Lower Bounds for Arithmetic Circuits via the Hankel Matrix

Nathanaël Fijalkow
CNRS, LaBRI, Bordeaux, France
The Alan Turing Institute of data science, London, United Kingdom
Nathanael.Fijalkow@labri.fr

Guillaume Lagarde
LaBRI, Bordeaux, France
guillaume.lagarde@labri.fr

Pierre Ohlmann
Université de Paris, IRIF, CNRS, F-75013 Paris, France
Pierre.Ohlmann@irif.fr

Olivier Serre
Université de Paris, IRIF, CNRS, F-75013 Paris, France
Olivier.Serre@cnrs.fr

Abstract

We study the complexity of representing polynomials by arithmetic circuits in both the commutative and the non-commutative settings. To analyse circuits we count their number of parse trees, which describe the non-associative computations realised by the circuit.

In the non-commutative setting a circuit computing a polynomial of degree d has at most $2^{O(d)}$ parse trees. Previous superpolynomial lower bounds were known for circuits with up to $2^{d^{1/3-\epsilon}}$ parse trees, for any $\varepsilon > 0$. Our main result is to reduce the gap by showing a superpolynomial lower bound for circuits with just a small defect in the exponent for the total number of parse trees, that is $2^{d^{1-\varepsilon}}$, for any $\varepsilon > 0$.

In the commutative setting a circuit computing a polynomial of degree d has at most $2^{O(d \log d)}$ parse trees. We show a superpolynomial lower bound for circuits with up to $2^{d^{1/3-\varepsilon}}$ parse trees, for any $\varepsilon > 0$. When d is polylogarithmic in n, we push this further to up to $2^{d^{1-\varepsilon}}$ parse trees.

While these two main results hold in the associative setting, our approach goes through a precise understanding of the more restricted setting where multiplication is not associative, meaning that we distinguish the polynomials $(xy)z$ and $x(yz)$. Our first and main conceptual result is a characterization result: we show that the size of the smallest circuit computing a given non-associative polynomial is exactly the rank of a matrix constructed from the polynomial and called the Hankel matrix. This result applies to the class of all circuits in both commutative and non-commutative settings, and can be seen as an extension of the seminal result of Nisan giving a similar characterization for non-commutative algebraic branching programs. Our key technical contribution is to provide generic lower bound theorems based on analyzing and decomposing the Hankel matrix, from which we derive the results mentioned above.

The study of the Hankel matrix also provides a unifying approach for proving lower bounds for polynomials in the (classical) associative setting. We demonstrate this by giving alternative proofs of recent lower bounds as corollaries of our generic lower bound results.

2012 ACM Subject Classification Theory of computation → Circuit complexity; Theory of computation → Algebraic complexity theory

Keywords and phrases Arithmetic Circuit Complexity, Lower Bounds, Parse Trees, Hankel Matrix

Digital Object Identifier 10.4230/LIPIcs.STACS.2020.24

Related Version A full version of the paper is available at `https://hal.archives-ouvertes.fr/hal-02440692`.

37th International Symposium on Theoretical Aspects of Computer Science (STACS 2020).
Editors: Christophe Paul and Markus Bläser; Article No. 24; pp. 24:1–24:16

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

1 Introduction

The model of arithmetic circuits is the algebraic analogue of Boolean circuits: the latter computes Boolean functions and the former computes polynomials, replacing OR gates by addition and AND gates by multiplication. Computational complexity theory is concerned with understanding the expressive power of such models. A rich theory investigates the algebraic complexity classes **VP** and **VNP** introduced by Valiant [25]. A widely open problem in this area of research is to explicitly construct hard polynomials, meaning for which we can prove super polynomial lower bounds. To this day the best general lower bounds for arithmetic circuits were given by Baur and Strassen [4] for the polynomial $\sum_{i=1}^{n} x_i^d$, which requires $\Omega(n \log d)$ operations.

