max_{\substack{k_0, \ell_0 \geq 0 \\ k_0 + \frac{k}{d-k} \cdot \ell_0 \leq k - \mathsf{residue}_k(d_1, \ldots, d_t)}} M(n, k_0) \cdot M(n, \ell_0 + \ell).$$

The details for an upper bound on APP are similar.

$$\mathsf{APP}_{k,n_0}(Q) = \max_{L : \mathbf{x} \to \langle \mathbf{z} \rangle} \dim \left\langle \pi_L \left( \partial^k (Q_1 \cdots Q_t) \right) \right\rangle$$

$$\leq \max_{L:\mathbf{x}\to\langle\mathbf{z}\rangle} \dim\left\langle \pi_L\left(\sum_{\substack{S\subseteq[t];\, k_0,\ell_0\geq 0\\ k_0+\frac{k}{d-k}\cdot\ell_0\leq k-\mathsf{residue}_k(d_1,\ldots,d_t)}} \left\langle \mathbf{x}^{\ell_0}\cdot\boldsymbol{\partial}^{k_0}\left(\prod_{i\in S}Q_i\right)\right\rangle\right)\right\rangle$$

(from Lemma 7.3)

$$\leq \max_{L:\mathbf{x}\to\langle\mathbf{z}\rangle} \dim\left\langle \sum_{\substack{S\subseteq[t];\, k_0,\ell_0\geq 0\\ k_0+\frac{k}{d-k}\cdot\ell_0\leq k-\mathsf{residue}_k(d_1,\ldots,d_t)}} \left\langle \pi_L\left(\mathbf{x}^{\ell_0}\cdot\boldsymbol{\partial}^{k_0}\left(\prod_{i\in S}Q_i\right)\right)\right\rangle\right\rangle$$

(as $\pi_L$ distributes over addition)

$$\leq \max_{L:\mathbf{x}\to\langle\mathbf{z}\rangle} \sum_{\substack{S\subseteq[t];\, k_0,\ell_0\geq 0\\ k_0+\frac{k}{d-k}\cdot\ell_0\leq k-\mathsf{residue}_k(d_1,\ldots,d_t)}} \dim\left\langle \pi_L\left(\mathbf{x}^{\ell_0}\right)\cdot\pi_L\left(\boldsymbol{\partial}^{k_0}\left(\prod_{i\in S}Q_i\right)\right)\right\rangle$$

(Using $\pi_L$ distributes over multiplication)

$$\leq \max_{L:\mathbf{x}\to\langle\mathbf{z}\rangle} \sum_{\substack{S\subseteq[t];\, k_0,\ell_0\geq 0\\ k_0+\frac{k}{d-k}\cdot\ell_0\leq k-\mathsf{residue}_k(d_1,\ldots,d_t)}} \dim\left\langle \pi_L\left(\mathbf{x}^{\ell_0}\right)\right\rangle\cdot\dim\left\langle \pi_L\left(\boldsymbol{\partial}^{k_0}\left(\prod_{i\in S}Q_i\right)\right)\right\rangle$$

$$\leq \max_{L:\mathbf{x}\to\langle\mathbf{z}\rangle} \sum_{\substack{S\subseteq[t];\, k_0,\ell_0\geq 0\\ k_0+\frac{k}{d-k}\cdot\ell_0\leq k-\mathsf{residue}_k(d_1,\ldots,d_t)}} \left|\pi_L\left(\mathbf{x}^{\ell_0}\right)\right|\cdot\left|\pi_L\left(\boldsymbol{\partial}^{k_0}\left(\prod_{i\in S}Q_i\right)\right)\right|$$

$$\leq \max_{L:\mathbf{x}\to\langle\mathbf{z}\rangle} \sum_{\substack{S\subseteq[t];\, k_0,\ell_0\geq 0\\ k_0+\frac{k}{d-k}\cdot\ell_0\leq k-\mathsf{residue}_k(d_1,\ldots,d_t)}} \left|\mathbf{z}^{\ell_0}\right|\cdot\left|\pi_L\left(\boldsymbol{\partial}^{k_0}\left(\prod_{i\in S}Q_i\right)\right)\right|$$

(as $L$ is a map from $\mathbf{x}$ to $\langle\mathbf{z}\rangle$, $\pi_L(m)\in\mathbf{z}^{\ell_0}$ for any monomial $m$ over $\mathbf{x}$ of degree $\ell_0$)

$$\leq 2^t\cdot d^2\cdot \max_{\substack{k_0,\ell_0\geq 0\\ k_0+\frac{k}{d-k}\cdot\ell_0\leq k-\mathsf{residue}_k(d_1,\ldots,d_t)}} M(n_0,\ell_0)\cdot M(n,k_0).$$

□

## 7.5 Lower bound for low-depth homogeneous formulas

In this section, we present a superpolynomial lower bound for "low-depth" homogeneous formulas computing the *IMM* and *NW* polynomials. We begin by proving a structural result for homogeneous formulas.

### 7.5.1 Decomposition of low-depth formulas

We show that any homogeneous formula can be decomposed as a sum of products of homogeneous polynomials of lower degrees, where the number of summands is bounded by the number of gates in the original formula. The decomposition lemma given below bears some resemblance to a decomposition of homogeneous formulas in [HY11]. In the decomposition in [HY11], the degrees of the factors of every summand roughly form a geometric sequence, and hence each summand is a product of a 'large' number of factors. Here we show that each summand has 'many' low-degree factors. While the lower bound argument in [LST21] does not explicitly make use of such a decomposition, their inductive argument can be formulated as a depth-reduction or decomposition lemma (with slightly different thresholds for the degrees).

**Lemma 7.5 (Decomposition of low-depth formulas)** *Suppose $C$ is a homogeneous formula of product-depth $\Delta \geq 1$ computing a homogeneous polynomial in $\mathbb{F}[x_1, \ldots, x_n]$ of degree* at least *$d > 0$. Then, there exist homogeneous polynomials $\{Q_{i,j}\}_{i,j}$ in $\mathbb{F}[x_1, \ldots, x_n]$ such that*

1. $C = \sum_{i=1}^{s} Q_{i,1} \cdots Q_{i,t_i}$, *for some* $s \leq \text{size}(C)$, *and*
2. *for all* $i \in [s]$, *either*

$$\left|\left\{j \in [t_i] : \deg(Q_{i,j}) = 1\right\}\right| \geq d^{2^{1-\Delta}}, \text{ or}$$

$$\left|\left\{j \in [t_i] : \deg(Q_{i,j}) \approx_2 d^{2^{1-\delta}}\right\}\right| \geq d^{2^{1-\delta}} - 1 \text{ , for some } \delta \in [2..\Delta].$$

**Proof:** The decomposition is constructed inductively – at addition gates, we simply add the decompositions of the smaller sub-formulas, whereas the multiplication gates need to be handled more carefully. Consider a multiplication gate $Q_1 \times \cdots \times Q_t$. If all the factors ($Q_i$'s) have 'low' degrees, we use this expression directly to construct the decomposition. Otherwise, we go deeper into a factor which has a 'large' degree, but do not expand the other factors. The thresholds to decide whether a factor is of 'low' degree may appear arbitrary (and are indeed so) for this lemma, but we fix them to be $d^{2^{1-\delta}}$ for $\delta \in [2..\Delta]$ as these give us the desired lower bounds.

Without loss of generality, we may assume that $C$ has alternate layers of addition and multiplication gates. Further, we can assume that the degrees of the polynomials computed by all the multiplication gates that feed into an addition gate are the same as the degree of the polynomial computed by that addition gate. This is so because disconnecting all the

multiplication gates that compute polynomials of other degrees does not affect the polynomial computed by the addition gate. Also, for brevity, we will ignore the edge weights in $C$, i.e., we assume that all the field constants on the edges are equal to 1. As scaling with constant factors does not affect the homogeneity of polynomials, this is a valid assumption. Let

$$C = \sum_{i=1}^{u} C_i \text{ , and for } i \in [u],\ C_i = \prod_{j=1}^{u_i} C_{i,j},$$

where $u$ and $\{u_i\}_i$ are integers and $\left\{C_{i,j}\right\}_{i,j}$ are (homogeneous) sub-formulas of $C$ of product-depth $\Delta - 1$. The proof of this lemma is by induction on the product-depth. For $\Delta = 1$, for all $i \in [u]$, we have $u_i \geq d$ and for all $j \in [d]$, $\deg(C_{i,j}) = 1$ , so both the conditions in the lemma statement are met for $Q_{i,j} := C_{i,j}$.

Suppose that the lemma is true for all homogeneous formulas of product-depth at most $\Delta - 1$, $\Delta \geq 2$. For a formula $C$ with product-depth $\Delta$, we consider the term $C_{i,1} \cdots C_{i,u_i}$ for an arbitrary $i \in [u]$ and analyze the following two cases.

**Case 1:** There exists some $j^* \in [u_i]$ such that $\deg(C_{i,j^*}) \geq \sqrt{d}$. As the product-depth of $C_{i,j^*}$ is at most $\Delta - 1$, we have the following expression for the polynomial computed by $C_{i,j^*}$ from the induction hypothesis:

$$C_{i,j^*} = \sum_{\tilde{i}=1}^{\tilde{s}_i} \widetilde{Q}_{i,\tilde{i},1} \cdots \widetilde{Q}_{i,\tilde{i},\tilde{t}_{\tilde{i}}}\,, \tag{7.6}$$

where

$$\tilde{s}_i \leq \text{size}(C_{i,j^*}) \leq \text{size}(C_i), \tag{7.7}$$

and $\left\{\widetilde{Q}_{i,\tilde{i},\tilde{j}}\right\}_{i,\tilde{i},\tilde{j}}$ are homogeneous polynomials such that for all $\tilde{i} \in \tilde{s}_i$, either

$$\left|\left\{\tilde{j} \in [\tilde{t}_{\tilde{i}}] : \deg(\widetilde{Q}_{i,\tilde{i},\tilde{j}}) = 1\right\}\right| \geq \sqrt{d}^{2^{1-(\Delta-1)}} = d^{2^{1-\Delta}} \text{ , or} \tag{7.8}$$

$$\left|\left\{\tilde{j} \in [\tilde{t}_{\tilde{i}}] : \deg(\widetilde{Q}_{i,\tilde{i},\tilde{j}}) \approx_2 \sqrt{d}^{2^{1-\delta}}\right\}\right| \geq \sqrt{d}^{2^{1-\delta}} - 1 \text{ , for some } \delta \in [2..(\Delta-1)]. \tag{7.9}$$

Note that since $\sqrt{d}^{2^{1-\delta}} = d^{2^{1-(\delta+1)}}$, (7.9) is equivalent to

$$\left|\left\{\tilde{j} \in [\tilde{t}_{\tilde{i}}] : \deg(\widetilde{Q}_{i,\tilde{i},\tilde{j}}) \approx_2 d^{2^{1-\delta}}\right\}\right| \geq d^{2^{1-\delta}} - 1 \text{ , for some } \delta \in [3..\Delta]. \tag{7.10}$$

Indeed, when $\Delta = 2$, the above scenario never arises and the number of linear factors is

'large', i.e., (7.8) holds. Denoting $\prod_{j\in[u_i]\setminus\{j^*\}} C_{i,j}$ by $D_{i,j^*}$ and using (7.6), we have

$$C_i = C_{i,1}\cdots C_{i,u_i} = C_{i,j^*}.D_{i,j^*} = \sum_{\tilde{i}=1}^{\tilde{s}_i} \widetilde{Q}_{i,\tilde{i},1}\cdots\widetilde{Q}_{i,\tilde{i},\tilde{t}_{\tilde{i}}}\cdot D_{i,j^*}. \tag{7.11}$$

Thus, we are able to decompose the sub-formula $C_i$ as a sum of at most size$(C_i)$ many products.

**Case 2:** For all $j\in[u_i]$, $\deg(C_{i,j}) < \sqrt{d}$. Consider the polynomials computed by $C_{i,1},\ldots,C_{i,u_i}$. Suppose there exists $j_1\neq j_2\in[u_i]$ such that $\deg(C_{i,j_1}) < \frac{\sqrt{d}}{2}$ and $\deg(C_{i,j_2}) < \frac{\sqrt{d}}{2}$. Then $\deg(C_{i,j_1}\cdot C_{i,j_2}) < \sqrt{d}$. By repeatedly combining such low degree factors, we can express $C_i = C_{i,1}\cdots C_{i,u_i}$ as

$$C_i = D_{i,1}\cdots D_{i,v_i}\,, \tag{7.12}$$

where $\{D_{i,j}\}_{i,j}$ are homogeneous polynomials such that for all $j\in[v_i]$, we have $\deg(D_{i,j}) < \sqrt{d}$ and there exists at most one index $j^*\in[v_i]$ such that $\deg(D_{i,j^*}) < \frac{\sqrt{d}}{2}$. In other words, for at least $v_i-1$ indices $j\in[v_i]$, $\deg(D_{i,j}) \approx_2 \sqrt{d}$. Using the fact that $C$ is a homogeneous formula,

$$d \le \deg(C) = \deg(C_i) = \sum_{j=1}^{v_i}\deg(D_{i,j}) \le v_i\cdot\sqrt{d}.$$

Therefore, the number of indices $j\in[v_i]$ such that $\deg(D_{i,j}) \approx_2 \sqrt{d}$ is at least $v_i - 1 \ge \sqrt{d}-1$. In other words,

$$\left|\left\{j\in[v_i] : \deg(D_{i,j}) \approx_2 d^{2^{1-\delta}}\right\}\right| \ge d^{2^{1-\delta}} - 1 \ \text{, for } \delta = 2. \tag{7.13}$$

Now, expressing $C_i$ for each $i\in[u]$ using (7.11) if $i$ falls under Case 1, and using (7.12) if $i$ falls under Case 2, we get

$$C = \sum_{i=1}^{u} C_i = \sum_{i=1}^{s} Q_{i,1}\cdots Q_{i,t_i},$$

for polynomials $\{Q_{i,j}\}_{i,j}$ that are defined appropriately based on $\left\{\widetilde{Q}_{i,\tilde{i},\tilde{j}}\right\}_{i,\tilde{i},\tilde{j}}$ and $\{D_{i,j}\}_{i,j}$. Using (7.7) and (7.12), we get that the number of terms is $s \le \sum_{i=1}^{u}\text{size}(C_i) \le \text{size}(C)$. Item 2 in the lemma statement directly follows from (7.8), (7.13), or (7.10). □

### 7.5.2 Low-depth formulas have high residue

The following lemma gives us a value for the order of derivatives $k$ with respect to which low-depth formulas yield high residue. Its proof uses Lemma 7.5.

**Lemma 7.6 (Low-depth formulas have high residue)** *Suppose $C$ is a homogeneous formula of product-depth $\Delta \geq 1$ computing a polynomial in $\mathbb{F}[x_1, \ldots, x_n]$ of degree $d$, where $d^{2^{1-\Delta}} = \omega(1)$. Then, there exist homogeneous polynomials $\{Q_{i,j}\}_{i,j}$ in $\mathbb{F}[x_1, \ldots, x_n]$ such that $C = \sum_{i=1}^{s} Q_{i,1} \cdots Q_{i,t_i}$, for some $s \leq \mathsf{size}(C)$. Fixing an arbitrary $i \in [s]$, let $t := t_i$ and define $d_j := \deg(Q_{i,j})$ for $j \in [t]$. Then, $\mathsf{residue}_k(d_1, \ldots, d_t) \geq \Omega\left(d^{2^{1-\Delta}}\right)$, where $k := \left\lfloor \frac{\alpha \cdot d}{1+\alpha} \right\rfloor$, $\alpha := \sum_{\nu=0}^{\Delta-1} \frac{(-1)^\nu}{\tau^{2^\nu - 1}}$, and $\tau := \left\lfloor d^{2^{1-\Delta}} \right\rfloor$.*

**Proof:** We will show that the decomposition proven in Lemma 7.5 itself satisfies the required property. We first establish a range for the value of $k$ (and $\alpha$) given in the lemma statement. We have $\alpha \leq 1$ and

$$\alpha \geq \sum_{\nu=0}^{1} \frac{(-1)^\nu}{\tau^{2^\nu - 1}} = 1 - \frac{1}{\tau} = 1 - \frac{1}{\left\lfloor d^{2^{1-\Delta}} \right\rfloor} \geq \frac{1}{2}.$$

Hence, $k \in \left[\left\lfloor \frac{d}{3} \right\rfloor, \frac{d}{2}\right] \subseteq \left[\frac{d}{4}, \frac{d}{2}\right]$ because $d = \omega(1)$. As $C$ computes a polynomial of degree $d \geq \tau^{2^{\Delta-1}}$, we can apply Item 2 of Lemma 7.5 to $C$ using $\tau^{2^{\Delta-1}}$ (rather than $d$) as the threshold. Thus, we have that at least one of the following two cases will hold.

**Case 1:** $\left|\{j \in [t] : d_j = 1\}\right| \geq \left(\tau^{2^{\Delta-1}}\right)^{2^{1-\Delta}} = \tau$. Then,

$$\begin{aligned}
\mathsf{residue}_k(d_1, \ldots, d_t) &= \frac{1}{2} \cdot \min_{k_1, \ldots, k_t \in \mathbb{Z}} \sum_{j \in [t]} \left| k_j - \frac{k}{d} \cdot d_j \right| \\
&\geq \frac{1}{2} \cdot \sum_{j \in [t]} \min\left\{ \left\{ \frac{k}{d} \cdot d_j \right\}, 1 - \left\{ \frac{k}{d} \cdot d_j \right\} \right\} \\
&\geq \frac{1}{2} \cdot \sum_{j \in [t] : d_j = 1} \min\left\{ \left\{ \frac{k}{d} \cdot d_j \right\}, 1 - \left\{ \frac{k}{d} \cdot d_j \right\} \right\} \\
&\geq \frac{1}{2} \cdot \left|\{j \in [t] : d_j = 1\}\right| \cdot \min\left\{ \left\{ \frac{k}{d} \right\}, 1 - \left\{ \frac{k}{d} \right\} \right\} \\
&\geq \tau/8. && (\text{as } k/d \in [1/4, 1/2])
\end{aligned}$$

**Case 2:** $\left|\left\{j \in [t] : d_j \approx_2 \left(\tau^{2^{\Delta-1}}\right)^{2^{1-\delta}}\right\}\right| \geq \left(\tau^{2^{\Delta-1}}\right)^{2^{1-\delta}} - 1$ for some $\delta \in [2..\Delta]$ (this case cannot

occur when $\Delta < 2$). Equivalently, there exists a $\delta \in [0..(\Delta-2)]$ such that

$$\left|\left\{j \in [t] : d_j \approx_2 \tau^{2^\delta}\right\}\right| \geq \tau^{2^\delta} - 1.$$

Let $k_1, \ldots, k_t$ be arbitrary non-negative integers such that $k_j \leq d_j$ for all $j \in [t]$. Then for any $j \in [t]$ such that $d_j \approx_2 \tau^{2^\delta}$, we have

$$\begin{aligned}
\tau^{2^\delta-1} \cdot \left|k_j - \frac{k \cdot d_j}{d}\right| &= \tau^{2^\delta-1} \cdot \left|k_j - \frac{d_j}{d} \cdot \left\lfloor \frac{\alpha \cdot d}{1+\alpha} \right\rfloor\right| \\
&\geq \tau^{2^\delta-1} \cdot \left(\left|k_j - \frac{d_j}{d} \cdot \frac{\alpha \cdot d}{1+\alpha}\right| - \frac{d_j}{d} \cdot \left\{\frac{\alpha \cdot d}{1+\alpha}\right\}\right) \\
&\geq \tau^{2^\delta-1} \cdot \left|k_j - \frac{\alpha \cdot d_j}{1+\alpha}\right| - \tau^{2^\delta-1} \cdot \frac{d_j}{d} \\
&\geq \tau^{2^\delta-1} \cdot \left|k_j - \frac{\alpha \cdot d_j}{1+\alpha}\right| - \frac{\tau^{2^\delta-1} \cdot \tau^{2^\delta}}{d} && \text{(since } d_j \approx_2 \tau^{2^\delta}\text{)} \\
&\geq \tau^{2^\delta-1} \cdot \left|k_j - \frac{\alpha \cdot d_j}{1+\alpha}\right| - \frac{\left(d^{2^{1-\Delta}}\right)^{2^{\delta+1}-1}}{d} \\
&\geq \tau^{2^\delta-1} \cdot \left|k_j - \frac{\alpha \cdot d_j}{1+\alpha}\right| - \frac{1}{d^{2^{1-\Delta}}} && \text{(as } \delta \leq \Delta - 2\text{)} \\
&= \tau^{2^\delta-1} \cdot \left|k_j - \frac{\alpha \cdot d_j}{1+\alpha}\right| - o(1) \\
& && \text{(if } d^{2^{1-\Delta}} = O(1)\text{, then the lemma is not interesting)} \\
&\geq \frac{1}{2} \cdot \tau^{2^\delta-1} \cdot \left|k_j - \alpha \cdot (d_j - k_j)\right| - o(1) && \text{(as } \alpha \leq 1\text{)} \quad (7.14)
\end{aligned}$$

We use the following claim. For $j \in [t]$, let $m_j := d_j - k_j$, note that $m_j$ is a non-negative integer.

**Claim 7.3** $\eta := \tau^{2^\delta-1} \cdot \left|k_j - \alpha \cdot m_j\right| \geq \Omega(1)$.

**Proof:** We prove the claim by analysing the following three sub-cases.

**Case (i):** $m_j < k_j$. Then, $\eta \geq \left|k_j - \alpha \cdot m_j\right| = k_j - \alpha \cdot m_j \geq k_j - m_j \geq 1$.

Now, let $\alpha_1 := \sum_{\nu=0}^{\delta} \frac{(-1)^\nu}{\tau^{2^\nu-1}}$, $\alpha_2 := \frac{(-1)^{\delta+1}}{\tau^{2^{\delta+1}-1}}$ and $\alpha_3 := \sum_{\nu=\delta+2}^{\Delta-1} \frac{(-1)^\nu}{\tau^{2^\nu-1}}$. Then, let $\alpha_4 := \tau^{2^\delta-1} \cdot \left(k_j - m_j \cdot \alpha_1\right)$. Noting that $\alpha = \alpha_1 + \alpha_2 + \alpha_3$ we have,

$$\begin{aligned}
\eta = \tau^{2^\delta-1}\cdot\left|k_j - \alpha\cdot m_j\right| &= \left|\tau^{2^\delta-1}\cdot k_j - \tau^{2^\delta-1}\cdot m_j\cdot(\alpha_1+\alpha_2+\alpha_3)\right| \\
&\qquad\qquad \text{(as } \alpha = \alpha_1+\alpha_2+\alpha_3 \text{ by definition)} \\
&\geq \left|\alpha_4 - \tau^{2^\delta-1}\cdot m_j\cdot\frac{(-1)^{\delta+1}}{\tau^{2^{\delta+1}-1}} - \tau^{2^\delta-1}\cdot m_j\cdot\alpha_3\right| \\
&\geq \left|\alpha_4 - \tau^{2^\delta-1}\cdot m_j\cdot\frac{(-1)^{\delta+1}}{\tau^{2^{\delta+1}-1}}\right| - \left|\tau^{2^\delta-1}\cdot m_j\cdot\alpha_3\right| \\
&\geq \left|\left|\alpha_4\right| - \frac{m_j}{\tau^{2^\delta}}\right| - \left|\tau^{2^\delta-1}\cdot m_j\cdot\alpha_3\right| \\
&= \left|\left|\alpha_4\right| - \frac{m_j}{\tau^{2^\delta}}\right| - \left|\sum_{\nu=\delta+2}^{\Delta-1}\cdot\frac{(-1)^\nu\cdot\tau^{2^\delta-1}\cdot m_j}{\tau^{2^\nu-1}}\right| \\
&\geq \left|\left|\alpha_4\right| - \frac{m_j}{\tau^{2^\delta}}\right| - \left|\frac{\tau^{2^\delta-1}\cdot m_j}{\tau^{2^{\delta+2}-1}}\right| \\
&\qquad\qquad \text{(taking only the leading term of the summation)} \\
&\geq \left|\left|\alpha_4\right| - \frac{m_j}{\tau^{2^\delta}}\right| - \left|\frac{\tau^{2^\delta-1}\cdot\tau^{2^\delta}}{\tau^{2^{\delta+2}-1}}\right| \qquad \text{(since } m_j \leq d_j \approx_2 \tau^{2^\delta}\text{)} \\
&\geq \left|\left|\alpha_4\right| - \frac{m_j}{\tau^{2^\delta}}\right| - \frac{1}{\tau^2} \\
&= \left|\left|\alpha_4\right| - \frac{m_j}{\tau^{2^\delta}}\right| - o(1).
\end{aligned}$$

Notice that, as $m_j \leq d_j \approx_2 \tau^{2^\delta}$, $\frac{m_j}{\tau^{2^\delta}} \leq 1$.

**Case (ii):** $k_j \leq m_j \leq 6\cdot k_j$. Note that

$$m_j = \frac{6\cdot m_j + m_j}{7} \leq \frac{6\cdot m_j + 6\cdot k_j}{7} = \frac{6}{7}\cdot d_j \leq \frac{6}{7}\cdot\tau^{2^\delta}, \text{ and}$$

$$m_j \geq \frac{m_j + k_j}{2} = \frac{d_j}{2} \geq \frac{1}{4}\cdot\tau^{2^\delta}.$$

Thus $\frac{m_j}{\tau^{2^\delta}} \in \left[\frac{1}{4}, \frac{6}{7}\right]$. On the other hand, $\alpha_4 = \tau^{2^\delta-1}\cdot k_j - \tau^{2^\delta-1}\cdot m_j\cdot\alpha_1$ is an integer since the denominators of all the terms in $\alpha_1$ divide $\tau^{2^\delta-1}$. Therefore $\frac{m_j}{\tau^{2^\delta}}$ is at least $\min\{1/4, 3/4, 6/7, 1/7\} = 1/7$ distance from any integer, and from $|\alpha_4|$ in particular. That is, $\left|\left|\alpha_4\right| - \frac{m_j}{\tau^{2^\delta}}\right| \geq 1/7$ and

$\eta \geq \left||\alpha_4| - \frac{m_j}{\tau^{2^\delta}}\right| - o(1) \geq \Omega(1)$.

**Case (iii):** $m_j > 6 \cdot k_j$. Then,

$$\begin{aligned}
-k_j + m_j \cdot \alpha_1 &= -k_j + m_j \cdot \left( \sum_{\nu=0}^{\delta} \frac{(-1)^\nu}{\tau^{2^\nu - 1}} \right) \\
&= -k_j + m_j - m_j \cdot \left( \sum_{\nu=1}^{\delta} \frac{(-1)^{\nu-1}}{\tau^{2^\nu - 1}} \right) \\
&\geq -k_j + m_j - \frac{m_j}{\tau} \\
&\geq \frac{m_j}{2} - k_j && \text{(as } \tau = \omega(1)\text{)} \\
&\geq \frac{m_j}{2} - \frac{m_j}{6} = \frac{m_j}{3} \geq \frac{2}{7} \cdot (m_j + k_j) = \frac{2 \cdot d_j}{7} \geq \frac{\tau}{7} \geq 2.
\end{aligned}$$

Hence, $\left||\alpha_4| - \frac{m_j}{\tau^{2^\delta}}\right| = \left|\tau^{2^\delta - 1} \cdot (-k_j + m_j \cdot \alpha_1) - \frac{m_j}{\tau^{2^\delta}}\right| \geq 2 - 1 = 1$ and $\eta \geq \Omega(1)$. □

Let $k_1, \ldots, k_t \in \mathbb{Z}$ be the such that $\sum_{i=1}^{t} \left| k_i - \frac{k}{d} \cdot d_i \right|$ is minimised. Hence,

$$\begin{aligned}
\mathsf{residue}_k(d_1, \ldots, d_t) &\geq \frac{1}{2} \sum_{j \in [t] : d_j \approx_2 \tau^{2^\delta}} \left| k_j - \frac{k}{d} \cdot d_j \right| \\
&\geq \frac{1}{2} \cdot \left| \left\{ j \in [t] : d_j \approx_2 \tau^{2^\delta} \right\} \right| \cdot \min_{j \in [t] : d_j \approx_2 \tau^{2^\delta}} \left| k_j - \frac{k}{d} \cdot d_j \right| \\
&\geq \Omega\left(\tau^{2^\delta}\right) \cdot \min_{j \in [t] : d_j \approx_2 \tau^{2^\delta}} \left| k_j - \frac{k}{d} . d_j \right| \\
&= \Omega(\tau) \cdot \min_{j \in [t] : d_j \approx_2 \tau^{2^\delta}} \tau^{2^\delta - 1} \cdot \left| k_j - \frac{k}{d} \cdot d_j \right| \\
&\geq \Omega(\tau) \cdot \left( \frac{\eta}{2} - o(1) \right) && \text{(using (7.14))} \\
&\geq \Omega(\tau).
\end{aligned}$$

Therefore, $\mathsf{residue}_k(d_1, \ldots, d_t) \geq \Omega(\tau) \geq \Omega\left(\frac{\tau+1}{2}\right) \geq \Omega(\tau + 1) \geq \Omega\left(d^{2^{1-\Delta}}\right)$. □

### 7.5.3 High residue implies lower bounds

For a 'random' homogeneous degree-$d$ polynomial in $\mathbb{F}[x_1, \ldots, x_n]$, if the shift $\ell$ is not too large, we expect the SP measure to be close to the maximum number of operators used to

construct the shifted partials space, i.e., $M(n,k)\cdot M(n,\ell)$. In the lemma below, we derive a bound for such polynomials. Explicit examples of such polynomials are given in Section 7.5.4.

**Lemma 7.7 (High residue implies lower bounds)** *Let $P = \sum_{i=1}^{s} Q_{i,1}\cdots Q_{i,t_i}$ be a homogeneous $n$-variate polynomial of degree $d$ where $\{Q_{i,j}\}_{i,j}$ are homogeneous and $\mathsf{SP}_{k,\ell}(P) \geq 2^{-O(d)}\cdot M(n,k)\cdot M(n,\ell)$ for some $1 \leq k < \frac{d}{2}, n_0 \leq n$ and $\ell = \left\lfloor \frac{n\cdot d}{n_0} \right\rfloor$ such that $d \leq n_0 \approx 2(d-k)\cdot \left(\frac{n}{k}\right)^{\frac{k}{d-k}}$. If there is a $\gamma > 0$ such that for all $i \in [s]$, $\mathsf{residue}_k(\deg(Q_{i,1}),\ldots,\deg(Q_{i,t_i})) \geq \gamma$, then $s \geq 2^{-O(d)}\left(\frac{n}{d}\right)^{\Omega(\gamma)}$.*

**Proof:** Using Lemma 7.4 (Item 1) and the fact that SP is sub-additive), we get

$$\mathsf{SP}_{k,\ell}(P) \leq \sum_{i=1}^{s} \mathsf{SP}_{k,\ell}(Q_{i,1}\cdots Q_{i,t_i}) \leq s\cdot 2^t\cdot d^2 \cdot \max_{\substack{k_0,\ell_0\geq 0 \\ k_0+\frac{k}{d-k}\cdot\ell_0 \leq k-\gamma}} M(n,k_0)\cdot M(n,\ell+\ell_0),$$

where $t := \max_i t_i$ is at most $d$. On the other hand, by our assumption we have $\mathsf{SP}_{k,\ell}(P) \geq 2^{-O(d)}\cdot M(n,k)\cdot M(n,\ell)$. Putting these two together, we get for some integers $k_0 \in [0..k], \ell_0 \in [0..(d-k)]$ satisfying

$$k_0 + \frac{k}{d-k}\cdot \ell_0 \leq k-\gamma, \tag{7.15}$$

that,

$$\begin{aligned} s &\geq 2^{-O(d)}\cdot 2^{-t}\cdot d^{-2}\cdot \frac{M(n,k)\cdot M(n,\ell)}{M(n,k_0)\cdot M(n,\ell+\ell_0)} \qquad (7.16)\\ &\geq 2^{-O(d)}\cdot \frac{M(n,k)}{M(n,k_0)\cdot (2n/\ell)^{\ell_0}} \end{aligned}$$

(Lemma 7.1 (Item 2) is applicable as $n_0 \geq d$ implies that $n \geq \left\lfloor \frac{nd}{n_0} \right\rfloor = \ell$; also $n_0 \leq n$ implies $\ell = \left\lfloor \frac{nd}{n_0} \right\rfloor \geq d \geq \ell_0$)

$$\geq 2^{-O(d)}\cdot \frac{M(n,k)}{M(n,k_0)\cdot (n_0/\ell_0)^{\ell_0}} \tag{7.17}$$

(because $2n/\ell \leq 4n_0/\ell_0$ as $\ell_0 \leq d$ and absorbing $4^{\ell_0}$ in $2^{-O(d)}$)

$$\geq 2^{-O(d)} \cdot \frac{(n/k)^k}{(6n/k_0)^{k_0} \cdot (n_0/\ell_0)^{\ell_0}}$$

(assuming $k_0, \ell_0 \neq 0$ and using Lemma 7.1 (Item 1) as $n \geq n_0 \geq d \geq \max\{k_0, \ell_0\}$; the analysis is easier if any of $k_0$, $\ell_0$ is 0)

$$\geq 2^{-O(d)} \cdot \frac{(n/k)^k}{(n/k_0)^{k_0} \cdot (n_0/\ell_0)^{\ell_0}} \qquad \text{(since } k_0 = O(d)\text{)}$$

$$\geq 2^{-O(d)} \cdot \frac{(n/k)^k}{(n/k_0)^{k_0} \cdot \left(\frac{2(d-k)}{\ell_0} \cdot (n/k)^{\frac{k}{d-k}}\right)^{\ell_0}} \qquad \text{(substituting } n_0\text{)}$$

$$= 2^{-O(d)} \cdot \frac{(n/k)^k}{(n/k_0)^{k_0} \cdot (\frac{d-k}{\ell_0})^{\ell_0} \cdot (n/k)^{\frac{k\cdot\ell_0}{d-k}}} \qquad \text{(as } \ell_0 = O(d)\text{)}$$

$$= 2^{-O(d)} \cdot \frac{(n/k)^k}{(n/k)^{k_0} \cdot (k/k_0)^{k_0} \cdot (\frac{d-k}{\ell_0})^{\ell_0} \cdot (n/k)^{\frac{k\ell_0}{d-k}}} \qquad \text{(as } n/k_0 = (n/k)\cdot(k/k_0)\text{)}$$

$$= 2^{-O(d)} \cdot \frac{(n/k)^{k-k_0-\frac{k}{d-k}\cdot\ell_0}}{(k/k_0)^{k_0} \cdot (\frac{d-k}{\ell_0})^{\ell_0}}$$

$$\geq 2^{-O(d)} \cdot \frac{(n/d)^\gamma}{(d/k_0)^{k_0} \cdot (\frac{d}{\ell_0})^{\ell_0}} \qquad \text{(using } k < d \text{ and (7.15))}$$

$$= 2^{-O(d)} \cdot \left(\frac{n}{d}\right)^\gamma \cdot \left(\frac{k_0}{d}\right)^{k_0} \cdot \left(\frac{\ell_0}{d}\right)^{\ell_0}$$

$$\geq 2^{-O(d)} \cdot \left(\frac{n}{d}\right)^\gamma \cdot (e^{-1/e})^d \cdot (e^{-1/e})^d \qquad \text{(using } x^x \geq e^{-1/e} \text{ for } x > 0\text{)}$$

$$\geq 2^{-O(d)} \cdot \left(\frac{n}{d}\right)^{\Omega(\gamma)}.$$

□

We state an analogous lemma with APP instead of SP.

**Lemma 7.8 (High residue implies lower bounds, using APP)** *Let $P = \sum_{i=1}^{s} Q_{i,1} \cdots Q_{i,t_i}$ be a homogeneous $n$-variate polynomial of degree $d$ where $\{Q_{i,j}\}_{i,j}$ are homogeneous and $\mathsf{APP}_{k,n_0}(P) \geq 2^{-O(d)} \cdot M(n,k)$ for some $1 \leq k < \frac{d}{2}, n_0 \leq n$ such that $d \leq n_0 \approx 2(d-k).\left(\frac{n}{k}\right)^{\frac{k}{d-k}}$. If there is a $\gamma > 0$ such that for all $i \in [s]$, $\mathsf{residue}_k(\deg(Q_{i,1}), \ldots, \deg(Q_{i,t_i})) \geq \gamma$, then $s \geq 2^{-O(d)} \cdot \left(\frac{n}{d}\right)^{\Omega(\gamma)}$.*

**Proof:** Using Lemma 7.4 (Item 2) and the fact that APP is sub-additive, we get

$$\mathsf{APP}_{k,n_0}(P) \leq \sum_{i=1}^{s} \mathsf{APP}_{k,n_0}(Q_{i,1} \cdots Q_{i,t_i}) \leq s \cdot 2^t \cdot d^2 \cdot \max_{\substack{k_0,\ell_0 \geq 0 \\ k_0 + \frac{k}{d-k} \cdot \ell_0 \leq k-\gamma}} M(n,k_0) \cdot M(n_0,\ell_0).$$

On the other hand, we have $\mathsf{APP}_{k,n_0}(P) \geq 2^{-O(d)} \cdot M(n,k)$. Putting these two together, we get for some integers $k_0, \ell_0 \geq 0$ satisfying

$$k_0 + \frac{k}{d-k} \cdot \ell_0 \leq k - \gamma,$$

that,

$$s \geq 2^{-O(d)} \cdot 2^{-t} \cdot d^{-2} \cdot \frac{M(n,k)}{M(n,k_0) \cdot M(n_0,\ell_0)} \geq 2^{-O(d)} \cdot \left(\frac{n}{d}\right)^{\Omega(\gamma)}.$$

(Using Lemma 7.1, absorbing $6^{\ell_0}$ in $2^{-O(d)}$, and borrowing calculations beginning from (7.17))

□

**Remark 7.1** *In the above lemmas, although our lower bound appears as $2^{-O(d)} \cdot n^{\Omega(\gamma)}$, similar calculations actually give a lower bound of $2^{-O(k)} \cdot n^{\Omega(\gamma)}$ for any choice of $k$ and an appropriate choice of $\ell$ (or $n_0$ in the case of* APP*). We do not differentiate between the two, as for our applications (i.e., low-depth circuits and UPT formulas), the value of $k$ we choose is $\Theta(d)$.*

### 7.5.4 The hard polynomials

We shall prove our lower bound for the word polynomial $P_{\mathbf{w}}$ introduced in [LST21] as well as for the Nisan-Wigderson design polynomial. In order to do this, we show that the SP and APP measures of $P_{\mathbf{w}}$ and the SP measure of $NW$ are large for suitable choices of $k$, $\ell$ and $n_0$.

**Lemma 7.9 ($P_{\mathbf{w}}$ as a hard polynomial)** *For integers $h, d$ such that $h > 100$ and any $k \in \left[\frac{d}{30}, \frac{d}{2}\right]$, there exists an $h$-*unbiased *word $\mathbf{w} \in [-h..h]^d$, integers $n_0 \leq n$, $\ell = \left\lfloor \frac{n \cdot d}{n_0} \right\rfloor$ such that $n_0 \approx 2(d-k) \cdot \left(\frac{n}{k}\right)^{\frac{k}{d-k}}$ and the following bounds hold: $\mathsf{SP}_{k,\ell}(P_{\mathbf{w}}) \geq 2^{-O(d)} \cdot M(n,k) \cdot M(n,\ell)$ and $\mathsf{APP}_{k,n_0}(P_{\mathbf{w}}) \geq 2^{-O(d)} \cdot M(n,k)$. Here $n$ refers to the number of variables in $P_{\mathbf{w}}$, i.e., $n = \sum_{i \in [d]} 2^{|w_i|}$.*

**Proof:** We construct the word $\mathbf{w}$ as follows. Let $h' = \frac{h \cdot k}{d-k} \in \left[\frac{h}{29}, h\right]$. The word $\mathbf{w}$ shall consist of the following elements (the ordering of these elements shall be fixed shortly): $h, \ldots, h$ ($k$ times), $-\lfloor h' \rfloor, \ldots, -\lfloor h' \rfloor$ ($k_1$ times), $-\lceil h' \rceil, \ldots, -\lceil h' \rceil$ ($k_2$ times), where $k_1 :=$

$(d-k)\lceil h' \rceil - kh$ and $k_2 := d - k - k_1$. We note that $k_1, k_2 \in \mathbb{Z}_{\geq 0}$ and $k + k_1 + k_2 = d$. Assuming $\lfloor h' \rfloor = \lceil h' \rceil - 1$ (even if $h' \in \mathbb{Z}$, the calculations are similar), the total sum of the weights is

$$\begin{aligned}\sum_{i \in [d]} w_i &= kh - k_1 \lfloor h' \rfloor - k_2 \lceil h' \rceil = kh - k_1(\lceil h' \rceil - 1) - k_2 \lceil h' \rceil = kh - k_1 \lceil h' \rceil + k_1 - k_2 \lceil h' \rceil \\ &= kh - k_1 \lceil h' \rceil + (d-k) \lceil h' \rceil - kh - k_2 \lceil h' \rceil = 0. \end{aligned} \tag{7.18}$$

Now we fix the ordering of the above weights. For $i = 1$ to $d$ in this order, if the sum $\sum_{j \in [i-1]} w_j$ is non-negative (for example, this happens for $i = 1$), set $w_i$ to be an arbitrary negative weight that is available, otherwise set it to be the positive weight $h$ (if available).

If the above procedure never runs out of positive or negative weights at any step $i \in [d]$, then for all $i \in [d]$, $|w_1 + \cdots + w_i| \leq h$. In other words, $\mathbf{w}$ is *h-unbiased*. Now suppose the procedure runs out of negative weights at an index $i \in [d]$. This means that the sum $\sum_{j \in [i-1]} w_j$ is non-negative but there are no negative weights available among the unused weights. But then, the total sum of the weights would be equal to $\sum_{j \in [i-1]} w_j$ plus the sum of unused weights, which is greater than 0, contradicting (7.18). We get a similar contradiction if there are insufficient positive weights at any point. For the rest of the proof, we fix $\mathbf{w}$ to be the above word. Then,

$$k \cdot 2^h \leq n \leq d \cdot 2^h, \text{ so } 2^h \approx_{30} \binom{n}{k}. \tag{7.19}$$

Denoting the variables of $P_{\mathbf{w}}$ by $\mathbf{x} = \mathbf{y} \sqcup \mathbf{z}$, where $\mathbf{y}$ are the positive variables and $\mathbf{z}$ are the negative variables, we take

$$n_0 := |\mathbf{z}| \approx_2 (d-k) \cdot 2^{\lceil h' \rceil}.$$

Note that $n_0 \approx_2 (d-k) \cdot 2^{\lceil h' \rceil} \approx 2(d-k) \cdot 2^{h'} = 2(d-k) \cdot 2^{\frac{hk}{d-k}} \leq 2k \cdot 2^h = 2(n - n_0)$ where the last inequality follows from the fact that $\frac{d-k}{k} \cdot 2^{\frac{hk}{d-k}}$ is an increasing function of $k$ when $k \in \left[\frac{d}{30}, \frac{d}{2}\right]$ and $h > 100$. That is, $n_0 \leq 2n/3$ and $n_0 \approx 2(d-k) \cdot \left(\frac{n}{k}\right)^{\frac{k}{d-k}}$ by (7.19) and $k \leq \frac{d}{2}$. Also, $n_0 \geq \frac{d-k}{2} \cdot 2^{\lceil h' \rceil} \geq \frac{d-k}{2} \cdot 2^{h'} \geq \frac{d-k}{2} \cdot 2^{\frac{h}{29}} \geq \frac{d-k}{2} \cdot 2^3 \geq 2d$ as $h > 100$ and $k \leq \frac{d}{2}$. Define a map $L : \mathbf{x} \to \langle \mathbf{z} \rangle$ as follows:

$$L(x) = \begin{cases} 0, \text{ if } x \in \mathbf{y}, \\ x, \text{ if } x \in \mathbf{z}. \end{cases}$$

We can lower bound the APP measure by using $L$ and considering only the derivatives

with respect to the set-multilinear monomials over all the positive sets, i.e., $\mathcal{M}_+(\mathbf{w})$. By the definition of the polynomial $P_{\mathbf{w}}$ and because $\sum_{i\in[d]} w_i = 0$, for every $m_+ \in \mathcal{M}_+$, there exists a unique $m_- \in \mathcal{M}_-$ such that $m_+ \cdot m_-$ is a monomial in $P_{\mathbf{w}}$ and vice versa.[1] Hence the set of all derivatives of $P_{\mathbf{w}}$ with respect to monomials in $\mathcal{M}_+$ is exactly $\mathcal{M}_-$, yielding

$$\partial^k (P_{\mathbf{w}}) \supseteq \mathcal{M}_-(\mathbf{w}). \tag{7.20}$$

Using the fact that $\sum_{i\in[d]} w_i = 0$ and (7.19), the size of $\mathcal{M}_-(\mathbf{w})$ is

$$|\mathcal{M}_-(\mathbf{w})| = 2^{\sum_{i\in[d]:w_i<0}|w_i|} = 2^{hk} \geq 2^{-O(k)} \cdot \left(\frac{n}{k}\right)^k \geq 2^{-O(d)} \cdot M(n,k). \tag{7.21}$$

The last bound follows from Lemma 7.1 (Item 1), as $n \geq n_0 \geq d \geq k$. As the substitution $\pi_L$ does not affect negative variables, thus,

$$\mathsf{APP}_{k,n_0}(P_{\mathbf{w}}) \geq \dim\left\langle \pi_L\left(\partial^k (P_{\mathbf{w}})\right)\right\rangle \geq \dim\langle \pi_L(\mathcal{M}_-(\mathbf{w}))\rangle = \dim\langle \mathcal{M}_-(\mathbf{w})\rangle \geq 2^{-O(d)} \cdot M(n,k).$$

We now analyze the shifted partials of the same polynomial with $\ell := \left\lfloor \frac{n\cdot d}{n_0} \right\rfloor$. Recall that $n_0 \leq 2n/3$.

$$\begin{aligned}
\mathsf{SP}_{k,\ell}(P_{\mathbf{w}}) &\geq \dim\left\langle \mathbf{x}^\ell \cdot \partial^k (P_{\mathbf{w}})\right\rangle \\
&\geq \dim\left\langle \mathbf{y}^\ell \cdot \mathcal{M}_-(\mathbf{w})\right\rangle && \text{(as } \mathbf{x} \supseteq \mathbf{y} \text{ and (7.20))} \\
&\geq \left|\mathbf{y}^\ell \cdot \mathcal{M}_-(\mathbf{w})\right| \\
&= \left|\mathbf{y}^\ell\right| \cdot |\mathcal{M}_-(\mathbf{w})| && \text{(since } \mathcal{M}_-(\mathbf{w}) \subseteq \mathbf{z}^{d-k} \text{ and } \mathbf{y}\cap\mathbf{z} = \Phi) \\
&= M(n-n_0,\ell)\cdot 2^{-O(d)} \cdot M(n,k) && \text{(using } |\mathbf{y}| = |\mathbf{x}| - |\mathbf{z}| \text{ and (7.21))} \\
&\geq M(n,\ell)\cdot\left(1-\frac{n_0}{n}\right)^\ell \cdot 2^{-O(d)} \cdot M(n,k) && \text{(using Lemma 7.1 (Item 3))} \\
&\geq M(n,\ell)\cdot\left(1-\frac{n_0}{n}\right)^{\frac{n}{n_0}\cdot d} \cdot 2^{-O(d)} \cdot M(n,k) \\
&\geq 2^{-O(d)} \cdot M(n,k)\cdot M(n,\ell). && \text{(since } (1-x)^{1/x} \geq 1/3\sqrt{3} \text{ for } x := n_0/n \leq 2/3)
\end{aligned}$$

□

The following lemma shows that the SP measure of the Nisan-Wigderson design polynomial is 'large' for $k$ as high as $\Theta(d)$, if $\ell$ is chosen suitably.

[1] Recall the definition of $P_{\mathbf{w}}$ from Section 2.3. Because $|\mathcal{M}_+| = |\mathcal{M}_-|$, the bit representations of $m_+$ and $m_-$ are the same. However, they can have different degrees.

**Lemma 7.10 (*NW* as a hard polynomial)** *For $n, d \in \mathbb{N}$ such that $120 \leq d \leq \frac{1}{150}\left(\frac{\log n}{\log\log n}\right)^2$, let $q$ be the largest prime number between $\left\lfloor \frac{n}{2d} \right\rfloor$ and $\left\lfloor \frac{n}{d} \right\rfloor$. For parameters $k \in \left[\frac{d}{30}, \frac{d}{2} - \frac{\sqrt{d}}{8}\right]$ and $\ell = \left\lfloor \frac{qd^2}{n_0} \right\rfloor$, where $n_0 = 2(d-k)\cdot\left(\frac{qd}{k}\right)^{\frac{k}{d-k}}$, $\mathsf{SP}_{k,\ell}(NW_{q,d,k}) \geq 2^{-O(d)} \cdot M(qd,k) \cdot M(qd,\ell)$.*

**Proof:** We begin by obtaining bounds on the value of $\ell$.

**Claim 7.4** *$n_0 = o(qd)$, $d^2 = o(\ell)$ and $\ell = o(qd)$.*

**Proof:**

$$\begin{aligned}
n_0 &= 2(d-k)\left(\frac{qd}{k}\right)^{\frac{k}{d-k}} \\
&\leq 2(d-k)\left(\frac{qd}{k}\right)^{\frac{d/2-\sqrt{d}/8}{d/2+\sqrt{d}/8}} && \left(\text{because } k = o(qd) \text{ and } k \leq \frac{d}{2} - \frac{\sqrt{d}}{8}\right) \\
&= 2(d-k)\cdot\frac{qd}{k}\cdot\left(\frac{k}{qd}\right)^{\frac{2}{4\sqrt{d}+1}} \\
&\leq 2d\cdot qd\cdot\frac{1}{(qd)^{\frac{1}{2.5\sqrt{d}}}} \\
&\leq 2d\cdot qd\cdot\frac{1}{2^{\frac{\log qd}{2.5\sqrt{d}}}}.
\end{aligned}$$

As $d \leq \frac{1}{150}\left(\frac{\log n}{\log\log n}\right)^2$ and $qd \geq \frac{n}{4}$, $\frac{\log qd}{2.5\sqrt{d}} \geq \frac{12\log\log n}{3} = 4\log\log n$. As $\log d^2 \leq 4\log\log n - \omega(1)$, $\frac{\log qd}{2.5\sqrt{d}} = \log d^2 + \omega(1)$ and $2^{\frac{\log qd}{2.5\sqrt{d}}} = \omega(d^2)$. Thus, $n_0 = o(q) = o(qd)$.[1] Now, $\ell \geq \frac{qd^2}{n_0} - 1 \geq \frac{qd^2}{o(q)} - 1 = \omega(d^2)$. Thus, $d^2 = o(\ell)$. Also,

$$\begin{aligned}
\ell &\leq \frac{qd^2}{n_0} \\
&= \frac{qd^2}{2(d-k)}\left(\frac{k}{qd}\right)^{\frac{k}{d-k}} \\
&\leq \frac{qd^2}{2(d-k)}\left(\frac{k}{qd}\right)^{\frac{1}{29}} && \left(\text{because } k = o(qd) \text{ and } \frac{d}{30} \leq k\right)
\end{aligned}$$

[1] In this proof, we need $n_0 = o(q)$. However, we require $n_0 = o(n)$, in Section 7.5.6 and so we have mentioned $n_0 = o(qd)$ in the statement of the claim.

$$\leq k \cdot (qd)^{\frac{28}{29}} \qquad \left(\text{as } k \leq \frac{d}{2}\right)$$

$$= o(n) \qquad \left(\text{because } k \leq \log^2 n \text{ and } qd \leq n\right).$$

□

Let

$$S = \left\{ \prod_{i \in [k+1 \ldots d]} x_{i,h(i)} : h \in \mathbb{F}_q[z], \deg(h) < k \right\}$$

and

$$T = \{m : \exists \text{ monomials } m_1,\ m_2,\ \deg(m_1) = \ell, m_2 \in S \text{ and } m = m_1 m_2\}.$$

Observe that $T \subseteq \left\langle \mathbf{x}^\ell \boldsymbol{\partial}^k NW_{q,d,k} \right\rangle$ and so, $\mathsf{SP}_{k,\ell}(NW_{q,d,k}) \geq |T|$. We obtain a lower bound on $|T|$. For $h \in \mathbb{F}_q[z]$ such that $\deg(h) < k$, let

$$T_h = \left\{ m_1 \prod_{i \in [k+1 \ldots d]} x_{i,h(i)} : \deg(m_1) = \ell \right\}.$$

Then, $T = \cup_{\substack{h(z) \in \mathbb{F}_q[z]: \\ \deg(h) < k}} T_h$. Thus, from the inclusion-exclusion principle,

$$|T| \geq \sum_{\substack{h \in \mathbb{F}_q[z]: \\ \deg(h) < k}} |T_h| - \sum_{\substack{h_1 \neq h_2 \in \mathbb{F}_q[z]: \\ \deg(h_1), \deg(h_2) < k}} \left|T_{h_1} \cap T_{h_2}\right|. \tag{7.22}$$

**Lower bound on $\sum_h |T_h|$.** Fix an $h \in \mathbb{F}_q[z]$ such that $\deg(h) < k$. Then, since for monomials $m_1 \neq m_2$, $m_1 \cdot \prod_{i \in [k+1 \ldots d]} x_{i,h(i)} \neq m_2 \cdot \prod_{i \in [k+1 \ldots d]} x_{i,h(i)}$, $|T_h| = \binom{qd+\ell-1}{qd-1}$. Hence,

$$\sum_{\substack{h \in \mathbb{F}_q[z]: \\ \deg(h) < k}} |T_h| = |S|^k \cdot \binom{qd+\ell-1}{qd-1} = q^k \cdot \binom{qd+\ell-1}{qd-1}. \tag{7.23}$$

**Upper bound on $\sum_{h_1 \neq h_2} |T_{h_1} \cap T_{h_2}|$.** For $h_1, h_2 \in \mathbb{F}[z]$ such that $\deg(h_1), \deg(h_2) < k$, we say that $|h_1 \cap h_2| = r$ if $|\{h_1(k+1), \ldots, h_1(d)\} \cap \{h_2(k+1), \ldots, h_2(d)\}| = r$. Now

$$\sum_{h_1 \neq h_2} |T_{h_1} \cap T_{h_2}| = \sum_{r=0}^{k-1} \sum_{\substack{h_1 \neq h_2: \\ |h_1 \cap h_2| = r}} |T_{h_1} \cap T_{h_2}|. \tag{7.24}$$

Fix $h_1$ and $h_2$ such that $|h_1 \cap h_2| = r$. Let $m_1 = \prod_{i\in[k+1..d]} x_{i,h_1(i)}$ and $m_2 = \prod_{i\in[k+1..d]} x_{i,h_2(i)}$. A monomial $m \in T_{h_1} \cap T_{h_2}$ if and only if there exist degree $\ell$ monomials $m_1'$ and $m_2'$ such that $m = m_1'm_1 = m_2'm_2$. Thus $\frac{m_2}{\gcd(m_1,m_2)}$ must divide $m_1'$. As $|h_1 \cap h_2| = r$, $\gcd(m_1, m_2)$ has degree $r$, and so $\frac{m_2}{\gcd(m_1,m_2)}$ has degree $d-k-r$. Hence the number of possible monomials $m_1'$, and thus the number of possible monomials $m$ is at most $\binom{qd+\ell-d+k+r-1}{qd-1}$. Now, the number of possible polynomials $h_1$ and $h_2$ such that $|h_1 \cap h_2| = r$ is at most $\binom{d-k}{r}q^{k-r}q^k = q^{2k-r}\binom{d-k}{r}$.[1] Hence,

$$\sum_{\substack{h_1\neq h_2:\\ |h_1\cap h_2|=r}} |T_{h_1} \cap T_{h_2}| \leq q^{2k-r} \cdot \binom{d-k}{r}\binom{qd+\ell-d+k+r-1}{qd-1}. \tag{7.25}$$

**Claim 7.5** *For $r \in [0..k-1]$, let $\chi(r) = q^{2k-r} \cdot \binom{d-k}{r}\binom{qd+\ell-d+k+r-1}{qd-1}$. Then $\chi(0) \geq \chi(r)$ for all $r \in [k-1]$.*

**Proof:** We shall show that for all $r \in [0..k-2]$, $\frac{\chi(r+1)}{\chi(r)} < 1$; this will prove the claim. Fix any $r \in [0..k-2]$.

$$\begin{aligned}
\frac{\chi(r+1)}{\chi(r)} &= \frac{q^{2k-r-1} \cdot \binom{d-k}{r+1}\binom{qd+\ell-d+k+r}{qd-1}}{q^{2k-r} \cdot \binom{d-k}{r}\binom{qd+\ell-d+k+r-1}{qd-1}} \\
&= \frac{1}{q} \cdot \frac{\frac{(d-k)!}{(r+1)!(d-k-r-1)!}}{\frac{(d-k)!}{r!(d-k-r)!}} \cdot \frac{\frac{(qd+\ell-d+k+r)!}{(qd-1)!(\ell-d+k+r+1)!}}{\frac{(qd+\ell-d+k+r-1)!}{(qd-1)!(\ell-d+k+r)!}} \\
&= \frac{1}{q} \cdot \frac{d-k-r}{r+1} \cdot \frac{qd+\ell-d+k+r}{\ell-d+k+r+1} \\
&\leq \frac{d}{q} \cdot \frac{(1+o(1))qd}{(1-o(1))\ell} && \text{(by Claim 7.4)} \\
&= (1+o(1))\frac{d^2}{\ell} \\
&= o(1), && \text{(by Claim 7.4)}
\end{aligned}$$

where the second to last inequality follows from $k, r \geq 0$, $\ell, d, k, r = o(n)$ and $d, k, r = o(\ell)$ (Claim 7.4), and the last equality from the fact that $d^2 = o(\ell)$ (Claim 7.4). □

[1] This is so because $|h_1 \cap h_2| = r$ implies that $h_1 - h_2 = (z-\alpha_1)\cdots(z-\alpha_r)\cdot g(z)$, where $\alpha_1, \ldots, \alpha_r$ are distinct elements in $[k+1..d]$ and $g(z)$ is a polynomial of degree at most $d-k$.