The seminal paper of Nisan [19] initiated the study of non-commutative computation: in this setting variables do not commute, and therefore xy and yx are considered as being two distinct monomials. Non-commutative computations arise in different scenarios, the most common mathematical examples being when working with algebras of matrices, group algebras of non-commutative groups or the quaternion algebra. A second motivation for studying the non-commutative setting is that it makes it easier to prove lower bounds which can then provide powerful ideas for the commutative case. Indeed, commutativity allows a circuit to rely on cancellations and to share calculations across different gates, making them more complicated to analyze.

1.1 Nisan's Characterization for ABP

The main result of Nisan [19] is to give a characterization of the smallest ABP computing a given polynomial. As a corollary of this characterization Nisan obtains exponential lower bounds for the non-commutative permanent against the subclass of circuits given by ABPs.

We sketch the main ideas behind Nisan's characterization, since our first contribution is to extend these ideas to the class of all non-associative circuits. An ABP is a layered graph with two distinguished vertices, a source and a target. The edges are labelled by affine functions in a given set of variables. An ABP computes a polynomial obtained by summing over all paths from the source to the target, with the value of a path being the multiplication of the affine functions along the traversed edges. Fix a polynomial f, and define following Nisan a matrix N_f whose rows and columns are indexed by monomials: for u, v two monomials, let $N_f(u, v)$ denote the coefficient of the monomial $u \cdot v$ in f.

The beautiful and surprisingly simple characterization of Nisan states that for a homogeneous (i.e., all monomials have the same degree) non-commutative polynomial f, the size of the smallest ABP computing f is exactly the rank of N_f. The key idea is that the computation of the polynomial in an ABP can be split into two parts: let r be a vertex in an ABP $\mathcal{C}$ computing the polynomial f, then we can split $\mathcal{C}$ into two ABPs, one with the original source and target r and the other one with source r and the original target. We let L_r and R_r denote the polynomials computed by these two ABPs. For u, v two monomials, we observe that the coefficient of uv in f is equal to $\sum_r L_r(u)R_r(v)$, where r ranges over all vertices of $\mathcal{C}$, $L_r(u)$ is the coefficient of u in L_r, and $R_r(v)$ is the coefficient of v in R_r. We see this as a matrix equality: $N_f = \sum_r L_r \cdot R_r$, where L_r is seen as a column vector, and R_r as a row vector. By subadditivity of the rank and since the product of a column vector by a row vector is a matrix of rank at most 1, this implies that $\operatorname{rank}(N_f)$ is bounded by the size of the ABP, yielding the lower bound in Nisan's result.

The crucial idea of splitting the computation of a monomial into two parts had been independently developed by Fliess when studying so-called *Hankel Matrices* in [9] to derive a very similar result in the field of *weighted automata*, which are finite state machines

recognising *words series*, i.e., functions from finite words into a field. Fliess' theorem [9, Th. 2.1.1] states that the size of the smallest weighted automaton recognising a word series f is exactly the rank of the Hankel matrix of f. The key insight to relate the two results is to see a non-commutative monomial as a finite word over the alphabet whose letters are the variables. Using this correspondence one can obtain Nisan's theorem from Fliess' theorem, observing that the Hankel matrix coincides with the matrix N_f defined by Nisan and that acyclic weighted automata correspond to ABPs. (We refer to an early technical report of this work for more details on this correspondence [8].)

1.2 Non-Associative Computations

Hrubeš, Wigderson and Yehudayoff in [12] drop the associativity rule and show how to define the complexity classes **VP** and **VNP** in the absence of either commutativity or associativity (or both) and prove that these definitions are sound in particular by obtaining the completeness of the permanent.