From Equations (7.24), (7.25), and the above claim, we get

$$\sum_{h_1 \neq h_2} |T_{h_1} \cap T_{h_2}| \leq k \cdot q^{2k} \cdot \binom{qd + \ell - d + k - 1}{qd - 1}. \tag{7.26}$$

Thus from Equations (7.22), (7.23), and (7.26),

$$\begin{aligned} |T| &\geq q^k \cdot \binom{qd + \ell - 1}{qd - 1} - k \cdot q^{2k} \cdot \binom{qd + \ell - d + k - 1}{qd - 1} \\ &= q^k \cdot \binom{qd + \ell - 1}{qd - 1} \left(1 - \frac{k \cdot q^{2k} \cdot \binom{qd+\ell-d+k-1}{qd-1}}{q^k \cdot \binom{qd+\ell-1}{qd-1}}\right). \end{aligned} \tag{7.27}$$

**Claim 7.6** $\frac{k \cdot q^{2k} \cdot \binom{qd+\ell-d+k-1}{qd-1}}{q^k \cdot \binom{qd+\ell-1}{qd-1}} \leq \frac{1}{2}$.

**Proof:**

$$\begin{aligned} \frac{k \cdot q^{2k} \cdot \binom{qd+\ell-d+k-1}{qd-1}}{q^k \cdot \binom{qd+\ell-1}{qd-1}} &= k \cdot q^k \cdot \frac{\frac{(qd+\ell-d+k-1)!}{(qd-1)!(\ell-d+k)!}}{\frac{(qd+\ell-1)!}{(qd-1)!\ell!}} \\ &= k \cdot q^k \cdot \frac{\ell \cdot (\ell-1) \cdot (\ell-2) \cdots (\ell-d+k+1)}{(qd+\ell-1) \cdot (qd+\ell-2) \cdot (qd+\ell-3) \cdots (qd+\ell-d+k)} \\ &= k \cdot q^k \cdot \frac{1}{\left(\frac{qd-1}{\ell}+1\right) \cdot \left(\frac{qd-1}{\ell-1}+1\right) \cdot \left(\frac{qd-1}{\ell-2}+1\right) \cdots \left(\frac{qd-1}{\ell-d+k+1}+1\right)} \\ &\leq k \cdot q^k \cdot \frac{1}{\left(\frac{qd-1}{\ell}\right)^{d-k}} \\ &\leq k \cdot q^k \cdot \left(\frac{\ell}{qd}\right)^{d-k} e^{\frac{2(d-k)}{qd}} \qquad \text{(as } 1 - x \geq e^{-2x} \text{ for } x \in [0, 1/2]\text{)} \\ &= (1 + o(1)) \cdot k \cdot q^k \cdot \left(\frac{d}{n_0}\right)^{d-k} \qquad \text{(as } d - k = o(qd) \text{ and } \ell \leq \frac{qd^2}{n_0}\text{)} \\ &= (1 + o(1)) \cdot k \cdot q^k \cdot \left(\frac{d}{2(d-k)}\right)^{d-k} \cdot \left(\frac{k}{qd}\right)^k \\ &\leq (1 + o(1)) \cdot k \cdot \left(\frac{1}{2}\right)^k \qquad \text{(as } k \leq \frac{d}{2}\text{)} \\ &\leq \frac{1}{2}, \end{aligned}$$

when $d \geq 120$. □

Thus, from Equation (7.27) and $k = \Theta(d)$, we get

$$|T| \geq \frac{1}{2} \cdot q^k \cdot \binom{qd+\ell-1}{qd-1} \geq 2^{-O(d)} \cdot \left(\frac{qd}{k}\right)^k \cdot \binom{qd+\ell-1}{qd-1} \geq 2^{-O(d)} \cdot \binom{qd+k-1}{qd-1} \cdot \binom{qd+\ell-1}{qd-1},$$

where the last inequality follows from Lemma 7.1. Recall that $\mathsf{SP}_{k,\ell}(NW_{q,d,k}) \geq |T|$. Hence, $\mathsf{SP}_{k,\ell}(NW_{q,d,k}) \geq 2^{-O(d)} \cdot M(qd,k) \cdot M(qd,\ell)$. □

**Remark 7.2** *An advantage of directly analysing the complexity measures for homogeneous formulas instead of for set-multilinear formulas is that our hard polynomial need not be set multilinear. In Section 7.5.6, we describe an explicit non set-multilinear polynomial $P$ (in* VNP*) with a large* APP *measure; the construction is similar to a polynomial in [GKS20]. The proof that* APP *of $P$ is large is considerably simpler than the proofs of the above lemmas.*

### 7.5.5 Putting everything together: the low-depth lower bound

**Theorem 7.1 (Low-depth homogeneous formula lower bound for $IMM$)** *For any $d, n, \Delta$ such that $n = \omega(d)$, any homogeneous formula of product-depth at most $\Delta$ computing $IMM_{n,d}$ over any field $\mathbb{F}$ has size at least $2^{-O(d)} \cdot n^{\Omega\left(d^{2^{1-\Delta}}\right)}$. In particular, when $d = O(\log n)$, we get a lower bound of $n^{\Omega\left(d^{2^{1-\Delta}}\right)}$.*

**Proof:** We can assume that $d^{2^{1-\Delta}} = \omega(1)$ and $h := \lfloor \log n \rfloor > 100$, as otherwise the lower bound is trivial. Suppose $IMM_{n,d}$ has a homogeneous formula $C$ of product-depth at most $\Delta$. Consider the polynomial $P_{\mathbf{w}}$, given by Lemma 7.9, by setting $k := \left\lfloor \frac{\alpha \cdot d}{1+\alpha} \right\rfloor$, where $\alpha := \sum_{\nu=0}^{\Delta-1} \frac{(-1)^\nu}{\tau^{2^\nu-1}}$ and $\tau := \left\lfloor d^{2^{1-\Delta}} \right\rfloor$; these parameters are the same as those in Lemma 7.6. It is easy to show that $k \in \left[\frac{d}{4}, \frac{d}{2}\right]$. As $\mathbf{w}$ is *h-unbiased*, by Lemma 2.1 there exists a homogeneous formula $C'$ of product-depth at most $\Delta$ computing $P_{\mathbf{w}}$ such that $\mathrm{size}(C') \leq \mathrm{size}(C)$. Hence, by Lemma 7.6, there exist homogeneous polynomials $\{Q_{i,j}\}_{i,j}$ such that $P_{\mathbf{w}} = \sum_{i \in [s]} Q_{i,1} \cdots Q_{i,t_i}$, $s \leq \mathrm{size}(C')$ and $\mathsf{residue}_k\left(\deg(Q_{i,1}), \ldots, \deg(Q_{i,t_i})\right) \geq \Omega\left(d^{2^{1-\Delta}}\right)$ for $i \in [s]$. Denoting the number of variables in $P_{\mathbf{w}}$ by $\widetilde{n}$, Lemma 7.9 guarantees that $n_0 \leq 2(d-k) \cdot \left(\frac{\widetilde{n}}{k}\right)^{\frac{k}{d-k}}$, $\ell = \left\lfloor \frac{\widetilde{n} \cdot d}{n_0} \right\rfloor$ and $\mathsf{SP}_{k,\ell}(P_{\mathbf{w}}) \geq 2^{-O(d)} \cdot M(\widetilde{n},k) \cdot M(\widetilde{n},\ell)$. Therefore, we can apply Lemma 7.7 to the same polynomial $P_{\mathbf{w}}$ which gives that $s \geq 2^{-O(d)} \cdot \left(\frac{\widetilde{n}}{d}\right)^{\Omega\left(d^{2^{1-\Delta}}\right)}$. Hence, $\mathrm{size}(C) \geq \mathrm{size}(C') \geq s \geq 2^{-O(d)} \cdot n^{\Omega\left(d^{2^{1-\Delta}}\right)}$, since $\widetilde{n} \geq 2^h \geq n/2 = \omega(d)$. □

**Theorem 7.2 (Low-depth homogeneous formula lower bound for $NW$)** *Let $n, d, \Delta$ be positive integers. If $\Delta = 1$, let $d = n^{1-\epsilon}$ for any constant $\epsilon > 0$ and $k = \left\lfloor \frac{d-1}{2} \right\rfloor$. Otherwise, let $d \leq \frac{1}{150}\left(\frac{\log n}{\log\log n}\right)^2$, let $\tau = \left\lfloor d^{2^{1-\Delta}} \right\rfloor$, $\alpha = \sum_{\nu=0}^{\Delta-1} \frac{(-1)^\nu}{\tau^{2^\nu - 1}}$, and $k = \left\lfloor \frac{\alpha \cdot d}{1+\alpha} \right\rfloor$. In both cases, let $q$ be the largest prime between $\left\lfloor \frac{n}{2d} \right\rfloor$ and $\left\lfloor \frac{n}{d} \right\rfloor$. Then, any homogeneous formula of product-depth at most $\Delta$ computing $NW_{q,d,k}$ over any field $\mathbb{F}$ has size at least $2^{-O(d)} \cdot n^{\Omega\left(d^{2^{1-\Delta}}\right)}$. In particular, when $d = O(\log n)$, we get a lower bound of $n^{\Omega\left(d^{2^{1-\Delta}}\right)}$.*

**Proof:** We analyse the cases $\Delta = 1$ and $\Delta \geq 2$ separately.

$\Delta = 1$. Let $C$ be a homogeneous formula of product-depth 1 computing $NW_{q,d,k}$. Then, $C = \sum_{i\in[s]} \prod_{j\in[d]} Q_{i,j}$, where $Q_{i,j}$ are linear forms. Observe that for any $i \in [k]$, $\partial^k\left(\prod_{j\in[d]} Q_{i,j}\right) \subseteq \left\langle \prod_{j\in[d]\setminus S} C_{i,j} : |S| = k \right\rangle$. Thus, $\dim\left\langle \partial^k\left(\prod_{j\in[d]} Q_{i,j}\right)\right\rangle \leq \binom{d}{k}$. As $\partial^k C \subseteq \sum_{i\in[s]} \left\langle \partial^k\left(\prod_{j\in[d]} Q_{i,j}\right)\right\rangle$, $\dim\left\langle \partial^k C\right\rangle \leq s \cdot \binom{d}{k}$.

On the other hand, $\dim\left\langle \partial^k(NW_{q,d,k})\right\rangle = \binom{d}{k} \cdot q^k$: For every $S \subseteq [d], |S| = k$,

$$T_S := \left\{ \prod_{i\in[d]\setminus S} x_{i,h(i)} : h \in \mathbb{F}[z], \deg(h) < k \right\} \subseteq \partial^k(NW_{q,d,k}).$$

Now, for $h_1 \neq h_2 \in \mathbb{F}[z], \deg(h_1), \deg(h_2) < k$, there exists an $i \in [d] \setminus S$ such that $h_1(i) \neq h_2(i)$ because $|[d] \setminus S| = d - k \geq k + 1$. Thus, $\prod_{i\in[d]\setminus S} x_{i,h_1(i)} \neq \prod_{i\in[d]\setminus S} x_{i,h_2(i)}$, and $|T_S| = q^k$. Also, for $S \neq S' \subseteq [d]$, $|S| = |S'| = k$, $T_S$ and $T_{S'}$ are disjoint. Hence, $\dim\left\langle \partial^k(NW_{q,d,k})\right\rangle \geq \binom{d}{k} \cdot q^k$.[1] Thus, $s \geq q^k = \left(\frac{n}{d}\right)^{O(d)}$ as $k = \Theta(d)$ and $qd = \Theta(n)$. Because $d \leq n^{1-\epsilon}$, this means that $s \geq n^{O(d)}$.

$\Delta \geq 2$. We can assume that $d^{2^{1-\Delta}} = \omega(1)$, as otherwise the given bound is trivial. Let $C$ be a homogeneous formula of product-depth at most $\Delta$ computing $NW_{q,d,k}$; $C$ is a formula in $qd$ variables. By Lemma 7.6, there exist homogeneous polynomials $\{Q_{i,j}\}_{i,j}$ such that $NW_{q,d,k} = \sum_{i\in[s]} Q_{i,1} \cdots Q_{i,t_i}$, $s \leq \text{size}(C)$, and $\text{residue}_k\left(\deg(Q_{i,1}), \ldots, \deg(Q_{i,t_i})\right) \geq \Omega\left(d^{2^{1-\Delta}}\right)$ for $i \in [s]$. From the proof of Lemma 7.6, $k \in \left[\frac{d}{4}, \frac{d}{2}\right]$. In fact, as $\frac{k}{d-k} \leq \alpha \leq 1 - \frac{1}{2\tau} \leq 1 - \frac{1}{2\sqrt{d}}$, $k \leq \frac{d}{2} - \frac{\sqrt{d}}{8}$. Thus, Lemma 7.10 guarantees that for $n_0 = 2(d-k) \cdot \left(\frac{qd}{k}\right)^{\frac{k}{d-k}}$

[1] In fact, it can be shown that this is an equality.

and $\ell = \left\lfloor \frac{qd^2}{n_0} \right\rfloor$, $\mathsf{SP}_{k,\ell}\left(NW_{q,k,d}\right) \geq 2^{-O(d)} \cdot M(qd,k) \cdot M(qd,\ell)$. Also, it follows from the proof of Lemma 7.10 (see Claim 7.4) that for $n_0 \leq qd$. So, applying Lemma 7.7 to $NW_{k,d,q}$ we get that $s \geq 2^{-O(d)} \cdot \left(\frac{qd}{d}\right)^{\Omega\left(d^{2^{1-\Delta}}\right)} = 2^{-O(d)} \cdot n^{\Omega\left(d^{2^{1-\Delta}}\right)}$ as $qd \geq \frac{n}{4}$ and $d = o(n)$. Hence, $\text{size}(C) \geq 2^{-O(d)} \cdot n^{\Omega\left(d^{2^{1-\Delta}}\right)}$. □

**Remark 7.3** *Notice that in the above theorem, as $k$ depends on the product-depth $\Delta$, the polynomial $NW_{q,d,k}$ may be different for different values of $\Delta$. However, much like in [KSS14], there is a way to 'stitch' all the different NW polynomials for different values of $\Delta$ into a single polynomial $P$ such that any homogeneous formula of product-depth $\Delta$ computing $P$ has size at least $2^{-O(d)} n^{\Omega\left(d^{2^{1-\Delta}}\right)}$.*

In [LST21], the authors showed how to convert a circuit of product-depth $\Delta$ computing a homogeneous polynomial to a homogeneous formula of product-depth $2\Delta$ without much increase in the size. Combining Lemma 11 from [LST21] with Theorems 7.1 and 7.2, we get:

**Corollary 7.1 (Low-depth circuit lower bound for $IMM$)** *For any positive integers $d, n, \Delta$ such that $n = \omega(d)$, any circuit of product-depth at most $\Delta$ computing $IMM_{n,d}$ over any field $\mathbb{F}$ with characteristic 0 or more than $d$ has size at least $2^{-O(d)} \cdot n^{\Omega\left(\frac{d^{2^{1-2\Delta}}}{\Delta}\right)}$.*

*In particular, when $d = O(\log n)$, we get a lower bound of $n^{\Omega\left(\frac{d^{2^{1-2\Delta}}}{\Delta}\right)}$.*

**Corollary 7.2 (Low-depth circuit lower bound for $NW$)** *Let $n, d, \Delta$ be positive integers. If $\Delta = 1$, let $d = n^{1-\epsilon}$ for any constant $\epsilon > 0$ and $k = \left\lfloor \frac{d-1}{2} \right\rfloor$. Otherwise, let $d \leq \frac{1}{150}\left(\frac{\log n}{\log\log n}\right)^2$, let $\tau = \left\lfloor d^{2^{1-\Delta}} \right\rfloor$, $\alpha = \sum_{\nu=0}^{\Delta-1} \frac{(-1)^\nu}{\tau^{2^\nu - 1}}$, and $k = \left\lfloor \frac{\alpha \cdot d}{1+\alpha} \right\rfloor$. In both cases, let $q$ be the largest prime number between $\left\lfloor \frac{n}{2d} \right\rfloor$ and $\left\lfloor \frac{n}{d} \right\rfloor$. Then, any circuit of product-depth at most $\Delta$ computing $NW_{q,d,k}$ over any field $\mathbb{F}$ of characteristic 0 or more than $d$ has size at least $2^{-O(d)} \cdot n^{\Omega\left(\frac{d^{2^{1-2\Delta}}}{\Delta}\right)}$.*

*In particular, when $d = O(\log n)$, we get a lower bound of $n^{\Omega\left(\frac{d^{2^{1-2\Delta}}}{\Delta}\right)}$.*

We note that our lower bounds quantitatively improve on the original homogeneous formula lower bound of [LST21] in terms of the dependence on the degree. While [LST21] gives a lower bound of $d^{O(-d)} \cdot n^{\Omega\left(d^{1/2^{\Delta}-1}\right)}$ (as the conversion from homogeneous to set-multilinear formulas increases the size by a factor of $d^{O(d)}$), our lower bound is $2^{-O(d)} \cdot n^{\Omega\left(d^{2^{1-\Delta}}\right)}$. Thus, we get slight improvement both in the multiplicative factor (from $d^{O(d)}$ to $2^{O(d)}$) and in the exponent of $n$ (from $d^{\frac{1}{2^\Delta - 1}}$ to $d^{\frac{1}{2^{(\Delta-1)}}}$). We point out what these improvements mean for

smaller depths: For $\Delta = 2$, our lower bound for homogeneous formulas computing $IMM$ is superpolynomial as long as $d \leq \epsilon \cdot \log^2 n$ for a small enough positive constant $\epsilon$, whereas the lower bound in [LST21] does not work beyond $d = O\left(\left(\frac{\log n}{\log\log n}\right)^2\right)$. In particular, we obtain a lower bound of $n^{\Omega(\log n)}$ for the size of homogeneous depth-5 formulas computing $IMM_{n,d}$ when $d = \Theta(\log^2 n)$.

Finally, for $\Delta = 3$ and $d \leq \epsilon \cdot \log^{4/3} n$, we get a lower bound of $n^{\Omega\left(d^{1/4}\right)}$, as compared to $n^{\Omega\left(d^{1/7}\right)}$ from [LST21].

### 7.5.6 A non-set-multilinear hard polynomial

For any $n, d, \Delta \in \mathbb{N}$ such that $120 \leq d \leq \frac{1}{150}\left(\frac{\log n}{\log\log n}\right)^2$, define $k = \left\lfloor \frac{\alpha \cdot d}{1+\alpha} \right\rfloor$, where $\alpha := \sum_{\nu=0}^{\Delta-1} \frac{(-1)^\nu}{\tau^{2^\nu - 1}}$ and $\tau := \left\lfloor d^{2^{1-\Delta}} \right\rfloor$. Then, let $n_0 = \left\lfloor 2(d-k) \cdot \left(\frac{n}{k}\right)^{\frac{k}{d-k}} \right\rfloor$ ($n_0 \leq n$, see Section 7.5.6), $n_1 = n - n_0$, $\mathbf{y} = \{x_1, \ldots, x_{n_1}\}$ and $\mathbf{z} = \{x_{n_1+1}, \ldots, x_n\}$. Let $\mathcal{M}_y$ be the set of all (monic) monomials of degree $k$ in $\mathbf{y}$ variables and $\mathcal{M}_z$ be the set of all (monic) monomials in $\mathbf{z}$ variables of degree $d-k$; it can be verified that $|\mathcal{M}_y| \leq |\mathcal{M}_z|$. Fix any one-to-one function $\sigma : \mathcal{M}_y \to \mathcal{M}_z$. Then, it is easy to see that for $P_\sigma := \sum_{m \in \mathcal{M}_y} m \cdot \sigma(m)$, $\mathsf{APP}_{k,n_0}(P) = M(n_1, k)$. While $P_\sigma$ defined above might have a non-trivial set-multilinear component, it can be modified to ensure that there are no multilinear monomials in it. Notice that to prove a lower bound for such a polynomial, we must analyse the measure of a homogeneous formula computing it directly; we can not hope to get a lower bound by going via set-multilinearity as is done in [LST21].

**Lemma 7.11 (Non-set-multilinear hard polynomial)** $\mathsf{APP}(P_\sigma) \geq 2^{-O(k)} M(n,k)$.

**Proof:** Using an analysis similar to the one in the proof of Claim 7.4, it can be shown that $n_0 = o(n)$. Also, from the proof of Lemma 7.6, $k \in \left[\frac{d}{4}, \frac{d}{2}\right]$. Let us assume that $\Delta \geq 2$; the case of $\Delta = 1$ is simple and can be handled separately. Then, as $\frac{k}{d-k} \leq \alpha \leq 1 - \frac{1}{2\tau} \leq 1 - \frac{1}{2\sqrt{d}}$, $k \leq d - \frac{\sqrt{d}}{2} - k + \frac{k}{2\sqrt{d}}$, and hence $k \leq \frac{d}{2} - \frac{\sqrt{d}}{4} + \frac{k}{4\sqrt{d}} \leq \frac{d}{2} - \frac{\sqrt{d}}{4} + \frac{d}{8\sqrt{d}} = \frac{d}{2} - \frac{\sqrt{d}}{8}$.

**Claim 7.7** $|\mathcal{M}_y| \leq |\mathcal{M}_z|$.

**Proof:** $|\mathcal{M}_y| = \binom{n_1+k-1}{n_1-1}$ and $\mathcal{M}_z = \binom{n_0+d-k-1}{n_0-1}$. Thus,

$$\begin{aligned}
\frac{|\mathcal{M}_z|}{|\mathcal{M}_y|} &= \frac{(n_0+d-k-1)(n_0+d-k-2)\cdots n_0}{(n_1+k-1)(n_1+k-2)\cdots n_1} \cdot \frac{k!}{(d-k)!} \\
&\geq \frac{k!}{(d-k)!} \cdot \frac{n_0^{d-k}}{n_1^k} \cdot \frac{1}{\left(1+\frac{k-1}{n_1}\right)^k} \\
&\geq (1-o(1)) \cdot \frac{k!}{(d-k)!} \cdot ((2-o(1))(d-k))^{d-k} \left(\frac{n}{k}\right)^k \cdot \frac{1}{n^k} \\
&\qquad \text{(replacing } n_0 \text{ by its value and as } n_0 = o(n), n_1 = \Theta(n) = \omega(k^2)\text{)} \\
&\geq (1.9)^{d-k} \frac{(1-o(1))\sqrt{2\pi k}\left(\frac{k}{e}\right)^k}{(1+o(1))\sqrt{2\pi(d-k)}\left(\frac{d-k}{e}\right)^{d-k}} \cdot \frac{(d-k)^{d-k}}{k^k} \\
&\qquad \text{(using Sterling's approximation)} \\
&\geq (1.8)^{d-k} \cdot e^{d-2k} \cdot \sqrt{\frac{k}{d-k}} \\
&\geq 1,
\end{aligned}$$

for $d \geq 120$. □

**Claim 7.8** $M(n_1,k) \geq 2^{-O(k)} M(n,k)$.

**Proof:** As $n_1 = \Theta(n) \geq k$, from Lemma 7.1, we get

$$M(n_1,k) \geq \left(\frac{n_1}{k}\right)^k = \left(\frac{n}{k}\right)^k \left(1-\frac{n_0}{n}\right)^k \geq \left(\frac{n}{k}\right)^k \cdot e^{-\frac{2kn_0}{n}} \geq 2^{-O(k)} \cdot \left(\frac{6n}{k}\right)^k \geq 2^{-O(k)} \cdot M(n,k),$$

where the third to last inequality follows from $n_0 = o(n)$ and the last inequality follows from $n \geq k$ and Lemma 7.1. □

□

# Chapter 8

# Border of sums of constantly many read-once arithmetic formulas

This chapter studies the border of a sum of read-once arithmetic formulas. The contents of this chapter are part of a joint work with Pranjal Dutta.

## 8.1 Introduction

Recall that for an $f \in \mathbb{F}[\mathbf{x}]$ and a fresh variable $\epsilon \notin \mathbf{x}$, we say that $f$ is approximated by a $g \in \mathbb{F}[\epsilon, \mathbf{x}]$ if there exists an $h \in \mathbb{F}[\epsilon, \mathbf{x}]$ such that $f + \epsilon \cdot h = g$. We abbreviate $f + \epsilon \cdot h = g$ as $f + O(\epsilon) = g$. The border of a circuit $\mathtt{C}$ over $\mathbb{F}(\epsilon)$ is the set of all polynomials approximated by $\mathtt{C}$; the border of a circuit class $\mathcal{C}$ is just the union of the border of all circuits in it (See Section 2.10 for more details). In this chapter, we prove some results about the border of sums of ROFs. We prove the following three theorems in this chapter.

**Theorem 1.10 (Sum of ROFs not closed under border)** *For any $n \in \mathbb{N}$, $n \geq 10$ and $2 \leq k \leq \frac{n}{5}$, $\sum_k \mathrm{ROF}(n) \subsetneq \overline{\sum_k \mathrm{ROF}(n)}$ over any field.*

Recall that $\mathrm{ROF}_0(n)$ is the class of additive constant free ROFs in $n$-variables. (see Definition 2.25

**Theorem 1.11 (De-bordering the border of sum of 2 ROFs)** *Over fields of characteristic other than 2 and for any $n \in \mathbb{N}$, $\overline{\sum_2 \mathrm{ROF}_0(n)} \subseteq \sum_{O(n)} \mathrm{ROF}_0(n)$.*

In the following theorem, by a depth $\Delta$ ROF we mean a depth $\Delta$ ROF with a $+$ gate at the top.

**Theorem 1.12 (Hititng set for the border of sum of 2 homogeneous depth-5 ROFs)** *If $\mathbb{F}$ is a field of size* $\mathrm{poly}(n)$ *and characteristic other than 2, then there is an $n^{O(\log n)}$ time computable hitting set for the border of sums of two homogeneous depth-5 ROFs computing n-variate polynomials over $\mathbb{F}$.*

Theorems 1.10 and 1.11 are proved in Sections 8.2 and 8.3, respectively. The proof of Theorem 1.12 is given in Sections 8.4 and 8.5; in the former we prove a special case of the theorem for the border of sum of two depth-4 ROFs and in the latter we extend this proof to the depth-5 case. Throughout this section, we shall identify a circuit with a polynomial computed by it.

## 8.2 Lower bounds for sums of ROFs against their border

The proof of Theorem 1.10 follows from the two lemmas proved below.

**Lemma 8.1** *For all $n \in \mathbb{N}$ and $1 \leq d \leq n$, $\mathsf{ESym}_{n,d} \in \overline{\sum_{n-d+1} \mathrm{ROF}(n)}$*

**Proof:** We shall show that for all $1 \leq d \leq n$, there exist $f_i(\epsilon), g_i(\epsilon)$ with $\mathsf{val}(f_i) = \mathsf{val}(g_i) = 0$ for all $i \in [n-d]$ such that

$$\mathsf{ESym}_{n,d} + O(\epsilon) = \epsilon^d \prod_{j\in[n]} \left(\frac{x_j}{\epsilon} + 1\right) + \sum_{i=1}^{n-d} \frac{1}{\epsilon^{2^i-1}} \cdot \frac{f_i(\epsilon)}{g_i(\epsilon)} \cdot \epsilon^{(d+i)2^i} \prod_{j\in[n]} \left(\frac{x_j}{\epsilon^{2^i}} + 1\right). \tag{8.1}$$

Since the right hand side above equation has $n-d+1$ summands and each is an ROF over $\mathbb{F}(\epsilon)$, proving the above equation will establish the lemma.

Observe that for all $d \in [n]$,

$$\epsilon^d \prod_{j\in[n]} \left(\frac{x_j}{\epsilon} + 1\right) = \sum_{i=0}^{d-1} \epsilon^{d-i} \cdot \mathsf{ESym}_{n,i} + \mathsf{ESym}_{n,d} + \sum_{i=1}^{n-d} \frac{1}{\epsilon^i} \cdot \mathsf{ESym}_{n,d+i}$$

Thus,

$$\mathsf{ESym}_{n,d} + O(\epsilon) = \epsilon^d \prod_{j\in[n]} \left(\frac{x_j}{\epsilon} + 1\right) - \sum_{i=1}^{n-d} \frac{1}{\epsilon^i} \cdot \mathsf{ESym}_{n,d+i}. \tag{8.2}$$

Assume by the way of induction that for some $0 \leq m < n-d$, there exist $f_i'(\epsilon)$, $g_i'(\epsilon)$ for all

$m+1 \le i \le n-d$ with $\mathsf{val}(f_i') = \mathsf{val}(g_i') = 0$ such that

$$\begin{aligned}\mathsf{ESym}_{n,d} + O(\epsilon) = \epsilon^d \prod_{j\in[n]} \left(\frac{x_j}{\epsilon} + 1\right) \\ + \sum_{i=1}^{m} \frac{1}{\epsilon^{2^i-1}} \cdot \frac{f_i(\epsilon)}{g_i(\epsilon)} \cdot \epsilon^{(d+i)2^i} \prod_{j\in[n]} \left(\frac{x_j}{\epsilon^{2^i}} + 1\right) \\ - \sum_{i=m+1}^{n-d} \frac{1}{\epsilon^{(i-m+1)2^m-1}} \cdot \frac{f_i'(\epsilon)}{g_i'(\epsilon)} \cdot \mathsf{ESym}_{n,d+i}. \end{aligned} \tag{8.3}$$

Note that the base case of $m = 0$ is just Equation (8.2) (with $f_i'(\epsilon) = g_i'(\epsilon) = 1$). As

$$\begin{aligned}\left(\epsilon^{2^{m+1}}\right)^{d+m+1} \prod_{j\in[n]} \left(\frac{x_j}{\epsilon^{2^{m+1}}} + 1\right) = \sum_{i=0}^{d+m} \left(\epsilon^{2^{m+1}}\right)^{d+m+1-i} \cdot \mathsf{ESym}_{n,i} \\ + \mathsf{ESym}_{n,d+m+1} \\ + \sum_{i=m+2}^{n-d} \frac{1}{\epsilon^{(i-m-1)2^{m+1}}} \cdot \mathsf{ESym}_{n,d+i}, \end{aligned}$$

$$\begin{aligned}&\frac{1}{\epsilon^{(m+1-m+1)2^m-1}} \cdot \frac{f_{m+1}'(\epsilon)}{g_{m+1}'(\epsilon)} \cdot \mathsf{ESym}_{n,d+m+1} \\ &\quad = \frac{1}{\epsilon^{2^{m+1}-1}} \cdot \frac{f_{m+1}'(\epsilon)}{g_{m+1}'(\epsilon)} \cdot \mathsf{ESym}_{n,d+m+1} \\ &\quad = \frac{1}{\epsilon^{2^{m+1}-1}} \cdot \frac{f_{m+1}'(\epsilon)}{g_{m+1}'(\epsilon)} \cdot \left(\epsilon^{2^{m+1}}\right)^{d+m+1} \cdot \prod_{j\in[n]} \left(\frac{x_j}{\epsilon^{2^{m+1}}} + 1\right) \\ &\qquad - \sum_{i=0}^{d+m} \frac{1}{\epsilon^{2^{m+1}-1}} \cdot \frac{f_{m+1}'(\epsilon)}{g_{m+1}'(\epsilon)} \cdot \left(\epsilon^{2^{m+1}}\right)^{d+m+1-i} \cdot \mathsf{ESym}_{n,i} \\ &\qquad - \sum_{i=m+2}^{n-d} \frac{1}{\epsilon^{2^{m+1}-1}} \cdot \frac{f_{m+1}'(\epsilon)}{g_{m+1}'(\epsilon)} \cdot \frac{1}{\epsilon^{(i-m-1)2^{m+1}}} \cdot \mathsf{ESym}_{n,d+i} \\ &\quad = \frac{1}{\epsilon^{2^{m+1}-1}} \cdot \frac{f_{m+1}'(\epsilon)}{g_{m+1}'(\epsilon)} \cdot \epsilon^{(d+m+1)2^{m+1}} \cdot \prod_{j\in[n]} \left(\frac{x_j}{\epsilon^{2^{m+1}}} + 1\right) \\ &\qquad - \sum_{i=m+2}^{n-d} \frac{1}{\epsilon^{(i-m)2^{m+1}-1}} \cdot \frac{f_{m+1}'(\epsilon)}{g_{m+1}'(\epsilon)} \cdot \mathsf{ESym}_{n,d+i} \\ &\qquad + O(\epsilon). \end{aligned}$$

Putting this in (8.3), we get

$$\begin{aligned}\mathsf{ESym}_{n,d} + O(\epsilon) = \epsilon^d \prod_{j\in[n]} \left(\frac{x_j}{\epsilon} + 1\right) \\ + \sum_{i=1}^{m} \frac{1}{\epsilon^{2^i-1}} \cdot \frac{f_i(\epsilon)}{g_i(\epsilon)} \cdot \epsilon^{(d+i)2^i} \prod_{j\in[n]} \left(\frac{x_j}{\epsilon^{2^i}} + 1\right) \\ - \frac{1}{\epsilon^{2^{m+1}-1}} \cdot \frac{f'_{m+1}(\epsilon)}{g'_{m+1}(\epsilon)} \cdot \epsilon^{(d+m+1)2^{m+1}} \cdot \prod_{j\in[n]} \left(\frac{x_j}{\epsilon^{2^{m+1}}} + 1\right) \\ + \sum_{i=m+2}^{n-d} \frac{1}{\epsilon^{(i-m)2^{m+1}-1}} \cdot \frac{f'_{m+1}(\epsilon)}{g'_{m+1}(\epsilon)} \cdot \mathsf{ESym}_{n,d+i} \\ - \sum_{i=m+2}^{n-d} \frac{1}{\epsilon^{(i-m+1)2^m-1}} \cdot \frac{f'_i(\epsilon)}{g'_i(\epsilon)} \cdot \mathsf{ESym}_{n,d+i}.\end{aligned}$$

Defining $f_{m+1} = -f'_{m+1}$ and $g_{m+1} = -g'_{m+1}$ we get

$$\begin{aligned}\mathsf{ESym}_{n,d} + O(\epsilon) = \epsilon^d \prod_{j\in[n]} \left(\frac{x_j}{\epsilon} + 1\right) \\ + \sum_{i=1}^{m+1} \frac{1}{\epsilon^{2^i-1}} \cdot \frac{f_i(\epsilon)}{g_i(\epsilon)} \cdot \epsilon^{(d+i)2^i} \prod_{j\in[n]} \left(\frac{x_j}{\epsilon^{2^i}} + 1\right) \\ - \sum_{i=m+2}^{n-d} \left(\frac{1}{\epsilon^{(i-m+1)2^m-1}} \cdot \frac{f'_i(\epsilon)}{g'_i(\epsilon)} - \frac{1}{\epsilon^{(i-m)2^{m+1}-1}} \cdot \frac{f'_{m+1}(\epsilon)}{g'_{m+1}(\epsilon)}\right) \cdot \mathsf{ESym}_{n,d+i}.\end{aligned}$$

Now $(i-m)2^{m+1} - 1 > (i-m+1)2^m - 1$ for all $m+2 \le i \le n-d$. Thus,

$$\begin{aligned}&\frac{1}{\epsilon^{(i-m+1)2^m-1}} \cdot \frac{f'_i(\epsilon)}{g'_i(\epsilon)} - \frac{1}{\epsilon^{(i-m)2^{m+1}-1}} \cdot \frac{f'_{m+1}(\epsilon)}{g'_{m+1}(\epsilon)} \\ &= \frac{1}{\epsilon^{(i-m)2^{m+1}-1}} \cdot \frac{\epsilon^{(i-m-1)2^m} \cdot f'_i(\epsilon) \cdot g'_{m+1}(\epsilon) \;-\; f'_{m+1}(\epsilon) \cdot g'_i(\epsilon)}{g'_i(\epsilon) \cdot g'_{m+1}(\epsilon)}.\end{aligned}$$

For all $m+2 \le i \le n-d$, define $f''_i(\epsilon) = \epsilon^{(i-m-1)2^m} \cdot f'_i(\epsilon) \cdot g'_{m+1}(\epsilon) \;-\; f'_{m+1}(\epsilon) \cdot g'_i(\epsilon)$ and $g''_i(\epsilon) = g'_i(\epsilon) \cdot g'_{m+1}(\epsilon)$. Since $\mathsf{val}(f'_{m+1}) = \mathsf{val}(g'_i) = 0$, $\mathsf{val}(f''_i) = 0$. Similarly, $\mathsf{val}(g'_{m+1}) = \mathsf{val}(g'_i) = 0$ imply $\mathsf{val}(g''_i) = 0$. Hence,

$$\mathsf{ESym}_{n,d} + O(\epsilon) = \epsilon^d \prod_{j\in[n]} \left(\frac{x_j}{\epsilon} + 1\right)$$

$$+\sum_{i=1}^{m+1}\frac{1}{\epsilon^{2^i-1}}\cdot\frac{f_i(\epsilon)}{g_i(\epsilon)}\cdot\epsilon^{(d+i)2^i}\prod_{j\in[n]}\left(\frac{x_j}{\epsilon^{2^i}}+1\right)$$
$$-\sum_{i=m+2}^{n-d}\frac{1}{\epsilon^{(i-(m+1)+1)2^{m+1}-1}}\cdot\frac{f_i''(\epsilon)}{g_i''(\epsilon)}\cdot\mathsf{ESym}_{n,d+i};$$

this concludes the induction step. To complete the proof of the lemma, we observe that for $m = n - d$, Equation (8.3) is the same as Equation (8.1). □

The following lemma strengthens Theorem 2.2 which was proved in [MT18].

**Lemma 8.2** *For all $n \geq 10$ and $2 \leq k \leq \frac{n}{5}$, $\mathsf{ESym}_{n,n-k+1} \notin \sum_k \mathrm{ROF}(n)$.*

**Proof:** Fix an $n \geq 3$. We first prove the following special case of the lemma and then show how to reduce the general case to this special case. Recall the definition of the gate graph of an ROF (Definition 2.26). Also, recall that $\mathrm{fca}(x, y)$ denotes the first common ancestor of $x$ and $y$ in an ROF.

**Claim 8.1** *Let $f = \mathsf{C}_1 + \cdots + \mathsf{C}_k$ for any $2 \leq k \leq \frac{n}{3}$. If $\mathsf{C}_1, \ldots, \mathsf{C}_k$ have the same gate graph, then $\mathsf{C} \neq \mathsf{ESym}_{n,n-k+1}$.*

**Proof:** Since $\mathsf{C}_1, \ldots, \mathsf{C}_k$ have the same gate graph, we may assume that their gate graphs must be complete graphs. If not, then there will exist $x, y \in \mathbf{x}$ such that $\mathrm{fca}(x, y) = +$ in all of $\mathsf{C}_1, \ldots, \mathsf{C}_k$. Then from Observation 5.10, $\frac{\partial^2 \mathsf{C}_1}{\partial x \partial y} = \ldots = \frac{\partial^2 \mathsf{C}_k}{\partial x \partial y} = 0$ and no monomial in $\mathsf{C}$ can contain $xy$. Because $n \geq 3$ and $2 \leq k \leq \frac{n}{3}$, $n - k + 1 \geq 2$. Thus, there exists a monomial in $\mathsf{ESym}_{n,k}$ containing $xy$. The claim follows.

Define $\mathsf{D}_1 := \mathsf{C}$ and $\mathsf{D}_{1,i} = \mathsf{C}_i$ for all $i \in [k]$. Let $T_1$ be a product gate in $\mathsf{D}_{1,1}$ with the largest depth. Then, $T_1$ must be a product of at least two affine forms, each in just one variable. Without loss of generality, let $\alpha_{1,1}x_1 - \alpha_{1,0}$ and $y_1 - \beta_1$ be two of its factors. Define $\mathsf{D}_2 = \frac{\partial \mathsf{D}_1}{\partial x_1}(y_1 = \beta_1)$. Then $\mathsf{D}_2 = \mathsf{D}_{2,2} + \cdots + \mathsf{D}_{2,k}$, where for all $j \in \{2, \ldots, k\}$, $\mathsf{D}_{2,j} = \frac{\partial \mathsf{D}_{1,j}}{\partial x_1}(y_1 = \beta_1)$; this is so because $\frac{\partial \mathsf{D}_{1,1}}{\partial x_1}(y_1 = \beta_1) = 0$ (follows from Observation 5.11). Also, $\mathsf{D}_{2,2}, \cdots, \mathsf{D}_{2,k}$ have the same gate graph. Thus, we can repeat the above process with $\mathsf{D}_2$. Proceeding in this manner, we obtain $\mathsf{D}_i = \mathsf{D}_{i,i} + \cdots + \mathsf{D}_{i,k}$, and $x_i, y_i, \beta_i$ for all $i \in \{2, \ldots, k\}$ such that $\mathsf{D}_i = \frac{\partial \mathsf{D}_{i-1}}{\partial x_{i-1}}(y_{i-1} = \beta_{i-1})$ and $\frac{\partial \mathsf{D}_{i,i}}{\partial x_i}(y_i = \beta_i) = 0$. Now there are two cases:

**Case 1:** Not all of $\beta_1, \ldots, \beta_k$ are 0. Note that $\frac{\partial \mathsf{D}_k}{\partial x_k}(y_k = \beta_k) = \frac{\partial \mathsf{D}_{k,k}}{\partial x_k}(y_k = \beta_k) = 0$. Since $\frac{\partial^k \mathsf{C}}{\partial x_1 \cdots \partial x_k}(y_1 = \beta_1, \ldots, y_k = \beta_k) = \frac{\partial \mathsf{D}_k}{\partial x_k}(y_k = \beta_k)$, it is also 0. Now, as $k \leq \frac{n}{3}$, $\frac{\partial^k \mathsf{C}}{\partial x_1 \cdots \partial x_k} = \mathsf{ESym}_{m,m-k+1}(\mathbf{x}')$, where $\mathbf{x}' = \mathbf{x} \setminus \{x_1, \ldots, x_k\}$ and $m = n - k$. It is not difficult to see that

$$\mathsf{ESym}_{m,m-k+1}(y_1 = \beta_1, \ldots, y_k = \beta_k) \\ = \sum_{i=1}^{k} \mathsf{ESym}_{m-k,m-k+1-i}\left(\mathbf{x}' \setminus \{y_1, \ldots, y_k\}\right) \cdot \mathsf{ESym}_{k,i}(\beta_1, \ldots, \beta_k).$$

Here we use the fact that $k \leq \frac{n}{3}$, for otherwise $m - 2k + 1 \leq 0$. Thus,

$$\mathsf{ESym}_{m,m-k+1}(y_1 = \beta_1, \ldots, y_k = \beta_k) = 0$$

only if for all $i \in [k]$, $\mathsf{ESym}_{k,i}(\beta_1, \ldots, \beta_k) = 0$; this follows from the fact that

$$\left\{\mathsf{ESym}_{m-k,m-k+1-i}\left(\mathbf{x}' \setminus \{y_1, \ldots, y_k\}\right) : i \in [k]\right\}$$

is linearly independent. This only happens when $\beta_1, \ldots, \beta_k = 0$. As at least one of them is non-zero, $\mathsf{ESym}_{m,m-k+1}(y_1 = \beta_1, \ldots, y_k = \beta_k) \neq 0$. If $\mathsf{C}$ computes $\mathsf{ESym}_{n,n-k+1}$, then

$$0 = \frac{\partial^k \mathsf{C}}{\partial x_1 \cdots \partial x_k}(y_1 = \beta_1, \ldots, y_k = \beta_k) = \mathsf{ESym}_{m,m-k+1}(y_1 = \beta_1, \ldots, y_k = \beta_k) \neq 0.$$

Hence $\mathsf{C}$ can not compute $\mathsf{ESym}_{n,n-k+1}$.

**Case 2:** $\beta_1 = \cdots = \beta_k = 0$. In this case $\mathsf{C}$ computes $\mathsf{ESym}_{n,n-k+1}$ only if $\frac{\partial \mathsf{D}_{k-1}}{\partial x_{k-1}} = \frac{\partial \mathsf{D}_{k-1,k-1}}{\partial x_{k-1}} + \frac{\partial \mathsf{D}_{k-1,k}}{\partial x_{k-1}}$ computes

$$\frac{\partial^{k-1} \mathsf{C}}{\partial x_1 \cdots \partial x_{k-1}}(y_1 = 0, \ldots, y_{k-2} = 0) = \mathsf{ESym}_{m,m-1}(\mathbf{x}'),$$

where $m = n - (k-1) - (k-2)$ and $\mathbf{x}' = \mathbf{x} \setminus \{x_1, \ldots, x_{k-1}, y_1, \ldots, y_{k-2}\}$. The claim follows from Theorem 2.2. □

We now show how to reduce the general case to the above claim. Let $\mathsf{C} = \mathsf{C}_1 + \cdots + \mathsf{C}_k$ and $\mathsf{C}_1, \ldots, \mathsf{C}_k$ do not have the same gate graph. There are two cases:

**Case 1:** Not all of the $\mathsf{C}_1, \ldots, \mathsf{C}_k$ contain all variables in $\mathbf{x}$. Without loss of generality $\mathsf{C}_k$ does

not contain $x$. Then we define $\mathtt{C}' = \frac{\partial \mathtt{C}}{\partial x}$ and $\mathtt{C}'_i = \frac{\partial \mathtt{C}_i}{\partial x}$ for all $i \in [k]$; note that $\mathtt{C}'_k = 0$. Now $\mathtt{C}$ computes $\mathsf{ESym}_{n,n-k+1}$ only if $\mathtt{C}' = \mathtt{C}'_1 + \cdots + \mathtt{C}'_{k-1}$ computes $\mathsf{ESym}_{n-1,(n-1)-k+1}$.

**Case 2:** Each of the $\mathtt{C}_1, \ldots, \mathtt{C}_k$ contain all variables in $\mathbf{x}$. Then there must exist an $i \in [k]$ and $x, y \in \mathbf{x}$ such that $\mathrm{fca}(x, y)$ in $\mathtt{C}_i$ is a $+$ gate; without loss of generality $i = k$. Define $\mathtt{C}' = \frac{\partial^2 \mathtt{C}}{\partial x \partial y}$, $\mathtt{C}'_i = \frac{\partial^2 \mathtt{C}_i}{\partial x \partial y}$ for all $i \in [k-1]$. Notice that $\mathtt{C}$ computes $\mathsf{ESym}_{n,n-k+1}$ only if $\mathtt{C}'$ computes $\mathsf{ESym}_{n-2,(n-2)+k-1}(\mathbf{x} \setminus \{x, y\})$.

Now, if $\mathtt{C}'_1, \ldots, \mathtt{C}'_{k-1}$ have the same gate graph, then we have successfully reduced to the case of Claim 8.1. Otherwise, we repeat the process described in Cases 1 and 2 until we get a $\mathtt{C}'' = \mathtt{C}''_1 + \cdots + \mathtt{C}''_\ell$ such that $\mathtt{C}''_1, \ldots, \mathtt{C}''_\ell$ have the same gate graph and $\mathtt{C}$ computes $\mathsf{ESym}_{n,n-k+1}$ only if $\mathtt{C}''$ computes $\mathsf{ESym}_{n-m,(n-m)-k+1}(\mathbf{x}')$, where $0 \leq m \leq 2(k-\ell) \leq 2(k-1)$ and $|\mathbf{x}'| = n - m$. This is so because the process needs to be repeated at most $k-1$ times and every time the above process reduces the number of summands by 1, the number of variables as well as the degree of $\mathsf{ESym}$ both decrease by either 1 or 2.

If $\ell = 1$, then we need to show that $\mathtt{C}''$ which is an ROF can not compute $\mathsf{ESym}_{n-m,(n-m)-k+1}$. As $2 \leq k \leq \frac{n}{5}$, $(n-m) - k + 1 \geq 3$. Now the gate graph of $\mathtt{C}''$ must be a complete graph, for otherwise there would be a pair of variables $x, y \in \mathbf{x}'$ such that $\frac{\partial^2 \mathtt{C}''}{\partial x \partial y} = 0$ while $\frac{\partial^2 \mathsf{ESym}_{n-m,(n-m)-k+1}}{\partial x \partial y} \neq 0$. But if the gate graph of $\mathtt{C}''$ is a complete graph, then it must compute $\prod_{x \in \mathbf{x}'} x$ which is not present in $\mathsf{ESym}_{n-m,(n-m)-k+1}$. So $\mathtt{C}''$ can not compute $\mathsf{ESym}_{n-m,(n-m)-k+1}$. If $\ell \geq 2$, then a simple calculation shows that as $2 \leq k \leq \frac{n}{5}$, implies $2 \leq k \leq \frac{n-m}{3}$. So we can use Claim 8.1 for a $\mathtt{C}'''$ obtained from $\mathtt{C}''$ by adding an $k - \ell$ ROFs with the same gate graphs as $\mathtt{C}''_1, \ldots, \mathtt{C}''_\ell$ whose sum equals 0. □

## 8.3 De-bordering the border of sum of 2 ROFs

In this section, we show that the border of sum of two additive constant free ROFs computing an $n$- variate polynomial is contained in the sum of $O(n)$ many ROFs. Recall that we say that an ROF is additive constant free if all children of every $+$ gate in it are non-constant polynomials. Also recall that we denote the class of additive constant free ROFs by $\mathrm{ROF}_0$.

### Proof of Theorem 1.11

The proof is by induction on $n$. For $n = 1$, and any $f \in \overline{\sum_2 \mathrm{ROF}_0(n)}$, $f + O(\epsilon) = \alpha x_1 + \beta x_1$ for some $\alpha, \beta \in \mathbb{F}(\epsilon)$ such that $\alpha + \beta \in \mathbb{F}[\epsilon]$. Putting $\epsilon = 0$ in $f + O(\epsilon) = (\alpha + \beta) x_1$, we get $f \in \mathrm{ROF}_0$.

Assume by the way of induction that the theorem is true for $1 \leq n' < n$. Fix an $f \in \overline{\sum_2 \mathrm{ROF}_0(n)}$. Then $f + O(\epsilon) = R_1 + R_2$ where $R_1$ and $R_2$ are additive constant free ROFs over $\mathbb{F}(\epsilon)$. We divide the proof into two cases depending on whether (one of) the deepest gate in $R_1$, say $G$, is a $+$ gate or a $\times$ gate.

**Case 1.** $G$ is a $\times$ gate. Notice that since $G$ is (one of) the deepest gate in $R_1$, it can only compute a product of variables in $\mathbf{x}$. Moreover, it must have degree at least 2. Then without loss of generality $G = x_1 x_2 \cdots x_m$. We further divide this case into sub-cases depending on what $\mathrm{fca}(x_1, x_2)$ in $R_2$ is.

Case 1.1. $\mathrm{fca}(x_1, x_2)$ in $R_2$ is a $+$ gate. Note that $\frac{\partial f}{\partial x_1} + O(\epsilon) = \frac{\partial R_1}{\partial x_1} + \frac{\partial R_2}{\partial x_1}$ and thus $\frac{\partial f}{\partial x_1} \in \overline{\sum_2 \mathrm{ROF}_0(n-1)}$.[1] Also, as polynomials over $\mathbb{F}(\epsilon)$, every monomial in $\frac{\partial R_1}{\partial x_1}$ contains $x_2$ (see Observation 5.11)[2] while no monomial in $\frac{\partial R_2}{\partial x_1}$ has $x_2$. Thus, there can be no cancellations between $\frac{\partial R_1}{\partial x_1}$ and $\frac{\partial R_2}{\partial x_1}$. Hence $\frac{\partial R_2}{\partial x_1} \in \mathbb{F}[\epsilon][\mathbf{x}]$ and Claim 2.3 implies that $\frac{\partial R_2}{\partial x_1}(\epsilon = 0) \in \mathrm{ROF}_0$. Because $R_2$ is a multilinear polynomial, $R_2 = x_1 \frac{\partial R_2}{\partial x_1} + R_2(x_1 = 0)$. Hence, $f + O(\epsilon) = (R_1 + R_2(x_1 = 0)) + R_3$, where $R_3 = x_1 \cdot \frac{\partial R_2}{\partial x_1}(\epsilon = 0)$ is an ROF (over $\mathbb{F}$).

We can repeat the argument made above for $f' := (R_1 + R_2(x_1 = 0))(\epsilon = 0)$, to get that $f + O(\epsilon) = (R_1 + R_2(x_1, x_2 = 0)) + R_3 + R_4$, where $R_4$ is also an ROF (over $\mathbb{F}$). Observe that a monomial in $R_1 + R_2(x_1, x_2 = 0)$ contains $x_1$ if and only if it contains $x_2$. Consider $f'' := (R_1' + R_2(x_1, x_2 = 0))(\epsilon = 0)$, where $R_1'$ is obtained from $R_1$ by replacing $G$ with $y x_3 \cdots x_m$ where $y \notin \mathbf{x}$ is a fresh variable. As $f'' \in \overline{\sum_2 \mathrm{ROF}_0(n-1)}$, by the induction hypothesis, $f'' \in \sum_{O(n-1)} \mathrm{ROF}_0(n-1)$, say $f'' = P_1 + \cdots + P_m$, where $m = O(n-1)$. For all $i \in [m]$, let $P_i' = P_i(y = x_1 x_2)$. Observe that $P_1' + \cdots + P_m' = (R_1 + R_2(x_1, x_2 = 0))(\epsilon = 0)$. Further $P_1', \ldots, P_m' \in \mathrm{ROF}_0$. Hence, $f + O(\epsilon) = P_1' + \cdots + P_m' + O(\epsilon) + R_3 + R_4$ and $f \in \sum_{O(n)} \mathrm{ROF}_0$.

Case 1.2. $\mathrm{fca}(x_1, x_2)$ in $R_2$ is a $\times$ gate; suppose this gate is at product-depth $\delta - 1$. We assume that the top gate of $R_2$ is a $+$ gate, if it is not, we add a dummy $+$ gate. Then, we can write $R_2$ as $Q_{1,1} + \cdots + Q_{1,s_1}$ where each of the $Q_{1,i}$ are $\times$ gates (or 0 if the top $+$ gate is a dummy gate) and $x_1, x_2 \in \mathrm{var}(Q_{1,1})$. Let $Q_{1,1} = T_{1,1} \cdots T_{1,m_1}$ and $x_1, x_2 \in \mathrm{var}(T_{1,1})$. Having inductively defined $Q_{i,j}$ and $T_{i,j'}$ for all $i \in [\ell]$, $\ell < \delta - 1$, $j \in [s_i]$, and $j' \in [m_i]$, let, $T_{\ell,1} = Q_{\ell+1,1} + \cdots + Q_{\ell+1,s_{\ell+1}}$, with $x_1, x_2 \in \mathrm{var}(Q_{\ell+1,1})$, and $Q_{\ell+1,1} = T_{\ell+1,1} \cdots T_{\ell+1,m_{\ell+1}}$, with $x_1, x_2 \in \mathrm{var}(T_{\ell+1,1})$. Finally, $T_{\delta-1} = Q_{\delta,1} + \cdots + Q_{\delta,s_\delta}$ with $x_1, x_2 \in \mathrm{var}(Q_{\delta,1})$ and

[1] This is so, becasue the derivative of an additive constant free ROF is also an additive constant free ROF.

[2] Even though we have only proved Observation 5.11 for canonical ROFs, it actually holds for any ROF with alternating layers of $+$ and $\times$ gates. Recall that all ROFs considered in this work have this structure.

$Q_{\delta,1} = T_{\delta,1} \cdot T_{\delta,2} \cdots T_{\delta,m_\delta}$, where $x_1 \in \mathrm{var}(T_{\delta,1})$ and $x_2 \in \mathrm{var}(T_{\delta,2})$. That is $\mathrm{fca}(x_1, x_2) = Q_{\delta,1}$.

Observe that as $\frac{\partial R_1}{\partial x_1}(x_2 = 0) = \frac{\partial R_1}{\partial x_2}(x_1 = 0) = 0$, $\frac{\partial R_2}{\partial x_1}(x_2 = 0)$ and $\frac{\partial R_2}{\partial x_2}(x_1 = 0) \in \mathbb{F}[\epsilon][\mathbf{x}]$. Thus $\left(\frac{\partial R_2}{\partial x_1}(x_2 = 0)\right)(\epsilon = 0)$ and $\left(\frac{\partial R_2}{\partial x_2}(x_1 = 0)\right)(\epsilon = 0)$ are both in $\overline{\mathrm{ROF}_0} = \mathrm{ROF}_0$. Now,

$$\frac{\partial R_2}{\partial x_1}(x_2 = 0) = \frac{\partial T_{\delta,1}}{\partial x_1} \cdot T_{\delta,2}(x_2 = 0) \cdot T_{\delta,3} \cdots T_{\delta,m_\delta} \cdot \prod_{i=1}^{\delta-1} T_{i,2} \cdots T_{i,m_i}$$
$$\frac{\partial R_2}{\partial x_2}(x_1 = 0) = \frac{\partial T_{\delta,2}}{\partial x_2} \cdot T_{\delta,1}(x_1 = 0) \cdot T_{\delta,3} \cdots T_{\delta,m_\delta} \cdot \prod_{i=1}^{\delta-1} T_{i,2} \cdots T_{i,m_i}.$$

Let $\mathsf{val}\left(T_{\delta,3} \cdots T_{\delta,m_\delta} \cdot \prod_{i=1}^{\delta-1} T_{i,2} \cdots T_{i,m_i}\right) = \gamma$ and $\mathsf{val}\left(\frac{\partial T_{\delta,1}}{\partial x_1}\right) = \alpha_1$, $\mathsf{val}\left(T_{\delta,2}(x_2 = 0)\right) = \beta_2$, $\mathsf{val}\left(\frac{\partial T_{\delta,2}}{\partial x_2}\right) = \alpha_2$, $\mathsf{val}\left(T_{\delta,1}(x_1 = 0)\right) = \beta_1$. Since $\frac{\partial R_2}{\partial x_1}(x_2 = 0), \frac{\partial R_2}{\partial x_2}(x_1 = 0) \in \mathbb{F}[\epsilon][\mathbf{x}]$, $\alpha_1 + \beta_2 + \gamma \geq 0$ and $\alpha_2 + \beta_1 + \gamma \geq 0$. A simple calculation shows that this implies that $\alpha_1 + \alpha_2 + \gamma \geq 0$ or $\beta_1 + \beta_2 + \gamma \geq 0$. We now analyse these two cases.