In the same way that a non-commutative monomial can be seen as a word, a non-commutative and non-associative monomial such as $(xy)(x(zy))$ can be seen as a tree, and more precisely as an ordered binary rooted tree whose leaves are labelled by variables. The starting point of our work was to exploit this connection. The work of Bozapalidis and Louscou-Bozapalidou [5] extends Fliess' result to trees; although we do not technically rely on their results they serve as a guide, in particular for understanding how to decompose trees.

Let us return to the key idea in Nisan's proof, which is to decompose the computation of an ABP into two parts. The way a monomial, e.g., $x_1x_2x_3\cdots x_d$, is evaluated in an ABP is very constrained, namely from left to right, or if we make the implicit non-associative structure explicit as $w = (\cdots(((x_1x_2)x_3)x_4)\cdots)x_d$. The decompositions of w into two monomials u, v are of the form $u = (\cdots((x_1x_2)x_3)\cdots)x_{i-1})$ and $v = (\cdots((\Box x_i)x_{i+1})\cdots)x_d$. Here $\Box$ is a new fresh variable (the *hole*) to be substituted by u. Moving to non-associative polynomials, a monomial is a tree whose leaves are labelled by variables. A *context* is a monomial over the set of variables extended with a new fresh one denoted $\Box$ and occurring exactly once. For instance the composition of the monomial $t = z((xx)y)$ with the context $c = (xy)((z\Box)y)$ is the monomial $c[t] = (xy)((z(z((xx)y)))y)$.

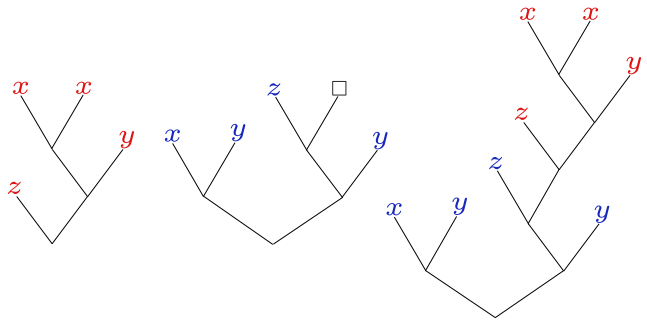

Figure 1 On the left hand side the monomial t, in the middle the context c, and on the right hand side the monomial $c[t]$.

Let f be a non-associative (possibly commutative) polynomial f, the *Hankel matrix* H_f of f is defined as follows: the rows of H_f are indexed by contexts and the columns by monomials, the value of $H_f(c, t)$ at row c and column t is the coefficient of the monomial $c[t]$ in f.

Extending Nisan's proof to computations in a *general circuit*, which are done along trees, we obtain a characterization in the non-associative setting.

▶ **Theorem 1.** *Let f be a non-associative homogeneous polynomial and let H_f be its Hankel matrix. Then, the size of the smallest circuit computing f is exactly rank(H_f).*

Note that this is a characterization result: the Hankel matrix exactly captures the size of the smallest circuit computing f (upper and lower bounds), exactly as in Nisan's result. Hence, understanding the rank of the Hankel matrix is equivalent to studying circuits for f. We recover and extend Nisan's characterization as a special case of our result.

Parse Trees

At an intuitive level, parse trees can be used to explain in what way a circuit uses the associativity rule. Consider the case of a circuit computing the (associative) monomial $2xyz$. Since this monomial corresponds to two non-associative monomials: $(xy)z$ and $x(yz)$, the circuit may sum different computations, for instance $3(xy)z - x(yz)$, which up to associativity is $2xyz$. We say that such a circuit contains two parse trees, corresponding to the two different ways of parenthesizing xyz.

The *shape* of a non-associative monomial is the tree obtained by forgetting the variables, e.g., the shape of $(z((xy)((xx)y)))$ is $(\_\ ((\_\ \_)((\_\ \_)\ \_)))$. The parse trees of a circuit $\mathcal{C}$ are the shapes induced by computations in $\mathcal{C}$.