Let $R_3 = \left(x_1 \frac{\partial R_2}{\partial x_1}(x_2 = 0)\right)(\epsilon = 0)$ and $R_4 = \left(x_2 \frac{\partial R_2}{\partial x_2}(x_1 = 0)\right)(\epsilon = 0)$. If $\alpha_1 + \alpha_2 + \gamma \geq 0$, then

$$\frac{\partial^2 R_2}{\partial x_1 \partial x_2}(\epsilon = 0) = \left(\frac{\partial T_{\delta,1}}{\partial x_1} \cdot \frac{\partial T_{\delta,2}}{\partial x_2} \cdot T_{\delta,3} \cdots T_{\delta,m_\delta} \cdot \prod_{i=1}^{\delta-1} T_{i,2} \cdots T_{i,m_i}\right)(\epsilon = 0)$$
$$\in \overline{\mathrm{ROF}_0} = \mathrm{ROF}_0.$$

Thus we can write

$$f + O(\epsilon) = (R_1 + R_2') + R_3 + R_4 + R_5$$

where $R_5 := x_1 x_2 \frac{\partial^2 R_2}{\partial x_1 \partial x_2}(\epsilon = 0) \in \mathrm{ROF}_0$, and $R_2' = R_2(x_1, x_2 = 0)$. Then we can argue as in Case 1.1: obtain $R_1'$ by replacing $x_1 x_2$ in $R_1$ by $y$ and define $f' := (R_1' + R_2')(\epsilon = 0)$. By induction hypothesis $f' \in \sum_{O(n-1)} \mathrm{ROF}_0(n-1)$ from which we can obtain a circuit for $f$ in $\sum_{O(n)} \mathrm{ROF}_0(n)$ by replacing $y$ with $x_1 x_2$ and adding $R_3 + R_4 + R_5$.

Similarly, if $\beta_1 + \beta_2 + \gamma \geq 0$ then we can write

$$f + O(\epsilon) = (R_1 + R_2') + R_3 + R_4 + R_5$$

where $R_5 = \left(R_2(x_1, x_2 = 0)\right)(\epsilon = 0) \in \overline{\mathrm{ROF}_0} = \mathrm{ROF}_0$ and $R_2' = x_1 x_2 \frac{\partial^2 R_2}{\partial x_1 \partial x_2}$. Note that a monomial in $R_1$ or $R_2'$ contains $x_1$ if and only if contains $x_2$. Let $R_1'$ and $R_2''$ be obtained from $R_1$ and $R_2'$ by replacing $x_1 x_2$ with a fresh variable $y$ and define $f' := (R_1' + R_2'')(\epsilon = 0)$. As

before, by the induction hypothesis, $f' \in \sum_{O(n-1)} \mathrm{ROF}_0(n-1)$. We can obtain a circuit for $f$ in $\sum_{O(n)} \mathrm{ROF}_0(n)$ from that of $f'$ by replacing $y$ with $x_1x_2$ and adding $R_3 + R_4 + R_5$.

**Case 2.** $G$ is a $+$ gate. Since $G$ is the deepest gate in $R_1$, it must compute a linear form. Without loss of generality $G = \alpha_1 x_1 + \cdots + \alpha_m x_m$, where $\mathsf{val}(\alpha_1) \leq \mathsf{val}(\alpha_2)$; in fact we can also assume that $\alpha_1 = 1$. Consider $f' := f\left(x_1 = x_1 - \alpha_2(\epsilon = 0)x_2\right) = f(x_1 = x_1 - \alpha_2 x_2) + O(\epsilon)$. Then $f' + O(\epsilon) = R_1' + R_2'$, where $R_1' = R_1(x_1 = x_1 - \alpha_2 x_2)$ and $R_2' = R_2(x_1 = x_1 - \alpha_2 x_2)$. Note that $R_1'$ does not contain $x_2$, but $R_2'$ need not be an ROF over $\mathbb{F}(\epsilon)$; it may contain $x_2$ at two places. $x_2\frac{\partial R_2'}{\partial x_2}$ can not cancel out and is in $\mathbb{F}[\epsilon][\mathbf{x}]$. We now analyse $x_2\frac{\partial R_2'}{\partial x_2}$. There are two cases: $\mathrm{fca}(x_1, x_2)$ in $R_2$ is a $+$ gate and $\mathrm{fca}(x_1, x_2)$ in $R_2$ is a $\times$ gate.

Case 2.1: $\mathrm{fca}(x_1, x_2)$ in $R_2$ is a $+$ gate. If there is a $+$ gate in $R_2$ such that $x_1$ and $x_2$ are directly connected to it, then the same is true in $R_2'$. Let us call this gate in $R_2'$ as $G'$. Then $R_2'$ is an ROF over $\mathbb{F}(\epsilon)$. Hence $x_2\frac{\partial R_2'}{\partial x_2}(\epsilon = 0) \in \overline{\mathrm{ROF}_0} = \mathrm{ROF}_0$. Then, $f' + O(\epsilon) = (R_1' + R_2'(x_2 = 0)) + R_3$, where $R_3 = x_2\frac{\partial R_2'}{\partial x_2}(\epsilon = 0)$. Now $f'' := (R_1' + R_2'(x_2 = 0))\,(\epsilon = 0) \in \overline{\sum_2 \mathrm{ROF}_0(n-1)}$ and thus from the induction hypothesis, $f'' \in \sum_{O(n-1)} \mathrm{ROF}_0(n-1)$. Hence, $f' \in \sum_{O(n)} \mathrm{ROF}_0(n)$. We can obtain a circuit for $f$ from that for $f'$ by replacing $x_1$ with $x_1 + \alpha_2(\epsilon = 0)x_2$. Since $f''$ does not contain $x_2$ and $R_3$ does not contain $x_1$, the circuit so obtained will still be in $\sum_{O(n)} \mathrm{ROF}_0(n)$. Now we look at the case in which such a $G'$ does not exist.

As in Case 1.2 we assume without loss of generality that the top gate of $R_2$ is a $+$ gate. Suppose that $\mathrm{fca}(x_1, x_2)$ is at product-depth $\delta$. Write $R_2$ as $Q_{1,1} + \cdots + Q_{1,s_1}$ where each of the $Q_{1,i}$ are $\times$ gates (or 0 if the top gate of $R_2$ is a $\times$ gate) and $x_1, x_2 \in \mathrm{var}(Q_{1,1})$. $Q_{1,1} = T_{1,1} \cdots T_{1,m_1}$ and $x_1, x_2 \in \mathrm{var}(T_{1,1})$. Having inductively defined $Q_{i,j}$ and $T_{i,j'}$ for all $i \in [\ell]$, $\ell < \delta$, $j \in [s_i]$, and $j' \in [m_i]$, let $T_{\ell,1} = Q_{\ell+1,1} + \cdots + Q_{\ell+1,s_{\ell+1}}$, with $x_1, x_2 \in \mathrm{var}(Q_{\ell+1,1})$, and $Q_{\ell+1,1} = T_{\ell+1,1} \cdots T_{\ell+1,m_{\ell+1}}$, with $x_1, x_2 \in \mathrm{var}(T_{\ell+1,1})$. Finally, $T_{\delta,1} = Q_{\delta+1,1} + \cdots + Q_{\delta+1,s_{\delta+1}}$ with $x_1, \in \mathrm{var}(Q_{\delta+1,1})$ and $x_2 \in \mathrm{var}(Q_{\delta+1,2})$. Since $G'$ does not exist, it must be that at least one of $Q_{\delta+1,1}$, $Q_{\delta+1,2}$ has fan-in at least 2. Because $R_1'$ does not contain $x_2$, $\frac{\partial R_2'}{\partial x_2} \in \mathbb{F}[\epsilon](\mathbf{x} \setminus \{x_2\})$ . Now,

$$\begin{aligned}\frac{\partial R_2'}{\partial x_2} &= \frac{\partial T_{\delta,1}(x_1 = x_1 - \alpha_2 x_2)}{\partial x_2} \cdot T_{\delta,2} \cdots T_{\delta,m_\delta} \cdot \prod_{i=1}^{\delta-1} T_{i,2} \cdots T_{i,m_i} \\ &= \left(\frac{\partial Q_{\delta+1,1}(x_1 = x_1 - \alpha_2 x_2)}{\partial x_2} + \frac{\partial Q_{\delta+1,2}}{\partial x_2}\right) \cdot T_{\delta,2} \cdots T_{\delta,m_\delta} \cdot \prod_{i=1}^{\delta-1} T_{i,2} \cdots T_{i,m_i}.\end{aligned}$$

Suppose that $Q_{\delta+1,1}$ has fan-in at least 2, the case when $Q_{\delta+1,2}$ has fan-in at least 2 is similar. Let $Q_{\delta+1,1} = T_{\delta+1,1} \cdots T_{\delta+1,m_{\delta+1}}$ with $x_1 \in \text{var}(T_{\delta+1,1})$ and $m_{\delta+1} \geq 2$. Then as $R_2$ and thus $R_2'$ is additive constant free, every monomial in $Q_{\delta+1,1}(x_1 = x_1 - \alpha_2 x_2)$ containing $x_2$ also contains a variable in $\text{var}(T_{\delta+1,2})$. Because $R_2$ is an ROF, $\text{var}(Q_{\delta+1,1})$ and $\text{var}(Q_{\delta+1,2})$ are disjoint. Thus $\text{var}(Q_{\delta+1,1})(x_1 = x_1 - \alpha_2 x_2) \cap \text{var}(Q_{\delta+1,2}) = x_2$. Hence, every monomial in $\frac{\partial Q_{\delta+1,1}}{\partial x_2}(x_1 - \alpha_2 x_2)$ contains a variable in $\text{var}(T_{\delta+1,2})$ which is not in $\frac{\partial Q_{\delta+1,2}}{\partial x_2}$. So no cancellations are possible between $R_3' := x_2 \frac{\partial Q_{\delta+1,1}(x_1 = x_1 - \alpha_2 x_2)}{\partial x_2} \cdot T_{\delta,2} \cdots T_{\delta,m_\delta} \cdot \prod_{i=1}^{\delta-1} T_{i,2} \cdots T_{i,m_i}$ and $R_4' := x_2 \frac{\partial Q_{\delta+1,1}}{\partial x_2} \cdot T_{\delta,2} \cdots T_{\delta,m_\delta} \cdot \prod_{i=1}^{\delta-1} T_{i,2} \cdots T_{i,m_i}$. Thus both of them are in $\mathbb{F}[\epsilon](\mathbf{x} \setminus \{x_2\})$. Also note that both are ROFs over $\mathbb{F}(\epsilon)$. Thus $R_3 := R_3'(\epsilon = 0)$ and $R_4 := R_4'(\epsilon = 0)$ both are ROFs over $\mathbb{F}$.

Notice that $R_2'$ computes a multilinear polynomial. So,

$$f' + O(\epsilon) = (R_1' + R_2'(x_2 = 0)) + R_3 + R_4. \tag{8.4}$$

$f'' := (R_1' + R_2'(x_2 = 0))\,(\epsilon = 0) \in \overline{\sum_2 \text{ROF}_0(n-1)}$; so by induction hypothesis, $f'' \in \sum_{O(n-1)} \text{ROF}_0(n-1)$. So, $f' \in \sum_{O(n)} \text{ROF}_0(n)$. We can obtain a circuit for $f$ from $f'$ by replacing $x_1$ with $x_1 + \alpha_2(\epsilon = 0)x_2$. Now $R_3, R_4$ do not contain $x_1$; this is because $R_2'$ does not contain any monomial containing both $x_1$ and $x_2$. Hence $f$ continues to be in $\sum_{O(n)} \text{ROF}_0(n)$.

Case 2.2 $\text{fca}(x_1, x_2)$ in $R_2$ is a $\times$ gate. As in Case 1.2, we assume without loss of generality that the top gate of $R_2$ is a $+$ gate. Suppose that $\text{fca}(x_1, x_2)$ is at product-depth $\delta - 1$. Write $R_2$ as $Q_{1,1} + \cdots + Q_{1,s_1}$ where each of the $Q_{1,i}$ are $\times$ gates (or 0 if the top gate of $R_2$ is a $\times$ gate) and $x_1, x_2 \in \text{var}(Q_{1,1})$. $Q_{1,1} = T_{1,1} \cdots T_{1,m_1}$ and $x_1, x_2 \in \text{var}(T_{1,1})$. Having inductively defined $Q_{i,j}$ and $T_{i,j'}$ for all $i \in [\ell]$, $\ell < \delta - 1$, $j \in [s_i]$, and $j' \in [m_i]$, let $T_{\ell,1} = Q_{\ell+1,1} + \cdots + Q_{\ell+1,s_{\ell+1}}$, with $x_1, x_2 \in \text{var}(Q_{\ell+1,1})$, and $Q_{\ell+1,1} = T_{\ell+1,1} \cdots T_{\ell+1,m_{\ell+1}}$, with $x_1, x_2 \in \text{var}(T_{\ell+1,1})$. Finally, $T_{\delta-1} = Q_{\delta,1} + \cdots + Q_{\delta,s_\delta}$ with $x_1, x_2 \in \text{var}(Q_{\delta,1})$ and $Q_{\delta,1} = T_{\delta,1} \cdot T_{\delta,2} \cdots T_{\delta,m_\delta}$, where $x_1 \in \text{var}(T_{\delta,1})$ and $x_2 \in \text{var}(T_{\delta,2})$. That is $\text{fca}(x_1, x_2) = Q_{\delta,1}$.

Observe that $R_2'$ has individual degree at most 1 in $\mathbf{x} \setminus \{x_2\}$ and individual degree at most 2 in $x_2$. Hence, as $\text{char}(\mathbb{F}) \neq 2$,

$$R_2' = R_2'(x_2 = 0) + x_2 \frac{\partial R_2'}{\partial x_2}(x_2 = 0) + \frac{1}{2} x_2^2 \frac{\partial^2 R_2'}{\partial x_2^2}.$$

As in $f'$, $R_2'$ is the only sub-circuit containing $x_2$, $x_2 \frac{\partial R_2'}{\partial x_2}(x_2 = 0) + \frac{1}{2} x_2^2 \frac{\partial^2 R_2'}{\partial x_2^2} \in \mathbb{F}[\epsilon][\mathbf{x}]$. This also means that both the summands are also in $\mathbb{F}[\epsilon][\mathbf{x}]$.

In $R_2'$ the only sub-circuits with $x_2$ are $T_{\delta,1}(x_1 = x_1 - \alpha_2 x_2)$ and $T_{\delta,2}$. So,

$$\frac{\partial^2 R_2'}{\partial x_2^2} = 2\left(-\alpha_2 \cdot \frac{\partial T_{\delta,1}}{\partial x_1} \cdot \frac{\partial T_{\delta,2}}{\partial x_2} \cdot T_{\delta,3} \cdots T_{\delta,m_\delta}\right) \cdot \prod_{i=1}^{\delta-1} T_{i,2} \cdot T_{i,m_i}$$

Thus, $\frac{\partial^2 R_2'}{\partial x_2^2}(\epsilon = 0) \in \overline{\text{ROF}_0} = \text{ROF}_0$. Call this ROF $R_3$.

$$\begin{aligned}
&\frac{\partial R_2'}{\partial x_2}(x_2 = 0) \\
&= \left(-\alpha_2 \cdot \frac{\partial T_{\delta,1}}{\partial x_1} \cdot T_{\delta,2}(x_2 = 0) + T_{\delta,1} \cdot \frac{\partial T_{\delta,2}}{\partial x_2}\right) \cdot T_{\delta,3} \cdots T_{\delta,m_\delta} \cdot \prod_{i=1}^{\delta-1} T_{i,2} \cdot T_{i,m_i} \\
&= \left(-\alpha_2 \cdot \frac{\partial T_{\delta,1}}{\partial x_1} \cdot T_{\delta,2}(x_2 = 0) + \left(x_1 \frac{\partial T_{\delta,1}}{\partial x_1} + T_{\delta,1}(x_1 = 0)\right) \cdot \frac{\partial T_{\delta,2}}{\partial x_2}\right) \cdot T_{\delta,3} \cdots T_{\delta,m_\delta} \cdot \prod_{i=1}^{\delta-1} T_{i,2} \cdot T_{i,m_i}.
\end{aligned}$$

Unless $T_{\delta,2}(x_2 = 0) = 0$, every monomial in $T_{\delta,2}(x_2 = 0)$ must contain a variable $y$ such that $\text{fca}(y, x_2)$ in $R_2$ is a $+$ gate. Such a variable can not be present in $T_{\delta,1} \cdot \frac{\partial T_{\delta,2}}{\partial x_2}$. Thus,

$$-\alpha_2 \cdot \frac{\partial T_{\delta,1}}{\partial x_2} \cdot T_{\delta,2}(x_2 = 0) \cdot \cdot T_{\delta,3} \cdots T_{\delta,m_\delta} \cdot \prod_{i=1}^{\delta-1} T_{i,2} \cdot T_{i,m_i} \in \mathbb{F}[\epsilon][\mathbf{x}].$$

Also, it is an ROF over $\mathbb{F}(\epsilon)$, so

$$\left(-\alpha_2 \cdot \frac{\partial T_{\delta,1}}{\partial x_2} \cdot T_{\delta,2}(x_2 = 0) \cdot \cdot T_{\delta,3} \cdots T_{\delta,m_\delta} \cdot \prod_{i=1}^{\delta-1} T_{i,2} \cdot T_{i,m_i}\right)(\epsilon = 0) \in \text{ROF}_0.$$

Call this ROF $R_4$. $\mathbb{F}[\epsilon][\mathbf{x}]$ also contains

$$\begin{aligned}
&T_{\delta,1} \cdot \frac{\partial T_{\delta,2}}{\partial x_2} \cdot T_{\delta,3} \cdots T_{\delta,m_\delta} \cdot \prod_{i=1}^{\delta-1} T_{i,2} \cdot T_{i,m_i} \\
&= \left(x_1 \frac{\partial T_{\delta,1}}{\partial x_1} + T_{\delta,1}(x_1 = 0)\right) \cdot \frac{\partial T_{\delta,2}}{\partial x_2} \cdot T_{\delta,3} \cdots T_{\delta,m_\delta} \cdot \prod_{i=1}^{\delta-1} T_{i,2} \cdot T_{i,m_i}.
\end{aligned}$$

Thus,

$$\left(x_1 \frac{\partial T_{\delta,1}}{\partial x_1} \cdot \frac{\partial T_{\delta,2}}{\partial x_2} \cdot T_{\delta,3} \cdots T_{\delta,m_\delta} \cdot \prod_{i=1}^{\delta-1} T_{i,2} \cdot T_{i,m_i}\right)(\epsilon = 0)$$

and

$$\left(T_{\delta,1}(x_1=0)\cdot\frac{\partial T_{\delta,2}}{\partial x_2}\cdot T_{\delta,3}\cdots T_{\delta,m_\delta}\cdot\prod_{i=1}^{\delta-1}T_{i,2}\cdot T_{i,m_i}\right)(\epsilon=0)$$

are both in $\overline{\text{ROF}_0} = \text{ROF}_0$. Call these two first ROFs $R_5$ and $R_6$. Then,

$$f' + O(\epsilon) = (R_1' + R_2'(x_2 = 0)) + x_2(R_4 + R_5 + R_6) + \frac{1}{2}x_2^2 R_3.$$

Note that $f'' := (R_1' + R_2'(x_2 = 0))(\epsilon = 0) \in \overline{\sum_2 \text{ROF}_0(n-1)}$. So by the induction hypothesis, $f'' \in \sum_{O(n-1)} \text{ROF}_0(n-1)$. Let $\alpha_2 = \alpha_{2,0} + O(\epsilon)$. Then,

$$\begin{aligned} f &= f'(x_1 = x_1 + \alpha_{2,0}x_2) \\ &= f''(x_1 = x_1 + \alpha_{2,0}x_2) + x_2(R_4 + R_6) + x_2(R_5(x_1 = x_1 + \alpha_{2,0}x_2)) + \frac{1}{2}x_2^2 R_3 \end{aligned}$$

Note that $R_4$ and $R_6$ do not contain $x_2$. So $x_2R_4$ and $x_2R_6$ are both ROFs. Then, from the proof of Claim 2.3, $R_3$ and $R_5$ look like $-2\alpha_{2,0}R'$ and $x_1R'$ respectively for some ROF $R'$. So, $x_2(R_5(x_1 = x_1 + \alpha_{2,0}x_2)) + \frac{1}{2}x_2^2R_6 = x_1x_2R'$. As $R_5$ does not contain $x_2$, $x_2R'$ is an ROF giving $f \in \sum_{O(n)} \text{ROF}_0(n)$.

## 8.4 PIT for the border of sum of 2 depth-4 ROFs

For ease of exposition, we first prove Theorem 1.12 for the special case of the border of sum of 2 depth-4 ROFs. The proof of the theorem follows in a straightforward way from the following lemma.

**Lemma 8.3** *Let $\mathcal{G} : \mathbb{F}^m \to \mathbb{F}^n$ be a hitting set generator for $n$-variate multilinear ROABPs as well as $\sum_2$ ROF. Then $\mathcal{G} + \mathcal{G}_3^{SV}$ is a hitting set generator for the border of sums of 2 additive constant free, depth-4 ROFs provided that $|\mathbb{F}| > n^2d$, where $d$ is the degree of $\mathcal{G} + \mathcal{G}_3^{SV}$.*[1]

**Proof:** Let $f + O(\epsilon) = R_1 + R_2$, where $R_1, R_2$ are additive constant free depth-4 ROFs over $\mathbb{F}(\epsilon)$ and $f \not\equiv 0 \in \mathbb{F}[\mathbf{x}]$. From Observation 2.2 it is sufficient to show that $\mathcal{G} + \mathcal{G}_2^{SV}$ is a hitting set generator for $\frac{\partial f}{\partial x_i}$ for some $i \in [n]$. If $f \in \mathbb{F}$, there is nothing to prove. Else, there exists an $i \in [n]$ such that $\frac{\partial f}{\partial x_i} \not\equiv 0$. Observe that $\frac{\partial R_1}{\partial x_i}$ and $\frac{\partial R_2}{\partial x_i}$ are in $\text{ROF}_0$ over $\mathbb{F}(\epsilon)$ with depth 3 and a product gate at top. So by re-defining $f = \frac{\partial f}{\partial x_i}$, $R_1 = \frac{\partial R_1}{\partial x_i}$, $R_2 = \frac{\partial R_2}{\partial x_i}$, we can reduce the lemma to the following: Let $R_1 = p_1 \cdots p_k$ and $R_2 = q_1 \cdots q_\ell$, then $\mathcal{G} + \mathcal{G}_2^{SV}$ is a hitting set generator for $f$. For all $i \in [k]$, let $p_i = \alpha_{i,1}m_{i,1} + \cdots + \alpha_{i,k_i}m_{i,k_i}$ where $m_{i,j}$ are monomials.

[1] For a polynomial map $\mathcal{G} = (g_1, \ldots, g_n)$, $\deg(\mathcal{G}) := \max\{\deg(g_i) : i \in [n]\}$.

Similarly, for all $i \in [\ell]$, let $q_i = \beta_{i,1}\mu_{i,1} + \cdots + \beta_{i,\ell_i}\mu_{i,\ell_i}$ where $\mu_{i,j}$ are monomials. We want to show that $f(\mathcal{G} + \mathcal{G}_2^{SV}) \equiv 0$ implies that $f \equiv 0$. We first show that $f(\mathcal{G} + \mathcal{G}_2^{SV}) \equiv 0$, then $\{m_{i,j}\}_{i,j} = \{\mu_{i,j}\}_{i,j}$.

**Claim 8.2** *If $f(\mathcal{G} + \mathcal{G}_2^{SV}) \equiv 0$, then $\{m_{i,j} : i \in [k], j \in k_i\} = \{\mu_{i,j} : i \in [\ell], j \in \ell_i\}$.*

**Proof:** The proof is by induction on the number of variables $n$ in $f$. If $n = 1$, then $R_1 = \alpha x_1$ and $R_2 = \beta x_1$ and the claim is clearly true. So assume, by the way of induction, that the claim is true for all $f'$ in the border of sum of 2 additive constant free depth-4 ROFs with at most $n-1$ variables. If $\{m_{i,j} : i \in [k], j \in k_i\} \neq \{\mu_{i,j} : i \in [\ell], j \in \ell_i\}$, then either there exists $x$ such that $x$ is present in only one of the $R_1$ or $R_2$, or there exist $x, y$ such that $x, y \in m_{i,j}$ but $x \in \mu_{i',j'}$ and $\mu_{i',j''}$, where $j' \neq j''$ or there exist $x, y$ such that $x, y \in m_{i,j}$ but $x \in \mu_{i',j'}$ and $\mu_{i'',j''}$, where $i' \neq i''$. We now analyse these three cases in detail.

**Case 1.** There exists $x$ such that $x$ is present in only one of the $R_1$ and $R_2$; without loss of generality, $x \in m_{1,1}$. Then,

$$\frac{\partial f}{\partial x} + O(\epsilon) = \alpha_{1,1}\frac{m_{1,1}}{x}p_2 \cdots p_k.$$

The right hand side must be in $\mathbb{F}[\epsilon][\mathbf{x}]$. Thus $\frac{\partial f}{\partial x} \in \overline{\text{ROF}_0}$; from Claim 2.3 it is in $\text{ROF}_0$. Observation 2.2 and $f(\mathcal{G} + \mathcal{G}_2^{SV}) \equiv 0$ imply that $\frac{\partial f}{\partial x}(\mathcal{G}) \equiv 0$. As $\mathcal{G}$ is a hitting set generator for $\Sigma_2\text{ROF}$, this means that $\frac{\partial f}{\partial x} \equiv 0$ and

$$\alpha_{1,1}m_{1,1}p_2 \cdots p_k = O(\epsilon),$$

So $f + O(\epsilon) = p_1' \cdot p_2 \cdots p_k + q_1 \cdots q_\ell$, where $p_1' = \alpha_{i,2}m_{i,2} + \cdots + \alpha_{i,k_i}m_{i,k_i}$. This has fewer than $n$ variables. So the claim follows from the induction hypothesis.[1]

**Case 2.** There exist $x, y$ such that $x, y \in m_{i,j}$ but $x \in \mu_{i',j'}$ and $\mu_{i',j''}$, where $j' \neq j''$. Without loss of generality $i = 1, j = 1, i' = 1, j' = 1, j'' = 2$. Then,

$$\frac{\partial f}{\partial x}(y = 0) + O(\epsilon) = \beta_{1,1}\frac{\mu_{1,1}}{x} \cdot q_2 \cdots q_\ell$$
$$\frac{\partial f}{\partial y}(x = 0) + O(\epsilon) = \beta_{1,2}\frac{\mu_{1,2}}{y} \cdot q_2 \cdots q_\ell$$

[1] By removing $m_{1,1}$ from $\{m_{i,j}\}_{i,j}$ since $f$ has an expression that does not contain $m_{1,1}$.

$$\frac{\partial^2 f}{\partial x \partial y} + O(\epsilon) = \alpha_{1,1} \frac{m_{1,1}}{xy} \cdot p_2 \cdots p_k.$$

The right hand sides of all the three expressions are in $\mathbb{F}[\epsilon][\mathbf{x}]$. So $\frac{\partial f}{\partial x}(y = 0), \frac{\partial f}{\partial y}(x = 0), \frac{\partial^2 f}{\partial x \partial y} \in \overline{\text{ROF}_0}$. Thus from Claim 2.3, they are in $\text{ROF}_0$. Also, from Observations 2.2 and 2.3, $f(\mathcal{G} + \mathcal{G}_2^{SV}) \equiv 0$ implies that $\left(\frac{\partial f}{\partial x}(y = 0)\right)(\mathcal{G}) \equiv \left(\frac{\partial f}{\partial y}(x = 0)\right)(\mathcal{G}) \equiv \left(\frac{\partial^2 f}{\partial x \partial y}\right)(\mathcal{G}) \equiv 0$. However, as $\mathcal{G}$ is a hitting set generator for $\Sigma_2\text{ROF}$, this means that

$$\begin{aligned}
\beta_{1,1}\mu_{1,1} \cdot q_2 \cdots q_\ell &= O(\epsilon) \\
\beta_{1,2}\mu_{1,2} \cdot q_2 \cdots q_\ell &= O(\epsilon) \\
\alpha_{1,1}m_{1,1} \cdot p_2 \cdots p_k &= O(\epsilon).
\end{aligned}$$

Hence, $f + O(\epsilon) = p_1' p_2 \cdots p_k + q_1' q_2 \cdots q_\ell$, where $p_1' = \alpha_{i,2}m_{i,2} + \cdots + \alpha_{i,k_i}m_{i,k_i}$ and $q_1' = \beta_{i,3}\mu_{i,3} + \cdots + \alpha_{i,\ell_i}\mu_{i,\ell_i}$. Thus $f$ has fewer than $n$ variables and the claim follows from the induction hypothesis.

**Case 3.** There exist $x, y$ such that $x, y \in m_{i,j}$ but $x \in \mu_{i',j'}$ and $y \in \mu_{i'',j''}$, where $i' \neq i''$. Without loss of generality, $i = 1, j = 1$, $i' = 1, j' = 1$, and $i'' = 2, j'' = 1$. Then, for $q_1' := q_1 - \beta_{1,1}\mu_{1,1}$ and $q_2' := q_2 - \beta_{2,1}\mu_{2,1}$,

$$\begin{aligned}
\frac{\partial f}{\partial x}(y = 0) + O(\epsilon) &= \beta_{1,1} \cdot \frac{\mu_{1,1}}{x} \cdot q_2' \cdot q_3 \cdots q_\ell \\
\frac{\partial f}{\partial y}(x = 0) + O(\epsilon) &= q_1' \cdot \beta_{2,1} \cdot \frac{\mu_{2,1}}{y} \cdot q_3 \cdots q_\ell.
\end{aligned}$$

The right hand sides of both of the above expressions are in $\mathbb{F}[\epsilon][\mathbf{x}]$. Hence $\frac{\partial f}{\partial x}(y = 0), \frac{\partial f}{\partial y}(x = 0) \in \overline{\text{ROF}_0}$. Thus from Claim 2.3 they are in $\text{ROF}_0$. Observations 2.2 and 2.3 give that $f(\mathcal{G} + \mathcal{G}_2^{SV}) \equiv 0$ implies that $\left(\frac{\partial f}{\partial x}(y = 0)\right)(\mathcal{G}) \equiv \left(\frac{\partial f}{\partial y}(x = 0)\right)(\mathcal{G}) \equiv 0$. As $\mathcal{G}$ is a hitting set generator for $\Sigma_2\text{ROF}$, this means that

$$\begin{aligned}
\beta_{1,1} \cdot \mu_{1,1} \cdot q_2' \cdot q_3 \cdots q_\ell &= O(\epsilon) \\
q_1' \cdot \beta_{2,1} \cdot \mu_{2,1} \cdot q_3 \cdots q_\ell &= O(\epsilon).
\end{aligned}$$

It is not difficult to see that this means that at least one of $\beta_{1,1}\mu_{1,1} \cdot \beta_{2,1}\mu_{2,1} \cdot q_3 \cdots q_\ell$ and $q_1' \cdot q_2' \cdot q_3 \cdots q_\ell$ is also $O(\epsilon)$.

Case 3.1. $\beta_{1,1}\mu_{1,1} \cdot \beta_{2,1}\mu_{2,1} \cdot q_3 \cdots q_\ell = O(\epsilon)$. Then

$$\frac{\partial^2 f}{\partial x \partial y} + O(\epsilon) = \alpha_{1,1}\frac{m_{1,1}}{x,y}p_2 \cdots p_k.$$

The right hand side is in $\mathbb{F}[\epsilon][\mathbf{x}]$. Thus $\frac{\partial^2 f}{\partial x \partial y} \in \overline{\text{ROF}_0}$ and from Claim 2.3 it is in $\text{ROF}_0$. Observation 2.2 gives that $f(\mathcal{G} + \mathcal{G}_2^{SV}) \equiv 0$ implies that $\left(\frac{\partial^2 f}{\partial x \partial y}\right)(\mathcal{G}) \equiv 0$. As $\mathcal{G}$ is a hitting set generator for $\Sigma_2\text{ROF}$, this means that

$$\alpha_{1,1}m_{1,1}p_2 \cdots p_k = O(\epsilon),$$

So $f + O(\epsilon) = p_1' \cdot p_2 \cdots p_k + q_1' \cdot q_2' \cdot q_3 \cdots q_\ell$, where $p_{1,1}' = p_{1,1} - \alpha_{1,1}m_{1,1}$. This has fewer than $n$ variables. So the claim follows from the induction hypothesis.

Case 3.2. $q_1'q_2' \cdot q_3 \cdots q_\ell = O(\epsilon)$. Then for $p_1' = p_1 - \alpha_{1,1}m_{1,1}$

$$f(x = 0, y = 0) + O(\epsilon) = p_1'p_2 \cdots p_k.$$

The right hand side is in $\mathbb{F}[\epsilon][\mathbf{x}]$. So $f(x = 0, y = 0) \in \overline{\text{ROF}_0}$ and from Claim 2.3 it is in $\text{ROF}_0$. Observation 2.3 gives that $f(\mathcal{G} + \mathcal{G}_2^{SV}) \equiv 0$ implies that $f(x = 0, y = 0) \equiv 0$. As $\mathcal{G}$ is a hitting set generator for $\Sigma_2\text{ROF}$, this means that

$$p_1'p_2 \cdots p_k = O(\epsilon).$$

So $f + O(\epsilon) = \alpha_{1,1}m_{1,1} \cdot p_2 \cdots p_k + \beta_{1,1}\mu_{1,1} \cdot \beta_{1,2}\mu_{2,1} \cdot q_3 \cdots q_\ell$. Note that variables in $\text{var}(q_1')$ and $\text{var}(q_2')$ (both of which can not be empty simultaneously) are not present in the second summand. Thus, $f$ is in Case 1 and the claim follows because it holds in Case 1. □

We now need to show that when $\{m_{i,j} : i \in [k], j \in k_i\} = \{\mu_{i,j} : i \in [\ell], j \in \ell_i\}$, $f(\mathcal{G} + \mathcal{G}_2^{SV}) \equiv 0$ implies $f \equiv 0$. We start by proving the following claim.

**Claim 8.3** *$k = \ell$ and $p_i = \gamma_i q_i$, where $\gamma_i \neq 0 \in \mathbb{F}[\epsilon]$ for all $i \in [k]$.*

**Proof:** By induction on $n$ the number of variables in $f$. If $n = 1$, then $f + O(\epsilon) = \alpha x_1 + \beta x_2$ and the claim is true. So suppose that it is true for all $f'$ in the border of sum of 2 additive constant free depth-4 ROFs with at most $n - 1$ variables.

Without loss of generality for all $i \in [k]$, $\text{val}(\alpha_{i,1}) \leq \text{val}(\alpha_{i,j})$ for all $j \in [k_i]$ and for all $i \in [m]$, $\text{val}(\beta_{i,1}) \leq \text{val}(\beta_{i,j})$ for all $j \in [\ell_i]$. Also, without loss of generality, $m_{1,1} = \mu_{1,j}$ for some

$j \in [\ell_1]$. Suppose that some $\mu_{1,j'}$ is not in $p_1$; without loss of generality $\mu_{1,j'} = m_{2,j''}$. Then observe that $\alpha_{1,1} m_{1,1} \cdot \alpha_{2,j''} m_{2,j''} \cdot p_3 \cdots p_k \in \mathbb{F}[\epsilon][\mathbf{x}]$. But this means that $\mathsf{val}(\alpha_{1,1}) + \mathsf{val}(\alpha_{2,j''}) + \mathsf{val}(p_3 \cdots p_k) \geq 0$. As $\mathsf{val}(\alpha_{1,1}) \leq \mathsf{val}(\alpha_{1,j})$ for all $j \in [k_1]$, this means $\mathsf{val}(p_1) + \mathsf{val}(\alpha_{2,j''}) + \mathsf{val}(p_3 \cdots p_k) \geq 0$. So, $\alpha_{2,j''} m_{2,j''} p_1 \cdot p_3 \cdots p_k \in \mathbb{F}[\epsilon][\mathbf{x}]$. But then so is $\beta_{1,j'} \mu_{1,j'} q_2 \cdots q_\ell$. Thus

$$\left(\alpha_{2,j''} m_{2,j''} p_1 \cdot p_3 \cdots p_k + \beta_{1,j'} \mu_{1,j'} q_2 \cdots q_\ell\right) (\epsilon = 0)$$

is in $\sum_2 \mathrm{ROF}_0$. Thus for all $x \in \mu_{1,j'} = m_{2,j''}$, $\frac{\partial f}{\partial x} + O(\epsilon) \in \sum_2 \mathrm{ROF}$. From Observation 2.2, $f(\mathcal{G} + \mathcal{G}_2^{SV}) \equiv 0$ implies that

$$\alpha_{2,j''} m_{2,j''} p_1 \cdot p_3 \cdots p_k + \beta_{1,j'} \mu_{1,j'} q_2 \cdots q_\ell = O(\epsilon)$$

Thus, $f + O(\epsilon) = p_1 \cdot p_2' \cdot p_3 \cdots p_k + q_1' \cdot q_2 \cdots q_\ell$, where $p_2' = p_2 - \alpha_{2,j''} m_{2,j''}$ and $q_2' = q_2 - \beta_{1,j'} \mu_{1,j'}$. So $f$ has fewer than $n$ variables and the claim follows from the induction hypothesis. □

Now for all $i \in [k]$, it is easy to see that $p_i$ and $q_i$ have multilinear ROABPs with the same variable order over $\mathbb{F}(\epsilon)$. Multiplying all of these ROABPs together we get multilinear ROABPs for $R_1$ and $R_2$ with the same variable order. Hence $f \in \overline{ROABP}$. Claim 2.2 implies that $f \in ROABP$. As $\mathcal{G}$ is a hitting set generator for ROABPs, $f(\mathcal{G} + \mathcal{G}_2^{SV}) \equiv 0$ along with the fact that $\mathcal{G}_2^{SV}$ contains 0 in its image implies that $f \equiv 0$. □

**Proof of Theorem 1.12, the depth-4 case**

Let $f \not\equiv 0$ be in the border of sums of 2 additive constant free, depth-4 ROFs. Then from Lemma 8.3, we have $f(\mathcal{G} + \mathcal{G}_3^{SV}) \not\equiv 0$. From Theorem 2.1 $\mathcal{G}$ is a polynomial map in $O(\log n)$ variables and of degree $\mathrm{poly}(n)$. $\mathcal{G}_3^{SV}$ is a polynomial in 6 variables and of degree $O(n)$. Hence, $\mathcal{G} + \mathcal{G}_3^{SV}$ is a polynomial map in $O(\log n)$ variables and of degree $\mathrm{poly}(n)$. Thus, there is a hitting set for $f$ that can be computed in time $n^{O(\log n)}$.

## 8.5 PIT for the border of sum of 2 depth-5 ROFs

In this section, we prove Theorem 1.12. The proof of the theorem follows in a straightforward manner from the following lemma.

**Lemma 8.4** *Let $\mathcal{G} : \mathbb{F}^m \to \mathbb{F}^n$ be a hitting set generator for $n$-variate multilinear ROABPs as well as $\sum_2$ ROF. Then $\mathcal{G} + \mathcal{G}_5^{SV}$ is a hitting set generator for the border of sums of 2 homogeneous depth-5*

*ROFs provided that* $|\mathbb{F}| > n^2 d$.

**Proof:** Let $f + O(\epsilon) = R_1 + R_2$, where $R_1, R_2$ are homogeneous depth-5 ROFs over $\mathbb{F}(\epsilon)$ and $f \not\equiv 0 \in \mathbb{F}[\mathbf{x}]$. Similarly to the proof of Lemma 8.3, it suffices to show that $\mathcal{G} + \mathcal{G}_4^{SV}$ is a hitting set generator when $R_1$ and $R_2$ are depth 4 ROFs with $\times$ gate at the top. Let $R_1 = p_1 \cdots p_r$ and $R_2 = q_1 \cdots q_s$, where for all $i \in [r]$, $p_i = \sum_{j \in [r_i]} \ell_{i,j,1} \cdots \ell_{i,j,d_i}$ and for all $i \in [s]$, $q_i = \sum_{j \in [s_i]} \ell'_{i,j,1} \cdots \ell'_{i,j,e_i}$.

**Claim 8.4** *If* $f(\mathcal{G} + \mathcal{G}_4^{SV}) \equiv 0$, *then for all* $i, j, k$, *there exist* $i', j', k'$ *such that* $\ell_{i,j,k} = \alpha_{i,j,k} \ell'_{i',j',k'}$ *for some* $\alpha_{i,j,k} \neq 0 \in \mathbb{F}(\epsilon)$.

**Proof:** Let $R_1, R_2$ be as above. We first argue that if $x$ is present in $R_1$, then it must also be present in $R_2$ and vice versa. Suppose not; without loss of generality $x$ is in $R_1$ but not in $R_2$. Then $\frac{\partial f}{\partial x} + O(\epsilon) = \frac{\partial R_1}{\partial x}$. Thus $f \in \overline{\text{ROF}_0}$[1] $= \text{ROF}_0$. $\frac{\partial f}{\partial x}(\mathcal{G} + \mathcal{G}_3^{SV}) \equiv 0$, for otherwise Observation 2.2 would imply that $f(\mathcal{G} + \mathcal{G}_4^{SV}) \not\equiv 0$. As $\mathcal{G}$ is a hitting set generator for $\sum_2 \text{ROF}$ and $\mathcal{G}_3^{SV}$ has 0 in its image, this means that $\frac{\partial f}{\partial x} \equiv 0$ and thus $x$ can be removed from $R_1$.

We now show how to ensure that for all $i, j, k$ a multiple of $\ell_{i,j,k}$ appears in some linear form in $R_2$. Then repeating this same argument but for showing that a multiple of $\ell'_{i',j',k'}$ appears in some linear form of $R_1$ for all $i', j', k'$ would prove the claim. For ease of exposition, fix $i = j = k = 1$. Let $x$ be the variable in $\ell_{1,1,1}$ whose coefficient has the smallest valuation among all the coefficients in $\ell_{1,1,1}$ and without loss of generality $x$ is in $\ell'_{1,1,1}$. Let the coefficients of $x$ in both these linear forms be $\alpha$ and $\alpha'$ respectively. Let $y$ be any other variable in $\ell_{1,1,1}$ and its coefficient be $\beta$. We will now show how to ensure that $\alpha' x + \frac{\alpha'}{\alpha} \cdot \beta y$ occurs in $\ell'_{1,1,1}$. Repeating this argument for all variables in $\ell_{1,1,1}$ would ensure that a multiple of $\ell_{1,1,1}$ is present in $\ell'_{1,1,1}$. There are two cases.

**Case 2.** $\text{fca}(x, y)$ in $R_2$ is a $+$ gate. Define $f' := f\left(x = x - \frac{\beta}{\alpha}(\epsilon = 0)y\right)$. This case has two sub-cases.

Case 2.1. $y \in \ell'_{1,1,1}$. Let $\alpha' x + \beta' y$ be present in $\ell'_{1,1,1}$. Observe that $f' + O(\epsilon) = R'_1 + R'_2$, where $R'_1$ is obtained from $R_1$ by replacing $\alpha x + \beta y$ in $\ell_{1,1,1}$ by $\alpha x$ and replacing $\alpha' x + \beta' y$ in $R'_2$ by $\alpha' x + \gamma y$, where $\gamma = \beta' - \alpha' \cdot \frac{\beta}{\alpha}$. This is so because $f' = f\left(x = x - \frac{\beta}{\alpha} y + O(\epsilon)\right) = f\left(x = x - \frac{\beta}{\alpha} y\right) + O(\epsilon) = R'_1 + R'_2$. From Observation 2.3, $f'(\mathcal{G} + \mathcal{G}_3^{SV}) \equiv 0$. Now, as only $R'_2$ contains $y$,

$$\frac{\partial f'}{\partial y} + O(\epsilon) = \gamma \cdot \prod_{k=2}^{e_1} \ell'_{1,1,k} \cdot q_2 \cdots q_\ell.$$

[1] Note that every homogeneous ROF is also an additive constant free ROF

Thus $\frac{\partial f'}{\partial y} \in \overline{\text{ROF}_0} = \text{ROF}_0$. Observation 2.2 implies that $\frac{\partial f'}{\partial y}(\mathcal{G} + \mathcal{G}_2^{SV}) \equiv 0$. As $\mathcal{G}$ is a hitting set generator for ROF, this means that $\frac{\partial f'}{\partial y} \equiv 0$. Thus $\gamma \cdot \prod_{k=2}^{e_1} \ell'_{1,1,k} \cdot q_2 \cdots q_\ell = O(\epsilon)$ and $f' + O(\epsilon) = R'_1 + R''_2$, where $R''_2$ is obtained from $R_2$ by replacing $\alpha' x + \beta' y$ by just $\alpha' x$. As $f = f'\left(x = x + \frac{\beta}{\alpha} y\right) + O(\epsilon)$ this means that $f + O(\epsilon) = R_1 + R'''_2$ where $R'''_2$ is obtained from $R_2$ by replacing $\beta' y$ with $\frac{\alpha'}{\alpha} \cdot \beta$.

Case 2.2. $y \notin \ell'_{1,1,1}$. Without loss of generality $y \in \ell'_{1,2,1}$. Observe that $f' + O(\epsilon) = R'_1 + R'_2$, where $R'_i := R_i(x = x - \frac{\beta}{\alpha} y)$. Now, if the coefficient of $y$ in $\ell'_{1,2,1}$ is $\beta'$, then

$$\frac{\partial f'}{\partial y} + O(\epsilon) = -\alpha' \cdot \frac{\beta}{\alpha} \prod_{k=2}^{e_1} \ell'_{1,1,k} \cdot q_2 \cdots q_s + \beta' \prod_{k=2}^{e_1} \ell'_{1,2,k} \cdot q_2 \cdots q_s.$$

Observe that there can be no cancellation between the two summands. This is so because $e_1 \geq 2$. Thus, every monomial of the first summand contains a variable in $\text{var}(\ell'_{1,2,k})$ which can not be present in the other summand (and vice versa). So, both summands are in $\mathbb{F}[\epsilon]$ and $\frac{\partial f'}{\partial y} \in \sum_2 \text{ROF}_0$. As in Case 2.1, $f'(\mathcal{G} + \mathcal{G}_3^{SV}) \equiv 0$ and thus from Observation 2.2, $\frac{\partial f'}{\partial y}(\mathcal{G} + \mathcal{G}_2^{SV}) \equiv 0$. Since $\mathcal{G}$ is a hitting set generator for $\sum_2 \text{ROF}$, this means that $\frac{\partial f'}{\partial y} \equiv 0$. Hence $f'$ does not have $y$ and $f' + O(\epsilon) = R'_1 + R''_2$ where $R''_2 = R'_2(y = 0) = R_2(y = 0)$. Hence $f + O(\epsilon) = R_1 + R'''_2$, where $R'''_2 = R''_2(x = x + \frac{\beta}{\alpha'} y)$ is an ROF which can be obtained by first setting $y = 0$ in $R_2$ and then replacing $\alpha' x$ by $\alpha' x + \frac{\alpha'}{\alpha} \cdot \beta y$.

**Case 3.** $\text{fca}(x, y)$ in $R_2$ is a $\times$ gate. Define $f' := f\left(x = x - \frac{\beta}{\alpha}(\epsilon = 0) y\right)$. There are two subcases.

Case 3.1. There exists $k \in [e_1]$ such that $y \in \ell'_{1,1,k}$. Without loss of generality, $k = 2$. Observe that $f' + O(\epsilon) = R'_1 + R'_2$, where $R'_i = R_i(x = x - \frac{\beta}{\alpha} y)$. Now, if the coefficient of $y$ in $\ell'_{1,1,2}$ is $\beta'$, then

$$\frac{\partial^2 f'}{\partial y^2} + O(\epsilon) = -2 \cdot \alpha' \cdot \frac{\beta}{\alpha} \cdot \beta' \prod_{k=3}^{e_1} \ell'_{1,1,k} \cdot q_2 \cdots q_s$$

$$\frac{\partial f'}{\partial y}(y = 0) + O(\epsilon) = -\alpha' \cdot \frac{\beta}{\alpha} \cdot \ell'_{1,1,2}(y = 0) \cdot \prod_{k=3}^{e_1} \ell'_{1,1,k} \cdot q_2 \cdots q_s + \ell'_{1,1,1} \cdot \beta' \prod_{k=2}^{e_1} \ell'_{1,2,k} \cdot q_2 \cdots q_s.$$

Thus, $\frac{\partial^2 f'}{\partial y^2} \in \overline{\text{ROF}_0} = \text{ROF}_0$. As in Case 2.1, $f'(\mathcal{G} + \mathcal{G}_3^{SV}) \equiv 0$ and thus from Observation 2.2, $\frac{\partial^2 f'}{\partial y^2}(\mathcal{G} + \mathcal{G}_2) \equiv 0$. Since $\mathcal{G}$ is a hitting set generator for $\sum_2 \text{ROF}$, this means that $\frac{\partial^2 f'}{\partial y^2} \equiv 0$.

Also, unless $\ell'_{1,1,2}(y=0) \equiv 0$, every monomial in $-\alpha' \cdot \frac{\beta}{\alpha} \cdot \ell'_{1,1,2}(y=0) \cdot \prod_{k=3}^{e_1} \ell'_{1,1,k} \cdot q_2 \cdots q_s$ must contain a variable in $\text{var}\left(\ell'_{1,1,2}\right) \setminus \{y\}$ which can not be present in the other summand. Hence there can be no cancellations between the two summands that make up $\frac{\partial f'}{\partial y}(y=0) + O(\epsilon)$ and both of them must be in $\mathbb{F}[\epsilon]$. Thus $\frac{\partial f'}{\partial y}(y=0) \in \sum_2 \text{ROF}_0$. Observations 2.2 and 2.3 imply that $\left(\frac{\partial f'}{\partial y}(y=0)\right)(\mathcal{G} + \mathcal{G}_1^{SV}) \equiv 0$. Since $\mathcal{G}$ is a hitting set generator for $\sum_2 \text{ROF}$, this means that $\frac{\partial f'}{\partial y}(y=0) \equiv 0$. As $\frac{\partial^2 f'}{\partial y^2}$ is also 0 and $\text{char}(\mathbb{F}) \neq 2$, $f'$ does not contain $y$ and $f' + O(\epsilon) = R'_1 + R''_2$ where $R''_2 = R'_2(y=0) = R_2(y=0)$. Hence $f + O(\epsilon) = R_1 + R'''_2$, where $R'''_2 = R''_2(x = x + \frac{\beta}{\alpha'}y)$ is an ROF which can be obtained by first setting $y=0$ in $R_2$ and then replacing $\alpha' x$ by $\alpha' x + \frac{\alpha'}{\alpha} \cdot \beta y$.

Case 3.2. There exists $i \in [s], j \in [s_i], k \in [e_i]$ such that $y \in \ell'_{i,j,k}$. Without loss of generality, $i=2, j=1, k=1$. Observe that $f' + O(\epsilon) = R'_1 + R'_2$, where $R'_i = R_i(x = x - \frac{\beta}{\alpha}y)$. Now, if the coefficient of $y$ in $\ell'_{2,1,1}$ is $\beta'$, then

$$\frac{\partial^2 f'}{\partial y^2} + O(\epsilon) = 2 \cdot \left(-\alpha' \cdot \frac{\beta}{\alpha} \prod_{k=2}^{e_1} \ell'_{1,1,k}\right) \cdot \left(\beta' \prod_{k=2}^{e_2} \ell'_{2,1,k}\right) \cdot q_3 \cdots q_s$$

$$\frac{\partial f'}{\partial y}(y=0) + O(\epsilon) = \left(-\alpha' \cdot \frac{\beta}{\alpha} \prod_{k=2}^{e_1} \ell'_{1,1,k}\right) \cdot q_2(y=0) \cdot q_3 \cdots q_s + q_1 \cdot \left(\beta' \prod_{k=2}^{e_2} \ell'_{2,1,k}\right) \cdot q_3 \cdots q_s.$$

Just like in Case 3.1, $\frac{\partial^2 f'}{\partial y^2} \equiv 0$. Also, unless $q_2(y=0) \equiv 0$, every monomial in $\left(-\alpha' \cdot \frac{\beta}{\alpha} \prod_{k=2}^{e_1} \ell'_{1,1,k}\right) \cdot q_2(y=0) \cdot q_3 \cdots q_s$ must contain a variable in $\text{var}(q_2) \setminus \{y\}$ which can not be present in the other summand. Hence there can be no cancellations between the two summands that make up $\frac{\partial f'}{\partial y}(y=0) + O(\epsilon)$ and both of them must be in $\mathbb{F}[\epsilon]$. Thus $\frac{\partial f'}{\partial y}(y=0) \in \sum_2 \text{ROF}_0$. Observations 2.2 and 2.3 imply that $\left(\frac{\partial f'}{\partial y}(y=0)\right)(\mathcal{G} + \mathcal{G}_1^{SV}) \equiv 0$. Since $\mathcal{G}$ is a hitting set generator for $\sum_2 \text{ROF}$, this means that $\frac{\partial f'}{\partial y}(y=0) \equiv 0$. As $\frac{\partial^2 f'}{\partial y^2}$ is also 0 and $\text{char}(\mathbb{F}) \neq 2$, $f'$ does not contain $y$ and we can again argue as in Cases 2.2 and 3.1 that $f + O(\epsilon) = R_1 + R'''_2$, where $R'''_2 = R''_2(x = x + \frac{\beta}{\alpha'}y)$ is an ROF which can be obtained by first setting $y=0$ in $R_2$ and then replacing $\alpha' x$ by $\alpha' x + \frac{\alpha'}{\alpha} \cdot \beta y$. □

Notice that because of the above claim, if $f(\mathcal{G} + \mathcal{G}_4^{SV}) \equiv 0$, then $f$ is in the *orbit* of the border of a sum of two homogeneous ROFs of depth 4. So we can expect a claim analogous to Claim 8.2 to hold for $f$. We now prove that this is indeed the case.

**Claim 8.5** *If* $f(\mathcal{G} + \mathcal{G}_4^{SV}) \equiv 0$, *then* $\left\{\prod_{k\in[d_i]} \ell_{i,j,k} : i \in [r], j \in r_i\right\} = \left\{\alpha_{i,j} \prod_{k\in[e_i]} \ell'_{i,j,k} : i \in [s], j \in s_i\right\}$

*for some* $\alpha_{i,j} \neq 0 \in \mathbb{F}(\epsilon)$.

**Proof:** The proof is similar to that of Claim 8.2. In the proof of that claim, we analyse various cases involving two variables $x, y$. In our case, we shall have to analyse the same cases, but with $x$ and $y$ replaced by $\ell_{i,j,k}$ and $\ell'_{i',j',k'}$. Let $\ell_{i,j,k} = \alpha x + h$ and $\ell'_{i',j',k'} = \beta y + h'$, where $h, h'$ are linear forms and $x, y$ are chosen such that $\mathsf{val}(\alpha) \leq \mathsf{val}(h)$ and $\mathsf{val}(\beta) \leq \mathsf{val}(h')$. Define $f' := f\left(x = x - \frac{h}{\alpha}(\epsilon = 0),\ y = y - \frac{h'}{\beta}(\epsilon = 0)\right)$. Notice that $f' + O(\epsilon) := f\left(x = x - \frac{h}{\alpha},\ y = y - \frac{h'}{\beta}\right)$. This can be obtained from $f$ by replacing $\ell_{i,j,k}$ by $\alpha x$ and $\ell'_{i',j',k'}$ by $\beta y$. Due to Claim 8.4, $f'$ is also in the border of sum of two homogeneous ROFs of depth 5. Observation 2.3 implies that $f(\mathcal{G} + \mathcal{G}_2^{SV}) \equiv 0$. It can be verified that the argument in the proof of Claim 8.2 would go through even if $m_{i,j_{i,j}}$ and $\mu_{i,j_{i,j}}$ are products of linear forms instead of being monomials; it only requires that we take derivatives with respect to variables and set variables to 0. So, the claim can be proved by arguing analogously. □

We now prove a claim analogous to Claim 8.3.

**Claim 8.6** *$s = r$ and $p_i = q_i$ for all $i \in [k]$.*

**Proof:** Again the proof is same as the proof of Claim 8.3; all we need to do is observe that that proof does not use the fact that $\{m_{i,j}\}_{i,j}$ and $\{\mu_{i,j}\}_{i,j}$ are monomials. □

Much like in the proof of Lemma 8.3, $p_i$ and $q_i$ have multilinear ROABPs with the same variable order. Multiplying all of these ROABPs together we get multilinear ROABPs for $R_1$ and $R_2$ with the same variable order. Hence $f \in \overline{ROABP}$. Claim 2.2 implies that $f \in ROABP$. As $\mathcal{G}$ is a hitting set generator for ROABPs, $f(\mathcal{G} + \mathcal{G}_5^{SV}) \equiv 0$ along with the fact that $\mathcal{G}_5^{SV}$ contains 0 in its image implies that $f \equiv 0$. □

**Proof of Theorem 1.12**

Let $f \not\equiv 0$ be in the border of sums of 2 homogeneous depth-5 ROFs. Then from Lemma 8.4, we have $f(\mathcal{G} + \mathcal{G}_5^{SV}) \not\equiv 0$. From Theorem 2.1, $\mathcal{G}$ is a polynomial map in $O(\log n)$ variables and of degree $\mathrm{poly}(n)$. $\mathcal{G}_5^{SV}$ is a polynomial in 10 variables and of degree $\mathrm{poly}(n)$. Hence, $\mathcal{G} + \mathcal{G}_5^{SV}$ is a polynomial map in $O(\log n)$ variables and of degree $O(n)$. Thus, there is a hitting set for $f$ that can be computed in time $n^{O(\log n)}$.

# Chapter 9

# Directions for future work

In this thesis, we described some results about hitting sets for orbits of various circuit classes, gave an equivalence test for ROFs, de-bordered the border of sums of two ROFs, and gave an alternative, and in our opinion, a more direct proof of a super-polynomial lower bound for constant depth arithmetic circuits. We now mention some directions of future work to extend these results.

## Hitting sets for orbits of circuit classes

In Chapters 3 and 4, we have studied the hitting set problem for the orbits of several important polynomial families and circuit classes that are not closed under affine projections. This line of research is both natural and interesting as affine projections of some of these circuit classes and polynomial families capture much larger circuit classes (in some cases, almost the entire class of VP circuits). The orbit of a polynomial $f$ is a natural and "dense" subset of affine projections of $f$ that, in turn, resides in the orbit closure of $f$. We have shown efficient hitting set constructions for the orbits of several well-studied circuit classes. Despite the progress made here, there are several natural questions that, if resolved, will strengthen and complete the set of results presented in this work. We leave these for future work:

1. **Removing the low individual degree restriction.** Theorems 1.1 and 1.3 give hitting sets for orbits of low-individual degree commutative ROABPs and constant width multilinear ROABPs. While the low individual degree restriction is natural as it subsumes the multilinear case, it would be ideal if we get rid of this limitation of our results. A subsequent work by Bhargava and Ghosh [BG21] has made progress in this direction in the small width setting. However, the problem of constructing efficient hitting-sets for the orbits of general commutative ROABPs remains open.