Many interesting classes of circuits can be defined by restricting the set of allowed parse trees, both in the commutative and the non-commutative setting. The simplest such class is that of Algebraic Branching Programs (ABP) [19, 7, 21], whose only parse trees are left-combs, that is, the variables are multiplied sequentially. Lagarde, Malod and Perifel introduced in [16] the class of Unique Parse Tree circuits (UPT), which are circuits computing non-commutative homogeneous (but associative) polynomials such that all monomials are evaluated in the same non-associative way. The class of skew circuits [24, 2, 18, 17] and its extension small non-skew depth circuits [17], together with the class of unambiguous circuits [3] are all defined via parse tree restrictions. Last but not least, the class of k-PT circuits [3, 15, 23] is simply the class of circuits having at most k parse trees.

Contributions and Outline

In this paper we prove lower bounds for classes of circuits with parse tree restrictions, both in the commutative and non-commutative setting.

Our first and conceptually main contribution is the characterization result stated in Theorem 5 and proved in Section 2, which gives an algebraic approach to understanding circuits in the non-associative setting. All the subsequent results in this paper are based on this approach.

Our most technical developments are discussed in Section 3. We prove generic lower bound results by further analyzing and decomposing the Hankel matrix, with the following proof scheme. We consider a polynomial f in the associative setting. Let $\mathcal{C}$ be a circuit computing f. Forgetting about associativity we can see $\mathcal{C}$ as computing a non-associative polynomial $\tilde{f}$, which projects onto f, meaning is equal to f assuming associativity. This induces a set of linear constraints: for instance if the monomial xyz has coefficient 3 in f, then we know that $\tilde{f}((xy)z) + \tilde{f}(x(yz)) = 3$. We make use of the linear constraints to derive lower bounds on the rank of the Hankel matrix $H_{\tilde{f}}$, yielding a lower bound on the size of $\mathcal{C}$.

Sections 3.1 and 3.2 are devoted to the definition of parse trees and a classical tool for proving lower bounds, partial derivative matrices. We can already show at this point how Theorem 5 can be specialized to give a characterization result for UPT circuits, extending Nisan's result. (We note that a characterization result for UPT circuits was already known [16], we slightly improve on it.) As a corollary we obtain exponential lower bounds on the size of the smallest UPT circuit computing the permanent.

The final section is devoted to applications of our results, where we obtain superpolynomial and exponential lower bounds for various classes. In the results mentioned below, n is the number of variables, d is the degree of the polynomial, and k the number of parse trees. We note that the lower bounds hold for any (prime) n, any d, and any field.

We obtain alternative proofs of some known lower bounds: unambiguous circuits [3], skew circuits [17] and small non-skew depth circuits (obtaining a much shorter proof than [17]).

Our novel results are:

- *Slightly unbalanced circuits.* We extend the exponential lower bound from [17] on $\frac{1}{5}$-unbalanced circuits to $\left(\frac{1}{2}-\varepsilon\right)$-unbalanced circuits.
- *Slightly balanced circuits.* We derive a new exponential lower bound for ε-balanced circuits.
- *Circuits with k parse trees in the non-commutative setting.* We extend the superpolynomial lower bound of [15] from $k = 2^{d^{1/3-\varepsilon}}$ to $k = 2^{d^{1-\varepsilon}}$, the total number of possible non-commutative parse trees being $2^{O(d)}$.
- *Circuits with k parse trees in the commutative setting.* We substantially extend the superpolynomial lower bound from [3] from $k = d^{1/2-\varepsilon}$ to $k = 2^{d^{1/3-\varepsilon}}$, and even to $k = 2^{d^{1-\varepsilon}}$ when d is polylogarithmic in n.

Related Work

We argued that proving lower bounds in the non-commutative setting is easier, but this has not yet materialized since the best lower bound for general circuits in this setting is the same as in the commutative setting (by Baur and Strassen, already mentionned above). Indeed, recent impressive results suggest that this may be hard: Carmosino, Impagliazzo, Lovett, and Mihajlin [6] (essentially) proved that a lower bound in the non-commutative setting which would be slightly stronger than superlinear can be amplified to get strong lower bounds (even exponential, in some cases).