2. **Lower bound and hitting set for the orbits of ROABPs.** We would also like to remove the requirements of commutativity and constant-width from Theorems 1.1 and 1.3 on hitting sets for the orbits of ROABPs. It is worth noting that an explicit hitting set for the orbits of ROABPs implies a lower bound for the same model computing some explicit polynomial due to a result by Agrawal [Agr05]. To our knowledge, no explicit lower bound is known for the orbits of ROABPs. Can we prove such a lower bound first?

3. **Hitting sets for the orbits of** Det **and** IMM**.** The determinant (Det) and the iterated matrix multiplication (IMM) polynomial families are complete for the class of algebraic branching programs under $p$-projections. Can we design efficiently constructible hitting sets for the orbits of Det and IMM? Observe that a hitting set for the orbits of *multilinear* ROABPs is a hitting set for orb(IMM). Also, a hitting set for the orbits of the polynomials computable by the Edmonds' model is a hitting set for the orbits of both Det and IMM.

## Equivalence test for ROFs

In Chapter 5, we gave the first randomized polynomial-time equivalence test for ROFs (Theorem 1.6) and use this result to solve PE for orbits of (slightly restricted) ROFs (Theorem 1.7). These results are substantial generalizations of two well-studied problems in algebraic complexity, namely quadratic form equivalence and reconstruction of ROFs. As PE is graph isomorphism hard for even cubic forms, it is indeed satisfying to know that PE can be solved efficiently for an unbounded-depth, unbounded-degree, and unbounded-fanin circuit class such as orbits of ROFs. Theorem 1.6 also implies efficient learning of random arithmetic formulas (without any restriction on the underlying tree structure) in the high number of variables setting. We now note a few future directions that can be pursued:

1. **Generalizing our results.** An interesting generalization of Theorem 1.6 would be an equivalence test for power-substituted ROFs and, more generally, for *univariate-substituted* ROFs [1]. An equivalence test for univariate-substituted ROFs would greatly generalize the equivalence test for the sums of univariates model studied in [GKP18] and the reconstruction algorithm for preprocessed ROFs in [SV14]. We believe that our equivalence test and its analysis can be extended to work for univariate-substituted

[1] A univariate-substituted ROF is obtained from an ROF by substituting every variable $x_i$ by an arbitrary (and unknown) univariate polynomial $g_i(x_i)$. Such ROFs were called *preprocessed* ROFs in [SV14].

ROFs. To support this belief, let us consider the power-substituted sum-product polynomial $\mathsf{SPP} := \sum_{i\in[s]} \prod_{j\in[d]} x_{i,j}^{e_{i,j}}$, where $e_{i,j} \in \mathbb{N}$. It turns out that $\det(H_{\mathsf{SPP}})$ factorizes as:

$$\det(H_{\mathsf{SPP}}) = (-1)^{s(d-1)} \cdot \prod_{i\in[s],j\in[d]} e_{i,j} \cdot \prod_{i\in[s]} (e_{i,1} + \ldots + e_{i,d} - 1) \cdot \prod_{i\in[s],j\in[d]} x_{i,j}^{e_{i,j}\cdot d-2}.$$

So the equivalence test for SP, described in Section 5.2.1, works (almost as it is) for SPP.

2. **Learning orbits of sparse polynomials and ROABPs.** Studying the orbit of a circuit class is a natural first step towards understanding affine projections of the class. Efficient proper learning algorithms are long known for sparse polynomials [KS01] and ROABPs [BBB+00, KS06]. Recall that affine projections of these classes capture immensely powerful circuit classes such as depth-3 circuits and ABPs. Like ROFs, can we design efficient learning algorithms for *orbits* of sparse polynomials and ROABPs? [BDS24] have shown that ET for sparse polynomials is NP-hard. Very recently, [BDGT24] show that the isomorphism testing problem of ROABPs is also NP-hard. In the isomorphism testing problem $A$ is a permutation matrix.

3. **Learning random formulas.** Theorem 1.6 solves the learning problem for random formulas when the number of variables $n$ is larger than the size $s$ of the underlying tree of the formula. A more interesting setting of parameters is $s = \text{poly}(n)$. Can we design an efficient learning algorithm for random formulas (of even constant depth) if $s \gg n$?

## Lower bounds for constant depth arithmetic circuits

Recently, [LST21] made remarkable progress on arithmetic circuit lower bounds by giving the first super-polynomial lower bound for low-depth formulas. In Chapter 7, we give an alternative and in our opinion a more direct proof of their result. Unlike [LST21] however, we are able to bypass the set-multilinearisation step. Since this step incurs a loss of a factor of $d^{O(d)}$, it is not clear if proving exponential lower bounds for low-depth set-multilinear formulas would yield exponential lower bounds for low-depth homogeneous formulas. A direct approach does not seem to incur an *inherent* exponential loss. So, it might be possible to prove stronger lower bounds for low-depth homogeneous formulas or other related models using this approach or an adaptation of it. We list some open problems below:

1. **Exponential lower bounds.** Prove exponential lower bounds for low-depth homogeneous arithmetic formulas. Prove exponential lower bounds for low-depth, *multi-r-ic*

formulas. A formula is said to be multi-r-ic, if the formal degree of every gate with respect to every variable is at most $r$ [KS17a, KST16b].

2. **Learning low depth circuits.** Our work also raises the prospect of learning low-depth homogeneous formulas given black-box access using the 'learning from lower bounds' paradigm proposed in [GKS20, KS19a]. Obtain learning algorithms for random low-depth homogeneous formulas.

3. **Exploiting the structure of the space of partials using an ideal.** To upper bound SP or APP of a homogeneous formula $C$, we first show in Section 7.4 that the space of partial derivatives of $C$ has some structure and then exploit this structure using shifts or affine projections. There might be a better way to exploit this structure, say by going modulo an appropriately chosen ideal or using random restrictions along with shifts as done in [KLSS17, KS17b]. Exploring this possibility is also an interesting direction for future work.

## Border of sums of ROFs

In Chapter 8, we showed that the border of the sum of two additive constant free $n$-variate ROFs is contained in the sum of $O(n)$ ROFs. We also gave a quasi-polynomial PIT for the border of the sum of two homogeneous depth 5 ROFs. A few natural directions to extend the results presented in Chapter 8 are as follows:

1. **Removing the additive constant free restriction.** Can we remove the requirement that the ROFs be additive constant free from Theorem 1.11 and de-border the border of sums of two ROFs?

2. **De-bordering the border of sums of constantly many ROFs.** Can we extend Theorem 1.11 to the sum of constantly many ROFs?

3. **PIT for the border of sums of two ROFs.** We are only able to give a PIT algorithm for the border of sums of two homogeneous, depth 5 ROFs. Can we extend Theorem 1.12 to the border of sums of two ROFs?

# Bibliography


[AB03] Manindra Agrawal and Somenath Biswas. Primality and identity testing via Chinese remaindering. *J. ACM*, 50(4):429–443, 2003. Conference version appeared in the proceedings of FOCS 1999. 8

[AFS+18] Matthew Anderson, Michael A. Forbes, Ramprasad Saptharishi, Amir Shpilka, and Ben Lee Volk. Identity Testing and Lower Bounds for Read-$k$ Oblivious Algebraic Branching Programs. *ACM Trans. Comput. Theory*, 10(1):3:1–3:30, 2018. Conference version appeared in the proceedings of CCC 2016. 10

[AGK+23] Prashanth Amireddy, Ankit Garg, Neeraj Kayal, Chandan Saha, and Bhargav Thankey. Low-depth arithmetic circuit lower bounds: Bypassing set-multilinearization. In Kousha Etessami, Uriel Feige, and Gabriele Puppis, editors, *50th International Colloquium on Automata, Languages, and Programming, ICALP 2023, July 10-14, 2023, Paderborn, Germany*, volume 261 of *LIPIcs*, pages 12:1–12:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023. 4, 24, 34, 181

[AGKS15] Manindra Agrawal, Rohit Gurjar, Arpita Korwar, and Nitin Saxena. Hitting-sets for ROABP and sum of set-multilinear circuits. *SIAM J. Comput.*, 44(3):669–697, 2015. 10, 62

[Agr05] Manindra Agrawal. Proving lower bounds via pseudo-random generators. In Ramaswamy Ramanujam and Sandeep Sen, editors, *FSTTCS 2005: Foundations of Software Technology and Theoretical Computer Science, 25th International Conference, Hyderabad, India, December 15-18, 2005, Proceedings*, volume 3821 of *Lecture Notes in Computer Science*, pages 92–105. Springer, 2005. 7, 8, 239

[AH19] Eric Allender and Shuichi Hirahara. New insights on the (non-)hardness of cir-

cuit minimization and related problems. *ACM Trans. Comput. Theory*, 11(4):27:1–27:27, 2019. 11

[AHU83] Alfred V. Aho, John E. Hopcroft, and Jeffrey D. Ullman. *Data Structures and Algorithms*. Addison-Wesley, 1983. 172

[AKS04] Manindra Agrawal, Neeraj Kayal, and Nitin Saxena. PRIMES is in P. *Annals of Mathematics*, 160(2):781–793, 2004. 8

[AKV18] Noga Alon, Mrinal Kumar, and Ben Lee Volk. Unbalancing Sets and an Almost Quadratic Lower Bound for Syntactically Multilinear Arithmetic Circuits. In Rocco A. Servedio, editor, *33rd Computational Complexity Conference, CCC 2018, June 22-24, 2018, San Diego, CA, USA*, volume 102 of *LIPIcs*, pages 11:1–11:16. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2018. 5

[Ara11] Manuel Araújo. Classification of quadratic forms. `https://www.math.tecnico.ulisboa.pt/~ggranja/manuel.pdf`, 2011. 105

[AS05] Manindra Agrawal and Nitin Saxena. Automorphisms of finite rings and applications to complexity of problems. In *23rd Annual Symposium on Theoretical Aspects of Computer Science, STACS 2005*, pages 1–17, 2005. 15

[ASS13] Manindra Agrawal, Chandan Saha, and Nitin Saxena. Quasi-polynomial hitting-set for set-depth-Δ formulas. In *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 321–330. ACM, 2013. iii, 8, 10, 18, 49, 50, 51, 54, 55, 61

[ASSS16] Manindra Agrawal, Chandan Saha, Ramprasad Saptharishi, and Nitin Saxena. Jacobian Hits Circuits: Hitting Sets, Lower Bounds for Depth-D Occur-k Formulas and Depth-3 Transcendence Degree-k Circuits. *SIAM J. Comput.*, 45(4):1533–1562, 2016. Conference version appeared in the proceedings of STOC 2012. 9, 18, 29, 38, 39, 72, 73, 81

[AV08] Manindra Agrawal 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*, pages 67–75. IEEE Computer Society, 2008. 6, 17

[AvMV15] Matthew Anderson, Dieter van Melkebeek, and Ilya Volkovich. Deterministic polynomial identity tests for multilinear bounded-read formulae. *Comput. Complex.*, 24(4):695–776, 2015. Conference version appeared in the proceedings of CCC 2011. 9, 30

[AW16] Eric Allender and Fengming Wang. On the power of algebraic branching programs of width two. *Comput. Complex.*, 25(1):217–253, 2016. Conference version appeared in the proceedings of ICALP 2011. 19

[BBB+00] Amos Beimel, Francesco Bergadano, Nader H. Bshouty, Eyal Kushilevitz, and Stefano Varricchio. Learning functions represented as multiplicity automata. *J. ACM*, 47(3):506–530, 2000. 13, 240

[BC92] Michael Ben-Or and Richard Cleve. Computing Algebraic Formulas Using a Constant Number of Registers. *SIAM J. Comput.*, 21(1):54–58, 1992. Conference version appeared in the proceedings of STOC 1988. 17, 19

[BDGT24] Vishwas Bhargava, Pranjal Dutta, Sumanta Ghosh, and Anamay Tengse. The complexity of order-finding for roabps, 2024. 15, 240

[BDS22] C. S. Bhargav, Sagnik Dutta, and Nitin Saxena. Improved lower bound, and proof barrier, for constant depth algebraic circuits. In Stefan Szeider, Robert Ganian, and Alexandra Silva, editors, *47th International Symposium on Mathematical Foundations of Computer Science, MFCS 2022, August 22-26, 2022, Vienna, Austria*, volume 241 of *LIPIcs*, pages 18:1–18:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022. 5, 6

[BDS24] Omkar Baraskar, Agrim Dewan, and Chandan Saha. Testing equivalence to design polynomials. In Olaf Beyersdorff, Mamadou Moustapha Kanté, Orna Kupferman, and Daniel Lokshtanov, editors, *41st International Symposium on Theoretical Aspects of Computer Science, STACS 2024, March 12-14, 2024, Clermont-Ferrand, France*, volume 289 of *LIPIcs*, pages 9:1–9:22. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024. 15, 240

[BDSS24] Omkar Baraskar, Agrim Dewan, Chandan Saha, and Pulkit Sinha. NP-Hardness of Testing Equivalence to Sparse Polynomials and to Constant-Support Polynomials. In Karl Bringmann, Martin Grohe, Gabriele Puppis, and Ola Svensson, editors, *51st International Colloquium on Automata, Languages, and Programming (ICALP 2024)*, volume 297 of *Leibniz International Proceedings in Informatics*

*(LIPIcs)*, pages 16:1–16:21, Dagstuhl, Germany, 2024. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. 15

[Ber70] Elwyn R Berlekamp. Factoring polynomials over large finite fields. *Mathematics of Computation*, 24:713–735, 1970. 105

[BESV24] Aditya Bhaskara, Eric Evert, Vaidehi Srinivas, and Aravindan Vijayaraghavan. New tools for smoothed analysis: Least singular value bounds for random matrices with dependent entries. In Bojan Mohar, Igor Shinkar, and Ryan O'Donnell, editors, *Proceedings of the 56th Annual ACM Symposium on Theory of Computing, STOC 2024, Vancouver, BC, Canada, June 24-28, 2024*, pages 375–386. ACM, 2024. 12

[BG21] Vishwas Bhargava and Sumanta Ghosh. Improved hitting set for orbit of ROABPs. In Mary Wootters and Laura Sanità, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2021, August 16-18, 2021, University of Washington, Seattle, Washington, USA (Virtual Conference)*, volume 207 of *LIPIcs*, pages 30:1–30:23. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021. 21, 238

[BGKS22] Vishwas Bhargava, Ankit Garg, Neeraj Kayal, and Chandan Saha. Learning generalized depth three arithmetic circuits in the non-degenerate case. In Amit Chakrabarti and Chaitanya Swamy, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2022, September 19-21, 2022, University of Illinois, Urbana-Champaign, USA (Virtual Conference)*, volume 245 of *LIPIcs*, pages 21:1–21:22. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022. iv, 12

[BGS75] Theodore Baker, John Gill, and Robert Solovay. Relativizations of the $\mathcal{P} =? \mathcal{NP}$ question. *SIAM Journal on Computing*, 4(4):431–442, 1975. 1

[BHH95] Nader H. Bshouty, Thomas R. Hancock, and Lisa Hellerstein. Learning arithmetic read-once formulas. *SIAM J. Comput.*, 24(4):706–735, 1995. Conference version appeared in the proceedings of STOC 1992. 23, 89, 159

[BHKX22] Mitali Bafna, Jun-Ting Hsieh, Pravesh K. Kothari, and Jeff Xu. Polynomial-time power-sum decomposition of polynomials. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 956–967, 2022. 12

[BIM+20] Markus Bläser, Christian Ikenmeyer, Meena Mahajan, Anurag Pandey, and Nitin Saurabh. Algebraic branching programs, border complexity, and tangent spaces. In Shubhangi Saraf, editor, *35th Computational Complexity Conference, CCC 2020, July 28-31, 2020, Saarbrücken, Germany (Virtual Conference)*, volume 169 of *LIPIcs*, pages 21:1–21:24. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020. 16

[BIZ18] Karl Bringmann, Christian Ikenmeyer, and Jeroen Zuiddam. On algebraic branching programs of small width. *J. ACM*, 65(5):32:1–32:29, 2018. Conference version appeared in the proceedings of CCC 2017. 19

[BLS16] Nikhil Balaji, Nutan Limaye, and Srikanth Srinivasan. An almost cubic lower bound for ΣΠΣ circuits computing a polynomial in VP. *Electronic Colloquium on Computational Complexity (ECCC)*, 23:143, 2016. 6

[BMS13] Malte Beecken, Johannes Mittmann, and Nitin Saxena. Algebraic independence and blackbox identity testing. *Inf. Comput.*, 222:2–19, 2013. Conference version appeared in the proceedings of ICALP 2011. 9, 18, 20, 38, 72, 73

[BS83] Walter Baur and Volker Strassen. The Complexity of Partial Derivatives. *Theor. Comput. Sci.*, 22:317–330, 1983. 4

[BSV20] Vishwas Bhargava, Shubhangi Saraf, and Ilya Volkovich. Reconstruction of depth-4 multilinear circuits. In Shuchi Chawla, editor, *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020, Salt Lake City, UT, USA, January 5-8, 2020*, pages 2144–2160. SIAM, 2020. 13

[BSV21] Vishwas Bhargava, Shubhangi Saraf, and Ilya Volkovich. Reconstruction algorithms for low-rank tensors and depth-3 multilinear circuits. In Samir Khuller and Virginia Vassilevska Williams, editors, *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*, pages 809–822. ACM, 2021. 13

[BSV23] Vishwas Bhargava, Shubhangi Saraf, and Ilya Volkovich. Linear independence, alternants, and applications. In Barna Saha and Rocco A. Servedio, editors, *Proceedings of the 55th Annual ACM Symposium on Theory of Computing, STOC 2023, Orlando, FL, USA, June 20-23, 2023*, pages 441–454. ACM, 2023. 9

[Bür00] Peter Bürgisser. *Completeness and Reduction in Algebraic Complexity Theory*, volume 7 of *Algorithms and computation in mathematics*. Springer, 2000. 4, 15

[Bür01] Peter Bürgisser. The complexity of factors of multivariate polynomials. In *42nd Annual Symposium on Foundations of Computer Science, FOCS 2001, 14-17 October 2001, Las Vegas, Nevada, USA*, pages 378–385. IEEE Computer Society, 2001. 47

[Car06] Enrico Carlini. Reducing the number of variables of a polynomial. In *Algebraic Geometry and Geometric Modeling*, pages 237–247, 2006. 42

[CCL10] Jin-yi Cai, Xi Chen, and Dong Li. Quadratic lower bound for permanent vs. determinant in any characteristic. *Comput. Complex.*, 19(1):37–56, 2010. 41

[CELS18] Suryajith Chillara, Christian Engels, Nutan Limaye, and Srikanth Srinivasan. A Near-Optimal Depth-Hierarchy Theorem for Small-Depth Multilinear Circuits. In Mikkel Thorup, editor, *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, pages 934–945. IEEE Computer Society, 2018. 5

[CGGR23] Abhranil Chatterjee, Sumanta Ghosh, Rohit Gurjar, and Roshan Raj. Border complexity of symbolic determinant under rank one restriction. In Amnon Ta-Shma, editor, *38th Computational Complexity Conference, CCC 2023, July 17-20, 2023, Warwick, UK*, volume 264 of *LIPIcs*, pages 2:1–2:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023. 16

[CGK+24] Pritam Chandra, Ankit Garg, Neeraj Kayal, Kunal Mittal, and Tanmay Sinha. Learning arithmetic formulas in the presence of noise: A general framework and applications to unsupervised learning. In Venkatesan Guruswami, editor, *15th Innovations in Theoretical Computer Science Conference, ITCS 2024, January 30 to February 2, 2024, Berkeley, CA, USA*, volume 287 of *LIPIcs*, pages 25:1–25:19. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024. 12

[CKSV22] Prerona Chatterjee, Mrinal Kumar, Adrian She, and Ben Lee Volk. Quadratic lower bounds for algebraic branching programs and formulas. *Comput. Complex.*, 31(2):8, 2022. Conference version appeared in the proceedings of CCC 2020. 5

[CLS19] Suryajith Chillara, Nutan Limaye, and Srikanth Srinivasan. Small-Depth Multilinear Formula Lower Bounds for Iterated Matrix Multiplication with Applications. *SIAM J. Comput.*, 48(1):70–92, 2019. Conference version appeared in the proceedings of STOC 2001. 5, 32

[Coo71] Stephen A. Cook. The complexity of theorem-proving procedures. In Michael A. Harrison, Ranan B. Banerji, and Jeffrey D. Ullman, editors, *Proceedings of the 3rd Annual ACM Symposium on Theory of Computing, May 3-5, 1971, Shaker Heights, Ohio, USA*, pages 151–158. ACM, 1971. 2

[Coo72] Stephen A. Cook. A hierarchy for nondeterministic time complexity. *Proceedings of the fourth annual ACM symposium on Theory of computing*, 1972. 1

[DDS21] Pranjal Dutta, Prateek Dwivedi, and Nitin Saxena. Demystifying the border of depth-3 algebraic circuits. In *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021, Denver, CO, USA, February 7-10, 2022*, pages 92–103. IEEE, 2021. 9, 16

[DGI+24] Pranjal Dutta, Fulvio Gesmundo, Christian Ikenmeyer, Gorav Jindal, and Vladimir Lysikov. Fixed-parameter debordering of waring rank. In Olaf Beyersdorff, Mamadou Moustapha Kanté, Orna Kupferman, and Daniel Lokshtanov, editors, *41st International Symposium on Theoretical Aspects of Computer Science, STACS 2024, March 12-14, 2024, Clermont-Ferrand, France*, volume 289 of *LIPIcs*, pages 30:1–30:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024. 16

[DL78] Richard A. DeMillo and Richard J. Lipton. A Probabilistic Remark on Algebraic Program Testing. *Inf. Process. Lett.*, 7(4):193–195, 1978. 7

[DMPY12] Zeev Dvir, Guillaume Malod, Sylvain Perifel, and Amir Yehudayoff. Separating multilinear branching programs and formulas. In Howard J. Karloff and Toniann Pitassi, editors, *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC 2012, New York, NY, USA, May 19 - 22, 2012*, pages 615–624. ACM, 2012. 5

[dOSlV16] Rafael Mendes de Oliveira, Amir Shpilka, and Ben lee Volk. Subexponential Size Hitting Sets for Bounded Depth Multilinear Formulas. *Comput. Complex.*, 25(2):455–505, 2016. Conference version appeared in the proceedings of CCC 2015. 8

[DS07] Zeev Dvir and Amir Shpilka. Locally decodable codes with two queries and polynomial identity testing for depth 3 circuits. *SIAM J. Comput.*, 36(5):1404–1434, 2007. Conference version appeared in the proceedings of STOC 2005. 8

[DS22] Pranjal Dutta and Nitin Saxena. Separated borders: Exponential-gap fanin-hierarchy theorem for approximative depth-3 circuits. In *63rd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2022, Denver, CO, USA, October 31 - November 3, 2022*, pages 200–211. IEEE, 2022. 16

[Edm67] Jack Edmonds. Systems of distinct representatives and linear algebra. *Journal of research of the National Bureau of Standards*, 71:241–245, 1967. 10

[Edm79] Jack Edmonds. Matroid intersection. In P.L. Hammer, E.L. Johnson, and B.H. Korte, editors, *Discrete Optimization I*, volume 4 of *Annals of Discrete Mathematics*, pages 39–49. Elsevier, 1979. 10

[FGS18] Michael A. Forbes, Sumanta Ghosh, and Nitin Saxena. Towards blackbox identity testing of log-variate circuits. In Ioannis Chatzigiannakis, Christos Kaklamanis, Dániel Marx, and Donald Sannella, editors, *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018, July 9-13, 2018, Prague, Czech Republic*, volume 107 of *LIPIcs*, pages 54:1–54:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018. 10, 62

[FGT16] Stephen A. Fenner, Rohit Gurjar, and Thomas Thierauf. Bipartite perfect matching is in quasi-nc. In Daniel Wichs and Yishay Mansour, editors, *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 754–763. ACM, 2016. 10

[FK09] Lance Fortnow and Adam R. Klivans. Efficient learning algorithms yield circuit lower bounds. *J. Comput. Syst. Sci.*, 75(1):27–36, 2009. 12

[FKS16] Michael A. Forbes, Mrinal Kumar, and Ramprasad Saptharishi. Functional lower bounds for arithmetic circuits and connections to boolean circuit complexity. In *31st Conference on Computational Complexity, CCC 2016, May 29 to June 1, 2016, Tokyo, Japan*, pages 33:1–33:19, 2016. 32

[FLMS15] Hervé Fournier, Nutan Limaye, Guillaume Malod, and Srikanth Srinivasan. Lower Bounds for Depth-4 Formulas Computing Iterated Matrix Multiplication. *SIAM J. Comput.*, 44(5):1173–1201, 2015. Conference version appeared in the proceedings of STOC 2014. 6

[For15] Michael A. Forbes. Deterministic divisibility testing via shifted partial derivatives. In Venkatesan Guruswami, editor, *IEEE 56th Annual Symposium on Foun-*

*dations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, pages 451–465. IEEE Computer Society, 2015. 9

[FS12] Michael A. Forbes and Amir Shpilka. On identity testing of tensors, low-rank recovery and compressed sensing. In Howard J. Karloff and Toniann Pitassi, editors, *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC 2012, New York, NY, USA, May 19 - 22, 2012*, pages 163–172. ACM, 2012. 8

[FS13] Michael A. Forbes and Amir Shpilka. Quasipolynomial-time identity testing of non-commutative and read-once oblivious algebraic branching programs. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, 26-29 October, 2013, Berkeley, CA, USA*, pages 243–252. IEEE Computer Society, 2013. 9, 10, 13, 18, 30, 61, 63

[FS18] Michael A. Forbes and Amir Shpilka. A PSPACE construction of a hitting set for the closure of small algebraic circuits. In Ilias Diakonikolas, David Kempe, and Monika Henzinger, editors, *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 1180–1192. ACM, 2018. 11

[FSS14] Michael A. Forbes, Ramprasad Saptharishi, and Amir Shpilka. Hitting sets for multilinear read-once algebraic branching programs, in any order. In David B. Shmoys, editor, *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 867–875. ACM, 2014. 10, 18, 49, 61, 63

[Gee99] James F. Geelen. Maximum rank matrix completion. *Linear Algebra and its Applications*, 288:211 – 217, 1999. 10

[GG20] Zeyu Guo and Rohit Gurjar. Improved explicit hitting-sets for ROABPs. In Jaroslaw Byrka and Raghu Meka, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2020, August 17-19, 2020, Virtual Conference*, volume 176 of *LIPIcs*, pages 4:1–4:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020. 10

[GGKS19] Ankit Garg, Nikhil Gupta, Neeraj Kayal, and Chandan Saha. Determinant equivalence test over finite fields and over Q. In *46th International Colloquium on Automata, Languages, and Programming, ICALP 2019, July 9-12, 2019, Greece*, volume 132 of *LIPIcs*, pages 62:1–62:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019. 15

[GHT22] Nashlen Govindasamy, Tuomas Hakoniemi, and Iddo Tzameret. Simple hard instances for low-depth algebraic proofs. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 188–199, 2022. 25

[GKKS14] Ankit Gupta, Pritish Kamath, Neeraj Kayal, and Ramprasad Saptharishi. Approaching the Chasm at Depth Four. *J. ACM*, 61(6):33:1–33:16, 2014. Conference version appeared in the proceedings of CCC 2013. 6, 34, 184

[GKKS16] Ankit Gupta, Pritish Kamath, Neeraj Kayal, and Ramprasad Saptharishi. Arithmetic Circuits: A Chasm at Depth 3. *SIAM J. Comput.*, 45(3):1064–1079, 2016. Conference version appeared in the proceedings of FOCS 2013. 6, 17, 18

[GKL11] Ankit Gupta, Neeraj Kayal, and Satyanarayana V. Lokam. Efficient Reconstruction of Random Multilinear Formulas. In *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, Palm Springs, CA, USA, October 22-25, 2011*, pages 778–787, 2011. 12

[GKL12] Ankit Gupta, Neeraj Kayal, and Satyanarayana V. Lokam. Reconstruction of depth-4 multilinear circuits with top fan-in 2. In Howard J. Karloff and Toniann Pitassi, editors, *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC 2012, New York, NY, USA, May 19 - 22, 2012*, pages 625–642. ACM, 2012. 12

[GKP18] Ignacio García-Marco, Pascal Koiran, and Timothée Pecatte. Polynomial Equivalence Problems for Sum of Affine Powers. In *Proceedings of the 2018 ACM on International Symposium on Symbolic and Algebraic Computation, ISSAC 2018, New York, NY, USA, July 16-19, 2018*, pages 303–310. ACM, 2018. 15, 41, 239

[GKQ13] Ankit Gupta, Neeraj Kayal, and Youming Qiao. Random Arithmetic Formulas Can Be Reconstructed Efficiently. In *Proceedings of the 28th Conference on Computational Complexity, CCC 2013, K.lo Alto, California, USA, 5-7 June, 2013*, pages 1–9, 2013. 13, 22

[GKS17] Rohit Gurjar, Arpita Korwar, and Nitin Saxena. Identity testing for constant-width, and any-order, read-once oblivious arithmetic branching programs. *Theory Comput.*, 13(1):1–21, 2017. Conference version appeared in the proceedings of CCC 2016. 10

[GKS20] Ankit Garg, Neeraj Kayal, and Chandan Saha. Learning sums of powers of low-degree polynomials in the non-degenerate case. In Sandy Irani, editor, *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, November 16-19, 2020*, pages 889–899. IEEE, 2020. iv, 12, 25, 34, 212, 241

[GKST17] Rohit Gurjar, Arpita Korwar, Nitin Saxena, and Thomas Thierauf. Deterministic Identity Testing for Sum of Read-Once Oblivious Arithmetic Branching Programs. *Comput. Complex.*, 26(4):835–880, 2017. Conference version appeared in the proceedings of CCC 2015. 10, 39, 62

[GP18] Joshua A. Grochow and Toniann Pitassi. Circuit complexity, proof complexity, and polynomial identity testing: The ideal proof system. *J. ACM*, 65(6), November 2018. 25

[GR08] Ariel Gabizon and Ran Raz. Deterministic extractors for affine sources over large fields. *Comb.*, 28(4):415–440, 2008. Conference version appeared in the proceedings of FOCS 2005. 8

[Gro12] Joshua Abraham Grochow. *Symmetry and equivalence relations in classical and geometric complexity theory*. PhD thesis, Department of Computer Science, The University of Chicago, Chicago, Illinois, 2012. 15

[GS19] Nikhil Gupta and Chandan Saha. On the symmetries of and equivalence test for design polynomials. In *44th International Symposium on Mathematical Foundations of Computer Science, MFCS 2019, August 26-30, 2019, Aachen, Germany*, volume 138 of *LIPIcs*, pages 53:1–53:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019. 15

[GSS18] Zeyu Guo, Nitin Saxena, and Amit Sinhababu. Algebraic dependencies and PSPACE algorithms in approximative complexity. In Rocco A. Servedio, editor, *33rd Computational Complexity Conference, CCC 2018, June 22-24, 2018, San Diego, CA, USA*, volume 102 of *LIPIcs*, pages 10:1–10:21. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018. 11

[GST20] Nikhil Gupta, Chandan Saha, and Bhargav Thankey. A super-quadratic lower bound for depth four arithmetic circuits. In Shubhangi Saraf, editor, *35th Computational Complexity Conference, CCC 2020, July 28-31, 2020, Saarbrücken, Germany (Virtual Conference)*, volume 169 of *LIPIcs*, pages 23:1–23:31. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020. 6, 21, 32

[GST23] Nikhil Gupta, Chandan Saha, and Bhargav Thankey. Equivalence test for read-once arithmetic formulas. In Nikhil Bansal and Viswanath Nagarajan, editors, *Proceedings of the 2023 ACM-SIAM Symposium on Discrete Algorithms, SODA 2023, Florence, Italy, January 22-25, 2023*, pages 4205–4272. SIAM, 2023. 42, 88, 98, 171

[GT20] Rohit Gurjar and Thomas Thierauf. Linear matroid intersection is in quasi-NC. *Comput. Complex.*, 29(2):9, 2020. Conference version appeared in the proceedings of STOC 2017. 10

[Gup14] Ankit Gupta. Algebraic geometric techniques for depth-4 PIT & Sylvester-Gallai conjectures for varieties. *Electron. Colloquium Comput. Complex.*, 21:130, 2014. 9

[Gup22] Nikhil Gupta. On symmetries of and equivalence tests for two polynomial families and a circuit class. 2022. Ph.D. Thesis. 42, 88, 98

[HH91] Thomas R. Hancock and Lisa Hellerstein. Learning read-once formulas over fields and extended bases. In Manfred K. Warmuth and Leslie G. Valiant, editors, *Proceedings of the Fourth Annual Workshop on Computational Learning Theory, COLT 1991, Santa Cruz, California, USA, August 5-7, 1991*, pages 326–336. Morgan Kaufmann, 1991. 23, 89, 159

[HLT24] Tuomas Hakoniemi, Nutan Limaye, and Iddo Tzameret. Functional lower bounds in algebraic proofs: Symmetry, lifting, and barriers. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*, STOC 2024, page 1396–1404, New York, NY, USA, 2024. Association for Computing Machinery. 25

[HS65] Juris Hartmanis and Richard Edwin Stearns. On the computational complexity of algorithms. *Transactions of the American Mathematical Society*, 117:285–306, 1965. 1

[HS80] Joos Heintz and Claus-Peter Schnorr. Testing polynomials which are easy to compute (extended abstract). In Raymond E. Miller, Seymour Ginsburg, Walter A. Burkhard, and Richard J. Lipton, editors, *Proceedings of the 12th Annual ACM Symposium on Theory of Computing, April 28-30, 1980, Los Angeles, California, USA*, pages 262–272. ACM, 1980. 7, 8

[HY11] Pavel Hrubes and Amir Yehudayoff. Homogeneous formulas and symmetric polynomials. *Comput. Complex.*, 20(3):559–578, 2011. 196

[IKS10] Gábor Ivanyos, Marek Karpinski, and Nitin Saxena. Deterministic polynomial time algorithms for matrix completion problems. *SIAM J. Comput.*, 39(8):3736–3751, 2010. 10

[JQS10] Maurice J. Jansen, Youming Qiao, and Jayalal Sarma. Deterministic Black-Box Identity Testing $pi$-Ordered Algebraic Branching Programs. In Kamal Lodaya and Meena Mahajan, editors, *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2010, December 15-18, 2010, Chennai, India*, volume 8 of *LIPIcs*, pages 296–307. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2010. 10

[Kal85] K. Kalorkoti. A Lower Bound for the Formula Size of Rational Functions. *SIAM J. Comput.*, 14(3):678–687, 1985. 4

[Kay10] Neeraj Kayal. Algorithms for arithmetic circuits. *Electron. Colloquium Comput. Complex.*, 17:73, 2010. 8

[Kay11] Neeraj Kayal. Efficient algorithms for some special cases of the polynomial equivalence problem. In Dana Randall, editor, *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2011, San Francisco, California, USA, January 23-25, 2011*, pages 1409–1421. SIAM, 2011. 15, 41, 42, 90

[Kay12a] Neeraj Kayal. Affine projections of polynomials: extended abstract. In *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC 2012, New York, NY, USA, May 19 - 22, 2012*, pages 643–662, 2012. 15

[Kay12b] Neeraj Kayal. An exponential lower bound for the sum of powers of bounded degree polynomials. *Electronic Colloquium on Computational Complexity (ECCC)*, 19:81, 2012. 6, 34, 184

[KC00] Valentine Kabanets and Jin-yi Cai. Circuit minimization problem. In F. Frances Yao and Eugene M. Luks, editors, *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing, May 21-23, 2000, Portland, OR, USA*, pages 73–79. ACM, 2000. 11

[KI04] Valentine Kabanets and Russell Impagliazzo. Derandomizing polynomial identity tests means proving circuit lower bounds. *Comput. Complex.*, 13(1-2):1–46, 2004. Conference version appeared in the proceedings of STOC 2003. 7, 8

[KLSS17] Neeraj Kayal, Nutan Limaye, Chandan Saha, and Srikanth Srinivasan. An Exponential Lower Bound for Homogeneous Depth Four Arithmetic Formulas. *SIAM J. Comput.*, 46(1):307–335, 2017. Conference version appeared in the proceedings of FOCS 2014. 6, 32, 241

[KMSV13] Zohar Shay Karnin, Partha Mukhopadhyay, Amir Shpilka, and Ilya Volkovich. Deterministic Identity Testing of Depth-4 Multilinear Circuits with Bounded Top Fan-in. *SIAM J. Comput.*, 42(6):2114–2131, 2013. Conference version appeared in the proceedings of STOC 2010. 9, 20, 30, 72

[KNS19] Neeraj Kayal, Vineet Nair, and Chandan Saha. Average-case linear matrix factorization and reconstruction of low width algebraic branching programs. *Comput. Complex.*, 28(4):749–828, 2019. 13, 130

[KNS20] Neeraj Kayal, Vineet Nair, and Chandan Saha. Separation between read-once oblivious algebraic branching programs (roabps) and multilinear depth-three circuits. *ACM Trans. Comput. Theory*, 12(1):2:1–2:27, 2020. 34

[KNST17] Neeraj Kayal, Vineet Nair, Chandan Saha, and Sébastien Tavenas. Reconstruction of Full Rank Algebraic Branching Programs. In *32nd Computational Complexity Conference, CCC 2017, July 6-9, 2017, Riga, Latvia*, pages 21:1–21:61, 2017. 15, 42, 105

[Koi12] Pascal Koiran. Arithmetic circuits: The chasm at depth four gets wider. *Theor. Comput. Sci.*, 448:56–65, 2012. 6, 17

[KS01] Adam R. Klivans and Daniel A. Spielman. Randomness efficient identity testing of multivariate polynomials. In *Proceedings on 33rd Annual ACM Symposium on Theory of Computing, July 6-8, 2001, Heraklion, Crete, Greece*, pages 216–223, 2001. 8, 10, 12, 19, 62, 70, 71, 73, 89, 95, 121, 122, 240

[KS06] Adam R. Klivans and Amir Shpilka. Learning Restricted Models of Arithmetic Circuits. *Theory of Computing*, 2(10):185–206, 2006. 13, 240

[KS07] Neeraj Kayal and Nitin Saxena. Polynomial identity testing for depth 3 circuits. *Comput. Complex.*, 16(2):115–138, 2007. Conference version appeared in the proceedings of CCC 2006. 8

[KS09a] Zohar Shay Karnin and Amir Shpilka. Reconstruction of generalized depth-3 arithmetic circuits with bounded top fan-in. In *Proceedings of the 24th Annual IEEE Conference on Computational Complexity, CCC 2009, Paris, France, 15-18 July 2009*, pages 274–285. IEEE Computer Society, 2009. 12

[KS09b] Neeraj Kayal and Shubhangi 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*, pages 198–207. IEEE Computer Society, 2009. 8

[KS11] Zohar Shay Karnin and Amir Shpilka. Black box polynomial identity testing of generalized depth-3 arithmetic circuits with bounded top fan-in. *Comb.*, 31(3):333–364, 2011. Conference version appeared in the proceedings of CCC 2008. 8

[KS16] Neeraj Kayal and Chandan Saha. Lower Bounds for Depth-Three Arithmetic Circuits with small bottom fanin. *Computational Complexity*, 25(2):419–454, 2016. Conference version appeared in the proceedings of CCC 2015. 32

[KS17a] Neeraj Kayal and Chandan Saha. Multi-k-ic depth three circuit lower bound. *Theory Comput. Syst.*, 61(4):1237–1251, 2017. The conference version appeared in the proceedings of STACS, 2015. 241

[KS17b] Mrinal Kumar and Shubhangi Saraf. On the Power of Homogeneous Depth 4 Arithmetic Circuits. *SIAM J. Comput.*, 46(1):336–387, 2017. Conference version appeared in the proceedings of FOCS 2014. 6, 32, 241

[KS19a] Neeraj Kayal and Chandan Saha. Reconstruction of non-degenerate homogeneous depth three circuits. In Moses Charikar and Edith Cohen, editors, *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, Phoenix, AZ, USA, June 23-26, 2019*, pages 413–424. ACM, 2019. iv, 12, 34, 241

[KS19b] Mrinal Kumar and Ramprasad Saptharishi. The computational power of depth five arithmetic circuits. *SIAM J. Comput.*, 48(1):144–180, 2019. 32

[KS21] Pascal Koiran and Mateusz Skomra. Derandomization and absolute reconstruction for sums of powers of linear forms. *Theor. Comput. Sci.*, 887:63–84, 2021. 10, 15, 20

[KS22] Deepanshu Kush and Shubhangi Saraf. Improved low-depth set-multilinear circuit lower bounds. In Shachar Lovett, editor, *37th Computational Complexity Conference, CCC 2022, July 20-23, 2022, Philadelphia, PA, USA*, volume 234 of *LIPIcs*, pages 38:1–38:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022. 6, 32

[KS23a] Pascal Koiran and Subhayan Saha. Absolute reconstruction for sums of powers of linear forms: degree 3 and beyond. *Comput. Complex.*, 32(2):8, 2023. 15

[KS23b] Deepanshu Kush and Shubhangi Saraf. Near-optimal set-multilinear formula lower bounds. In Amnon Ta-Shma, editor, *38th Computational Complexity Conference, CCC 2023, July 17-20, 2023, Warwick, UK*, volume 264 of *LIPIcs*, pages 15:1–15:33. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023. 6

[KSS14] Neeraj Kayal, Chandan Saha, and Ramprasad Saptharishi. A super-polynomial lower bound for regular arithmetic formulas. In *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 146–153, 2014. 6, 214

[KST16a] Neeraj Kayal, Chandan Saha, and Sébastien Tavenas. An Almost Cubic Lower Bound for Depth Three Arithmetic Circuits. In *43rd International Colloquium on Automata, Languages, and Programming, ICALP 2016, July 11-15, 2016, Rome, Italy*, pages 33:1–33:15, 2016. 6, 32

[KST16b] Neeraj Kayal, Chandan Saha, and Sébastien Tavenas. On the size of homogeneous and of depth four formulas with low individual degree. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 626–632, 2016. 32, 241

[KT90] Erich Kaltofen and Barry M. Trager. Computing with Polynomials Given By Black Boxes for Their Evaluations: Greatest Common Divisors, Factorization, Separation of Numerators and Denominators. *J. Symb. Comput.*, 9(3):301–320, 1990. Conference version appeared in the proceedings of FOCS 1988. 12, 23, 90, 97, 98, 105, 118, 126, 154, 159, 165

[Kum20] Mrinal Kumar. On the power of border of depth-3 arithmetic circuits. *ACM Trans. Comput. Theory*, 12(1), February 2020. 16

[KUW86] Richard M. Karp, Eli Upfal, and Avi Wigderson. Constructing a perfect matching is in random NC. *Comb.*, 6(1):35–48, 1986. Conference version appeared in the proceedings of STOC 1985. 10

[Lad75] Richard E. Ladner. On the structure of polynomial time reducibility. *J. ACM*, 22(1):155–171, January 1975. 1

[Lam04] T. Y. Lam. *Introduction To Quadratic Forms Over Fields*. American Mathematical Society, 2004. 105

[Lev73] Leonid A. Levin. Universal sequential search problems. *Problems of Information Transmission*, 9(3), 1973. 2

[LLL82] Arjen K Lenstra, Hendrik W Lenstra, and László Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261(4):515–534, 1982. 105

[Lov79] László Lovász. On determinants, matchings, and random algorithms. In Lothar Budach, editor, *Fundamentals of Computation Theory, FCT 1979, Proceedings of the Conference on Algebraic, Arthmetic, and Categorial Methods in Computation Theory, Berlin/Wendisch-Rietz, Germany, September 17-21, 1979*, pages 565–574. Akademie-Verlag, Berlin, 1979. 10

[Lov89] László Lovász. Singular spaces of matrices and their application in combinatorics. *Boletim da Sociedade Brasileira de Matemática - Bulletin/Brazilian Mathematical Society*, 20(1):87–99, 1989. 10

[LST21] Nutan Limaye, Srikanth Srinivasan, and Sébastien Tavenas. Superpolynomial lower bounds against low-depth algebraic circuits. In *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021, Denver, CO, USA, February 7-10, 2022*, pages 804–814. IEEE, 2021. iv, 5, 6, 9, 24, 25, 32, 33, 34, 35, 181, 184, 185, 186, 187, 196, 205, 214, 215, 240

[LV03] Richard J. Lipton and Nisheeth K. Vishnoi. Deterministic identity testing for multivariate polynomials. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms, January 12-14, 2003, Baltimore, Maryland, USA*, pages 756–760. ACM/SIAM, 2003. 19

[MNS20] Janaky Murthy, Vineet Nair, and Chandan Saha. Randomized Polynomial-Time Equivalence Between Determinant and Trace-IMM Equivalence Tests. In *45th*

*International Symposium on Mathematical Foundations of Computer Science, MFCS 2020, August 24-28, 2020, Prague, Czech Republic*, volume 170 of *LIPIcs*, pages 72:1–72:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020. 15

[MR04] Thierry Mignon and Nicolas Ressayre. A quadratic bound for the determinant and permanent problem. *International Mathematics Research Notes*, 2004(79):4241–4253, 2004. 41

[MS01] Ketan Mulmuley and Milind A. Sohoni. Geometric complexity theory I: an approach to the P vs. NP and related problems. *SIAM J. Comput.*, 31(2):496–526, 2001. 13

[MS21] Dori Medini and Amir Shpilka. Hitting sets and reconstruction for dense orbits in $\mathrm{VP}_e$ and $\Sigma\Pi\Sigma$ circuits. In Valentine Kabanets, editor, *36th Computational Complexity Conference, CCC 2021, July 20-23, 2021, Toronto, Ontario, Canada (Virtual Conference)*, volume 200 of *LIPIcs*, pages 19:1–19:27. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021. 15, 19, 20, 21, 23, 61, 72

[MT18] Meena Mahajan and Anuj Tawari. Sums of read-once formulas: How many summands are necessary? *Theor. Comput. Sci.*, 708:34–45, 2018. 41, 221

[Mur93] K. Murota. Mixed matrices: Irreducibility and decomposition. In R. A. Brualdi, S. Friedland, and V. Klee, editors, *Combinatorial and Graph-Theoretical Problems in Linear Algebra. The IMA Volumes in Mathematics and its Applications, vol 50.*, pages 39–71. Springer, New York, NY, 1993. 10

[MV18] Daniel Minahan and Ilya Volkovich. Complete derandomization of identity testing and reconstruction of read-once formulas. *ACM Trans. Comput. Theory*, 10(3):10:1–10:11, 2018. Conference version appeared in the proceedings of CCC 2017. 9, 13, 23, 89, 159

[MVV87] Ketan Mulmuley, Umesh V. Vazirani, and Vijay V. Vazirani. Matching is as easy as matrix inversion. *Comb.*, 7(1):105–113, 1987. Conference version appeared in the proceedings of STOC 1987. 8, 10

[Nis91] Noam Nisan. Lower Bounds for Non-Commutative Computation (Extended Abstract). In Cris Koutsougeras and Jeffrey Scott Vitter, editors, *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing, May 5-8, 1991, New Orleans, Louisiana, USA*, pages 410–418. ACM, 1991. 16, 24, 47

[NSV94] H. Narayanan, Huzur Saran, and Vijay V. Vazirani. Randomized Parallel Algorithms for Matroid Union and Intersection, With Applications to Arboresences and Edge-Disjoint Spanning Trees. *SIAM J. Comput.*, 23(2):387–397, 1994. Conference version appeared in the proceedings of SODA 1992. 10

[NW97] Noam Nisan and Avi Wigderson. Lower Bounds on Arithmetic Circuits Via Partial Derivatives. *Computational Complexity*, 6(3):217–234, 1997. Conference version appeared in the proceedings of FOCS 1995. 6, 8, 19, 32, 184

[Ore22] Ö. Ore. Über höhere Kongruenzen. *Norsk Mat. Forenings Skrifter Ser. I*, (7), 1922. 15 pages. 7

[PS20] Shir Peleg and Amir Shpilka. A generalized Sylvester-Gallai type theorem for quadratic polynomials. In Shubhangi Saraf, editor, *35th Computational Complexity Conference, CCC 2020, July 28-31, 2020, Saarbrücken, Germany (Virtual Conference)*, volume 169 of *LIPIcs*, pages 8:1–8:33. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020. 9

[PS21] Shir Peleg and Amir Shpilka. Polynomial time deterministic identity testing algorithm for $\Sigma^{[3]}\Pi\Sigma\Pi^{[2]}$ circuits via Edelstein-Kelly type theorem for quadratic polynomials. In Samir Khuller and Virginia Vassilevska Williams, editors, *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*, pages 259–271. ACM, 2021. 9

[Raz03] Ran Raz. On the Complexity of Matrix Product. *SIAM J. Comput.*, 32(5):1356–1369, 2003. Conference version appeared in the proceedings of STOC 2002. 184

[Raz06] Ran Raz. Separation of Multilinear Circuit and Formula Size. *Theory of Computing*, 2(6):121–135, 2006. Conference version appeared in the proceedings of FOCS 2004. 5

[Raz10] Ran Raz. Elusive Functions and Lower Bounds for Arithmetic Circuits. *Theory of Computing*, 6(1):135–177, 2010. Conference version appeared in the proceedings of STOC 2008. 6, 32

[RS05] Ran Raz and Amir Shpilka. Deterministic polynomial identity testing in non-commutative models. *Comput. Complex.*, 14(1):1–19, 2005. Conference version appeared in the proceedings of CCC 2004. 9

[RSY08] Ran Raz, Amir Shpilka, and Amir Yehudayoff. A Lower Bound for the Size of Syntactically Multilinear Arithmetic Circuits. *SIAM J. Comput.*, 38(4):1624–1647, 2008. Conference version appeared in the proceedings of FOCS 2007. 5

[RY08] Ran Raz and Amir Yehudayoff. Balancing Syntactically Multilinear Arithmetic Circuits. *Computational Complexity*, 17(4):515–535, 2008. 5

[RY09] Ran Raz and Amir Yehudayoff. Lower Bounds and Separations for Constant Depth Multilinear Circuits. *Computational Complexity*, 18(2):171–207, 2009. Conference version appeared in the proceedings of CCC 2008. 5

[Sap] Ramprasad Saptharishi. A selection of lower bounds in arithmetic circuit complexity. 6

[Sax06] Nitin Saxena. *Morphisms of rings and applications to complexity*. PhD thesis, Indian Institute of Technology, Kanpur, 2006. 14

[Sax08] Nitin Saxena. Diagonal circuit identity testing and lower bounds. In Luca Aceto, Ivan Damgård, Leslie Ann Goldberg, Magnús M. Halldórsson, Anna Ingólfsdóttir, and Igor Walukiewicz, editors, *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*, pages 60–71. Springer, 2008. 8, 16, 18

[Sax09] Nitin Saxena. Progress on polynomial identity testing. *Bull. EATCS*, 99:49–79, 2009. 11

[Sax14] Nitin Saxena. Progress on polynomial identity testing-II. In M. Agrawal and V. Arvind, editors, *Perspectives in Computational Complexity*, volume 26 of *Progress in Computer Science and Applied Logic*, pages 131–146. Birkhäuser, Cham, 2014. 11

[Sch80] Jacob T. Schwartz. Fast Probabilistic Algorithms for Verification of Polynomial Identities. *J. ACM*, 27(4):701–717, 1980. 7

[Ser73] Jean-Pierre Serre. *A course in arithmetic*. Springer, 1973. 105

[Sha92] Adi Shamir. IP = PSPACE. *J. ACM*, 39(4):869–877, October 1992. 2, 7

[Shp02] Amir Shpilka. Affine projections of symmetric polynomials. *J. Comput. Syst. Sci.*, 65(4):639–659, 2002. Conference version appeared in the proceedings of CCC 2001. 19

[Shp07] Amir Shpilka. Interpolation of depth-3 arithmetic circuits with two multiplication gates. In David S. Johnson and Uriel Feige, editors, *Proceedings of the 39th Annual ACM Symposium on Theory of Computing, San Diego, California, USA, June 11-13, 2007*, pages 284–293. ACM, 2007. 12

[Shp19] Amir Shpilka. Sylvester-Gallai type theorems for quadratic polynomials. In Moses Charikar and Edith Cohen, editors, *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, Phoenix, AZ, USA, June 23-26, 2019*, pages 1203–1214. ACM, 2019. 9

[Sin16] Gaurav Sinha. Reconstruction of real depth-3 circuits with top fan-in 2. In Ran Raz, editor, *31st Conference on Computational Complexity, CCC 2016, May 29 to June 1, 2016, Tokyo, Japan*, volume 50 of *LIPIcs*, pages 31:1–31:53. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2016. 12

[Sin22] Gaurav Sinha. Efficient reconstruction of depth three arithmetic circuits with top fan-in two. In Mark Braverman, editor, *13th Innovations in Theoretical Computer Science Conference, ITCS 2022, January 31 - February 3, 2022, Berkeley, CA, USA*, volume 215 of *LIPIcs*, pages 118:1–118:33. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022. 12

[SS97] Victor Shoup and Roman Smolensky. Lower Bounds for Polynomial Evaluation and Interpolation Problems. *Computational Complexity*, 6(4):301–311, 1997. Conference version appeared in the proceedings of FOCS 1991. 6

[SS12] Nitin Saxena and C. Seshadhri. Blackbox Identity Testing for Bounded Top-Fanin Depth-3 Circuits: The Field Doesn't Matter. *SIAM J. Comput.*, 41(5):1285–1298, 2012. Conference version appeared in the proceedings of STOC 2011. 8

[SS13] Nitin Saxena and C. Seshadhri. From Sylvester-Gallai configurations to rank bounds: Improved blackbox identity test for depth-3 circuits. *J. ACM*, 60(5):33:1–33:33, 2013. Conference version appeared in the proceedings of FOCS 2010. 8

[SSS09] Chandan Saha, Ramprasad Saptharishi, and Nitin Saxena. The power of depth 2 circuits over algebras. In Ravi Kannan and K. Narayan Kumar, editors, *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2009, December 15-17, 2009, IIT Kanpur, India*, volume 4 of *LIPIcs*, pages 371–382. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2009. 8, 19

[SSS13] Chandan Saha, Ramprasad Saptharishi, and Nitin Saxena. A case of depth-3 identity testing, sparse factorization and duality. *Comput. Complex.*, 22(1):39–69, 2013. 8, 18

[ST17] Ola Svensson and Jakub Tarnawski. The matching problem in general graphs is in quasi-nc. In Chris Umans, editor, *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 696–707. IEEE Computer Society, 2017. 10

[ST24] Chandan Saha and Bhargav Thankey. Hitting sets for orbits of circuit classes and polynomial families. *ACM Trans. Comput. Theory*, 16(3), September 2024. A conference version of this article appeared in the proceedings of the 25-th International Conference on Randomization and Computation (RANDOM), 2021. 8, 14, 16, 20, 49, 72

[Str73] Volker Strassen. Die berechnungskomplexiät von elementarysymmetrischen funktionen und von iterpolationskoeffizienten. *Numerische Mathematik*, 20:238–251, 1973. 4

[SV14] Amir Shpilka and Ilya Volkovich. On Reconstruction and Testing of Read-Once Formulas. *Theory Comput.*, 10:465–514, 2014. Conference version appeared in the proceedings of STOC 2008. 13, 23, 89, 159, 239

[SV15] Amir Shpilka and Ilya Volkovich. Read-once polynomial identity testing. *Comput. Complex.*, 24(3):477–532, 2015. Conference versions appeared in the proceedings of STOC 2008 and APPROX-RANDOM 2009. 9, 18, 35, 41, 72, 83

[SV18] Shubhangi Saraf and Ilya Volkovich. Black-Box Identity Testing of Depth-4 Multilinear Circuits. *Comb.*, 38(5):1205–1238, 2018. Conference version appeared in the proceedings of STOC 2011. 9, 20, 30, 72

[SW01] Amir Shpilka and Avi Wigderson. Depth-3 arithmetic circuits over fields of characteristic zero. *Computational Complexity*, 10(1):1–27, 2001. Conference version appeared in the proceedings of CCC 1999. 6

[SY10] Amir Shpilka and Amir Yehudayoff. Arithmetic Circuits: A survey of recent results and open questions. *Foundations and Trends in Theoretical Computer Science*, 5(3-4):207–388, 2010. 11

[Tav15] Sébastien Tavenas. Improved bounds for reduction to depth 4 and depth 3. *Inf. Comput.*, 240:2–11, 2015. Conference version appeared in the proceedings of MFCS 2013. 6, 17

[Tha22] Justin Thaler. Proofs, arguments, and zero-knowledge. *Foundations and Trends® in Privacy and Security*, 4(2–4):117–660, 2022. 2

[Thi98] Thomas Thierauf. The isomorphism problem for read-once branching programs and arithmetic circuits. *Chicago J. Theor. Comput. Sci.*, 1998, 1998. 14

[TLS22] Sébastien Tavenas, Nutan Limaye, and Srikanth Srinivasan. Set-multilinear and non-commutative formula lower bounds for iterated matrix multiplication. In Stefano Leonardi and Anupam Gupta, editors, *STOC '22: 54th Annual ACM SIGACT Symposium on Theory of Computing, Rome, Italy, June 20 - 24, 2022*, pages 416–425. ACM, 2022. 5

[Val79] Leslie G. Valiant. Completeness Classes in Algebra. In *Proceedings of the 11h Annual ACM Symposium on Theory of Computing, April 30 - May 2, 1979, Atlanta, Georgia, USA*, pages 249–261, 1979. 3, 10, 31

[Vol16] Ilya Volkovich. A guide to learning arithmetic circuits. In Vitaly Feldman, Alexander Rakhlin, and Ohad Shamir, editors, *Proceedings of the 29th Conference on Learning Theory, COLT 2016, New York, USA, June 23-26, 2016*, volume 49 of *JMLR Workshop and Conference Proceedings*, pages 1540–1561. JMLR.org, 2016. 12

[VSBR83] Leslie G. Valiant, Sven Skyum, S. Berkowitz, and Charles Rackoff. Fast Parallel Computation of Polynomials Using Few Processors. *SIAM J. Comput.*, 12(4):641–644, 1983. 6, 17

[Wal13] Lars Ambrosius Wallenborn. Computing the hilbert symbol, quadratic form equivalence and integer factoring. Diploma thesis, Rheinischen Friedrich-Wilhelms-Universität Bonn, 2013. 106

[Yau16] Morris Yau. Almost cubic bound for depth three circuits in VP. *Electronic Colloquium on Computational Complexity (ECCC)*, 23:187, 2016. 6

[Zip79] Richard Zippel. Probabilistic algorithms for sparse polynomials. In *Symbolic and Algebraic Computation, EUROSAM '79, An International Symposiumon Symbolic and Algebraic Computation, Marseille, France, June 1979, Proceedings*, pages 216–226, 1979. 7