Most approaches for proving lower bounds rely on algebraic techniques and the rank of some matrix. A different and beautiful approach was investigated by Hrubeš, Wigderson and Yehudayoff [12] in the non-commutative setting through the study of the so-called *sum-of-squares problem*. Roughly speaking, the goal is to decompose $(x_1^2 + \cdots + x_k^2) \cdot (y_1^2 + \cdots + y_k^2)$ into a sum of n squared bilinear forms in the variables x_i and y_j. They show that almost any superlinear bound on n implies non-trivial lower bounds on the size of any non-commutative circuit computing the permanent.

The quest of finding lower bounds is deeply connected to another problem called polynomial identity testing (PIT) for which the goal is to decide whether a given circuit computes the formal zero polynomial. The connection was shown in [13], in which it is proved that providing an efficient deterministic algorithm to solve the problem implies strong lower bounds either in the arithmetic or boolean setting. PIT was widely investigated in the commutative and non-commutative settings for classes of circuits based on parse trees restrictions, see e.g., [22, 10, 1, 11, 23].

2 Characterizing Non-Associative Circuits

2.1 Basic Definitions

For an integer $d \in \mathbb{N}$, we let $[d]$ denote the integer interval $\{1, \dots, d\}$.

Polynomials

Let K be a field and let X be a set of ***variables***. Following [12] we consider that unless otherwise stated multiplication is neither commutative nor associative. We assume however that addition is commutative and associative, and that multiplication distributes over addition. A ***monomial*** is a product of variables in X and a polynomial f is a formal finite sum $\sum_i c_i m_i$ where m_i is a monomial and $c_i \in K$ is a non-zero element called the coefficient of m_i in f. We let $f(m_i)$ denote the coefficient of m_i in f, so that $f = \sum_i f(m_i) m_i$.

The ***degree*** of a monomial is defined in the usual way, i.e., $\deg(x) = 1$ when $x \in X$ and $\deg(m_1 m_2) = \deg(m_1) + \deg(m_2)$; the degree of a polynomial f is the maximal degree of a monomial in f. A polynomial is ***homogeneous*** if all its monomials have the same degree. Depending on whether we include the relations $u \cdot v = v \cdot u$ (commutativity) and $u \cdot (v \cdot w) = (u \cdot v) \cdot w$ (associativity) we obtain four classes of polynomials.

Unless otherwise specified, for a polynomial f we use n for the number of variables and d for the degree.

Trees and Contexts

The ***trees*** we consider have a single root and binary branching (every internal node has exactly two children). To account for the commutative and for the non-commutative setting we use either ***unordered trees*** or ***ordered trees***, the only difference being that in the case of ordered trees we distinguish the left child from the right child. We let ***Tree*** denote the set of trees (it will be clear from the context whether they are ordered or not). The size of a tree is defined as its number of leaves.

A non-associative monomial m is a tree with leaves labelled by variables. If m is non-commutative then it is an ordered tree, and if m is commutative then it is an unordered tree. We let $\mathbf{Tree}(X)$ denote the set of trees whose leaves are labelled by variables in X and $\mathbf{Tree_i}(X)$ denote the subset of such trees with i leaves, which are monomials of degree i.

In this paper we see a non-associative polynomial as a mapping from monomials to K, i.e., an element $f : \mathrm{Tree}(X) \to K$. To avoid possible confusion, let us insist that the notation $f(m)$ refers to the coefficient of the monomial m in the polynomial f, not to be confused with the evaluation of f at a given point. Similarly, a non-commutative associative homogeneous polynomial of degree d is seen as a mapping $f : X^d \to K$.

A (ordered or unordered) ***context*** is a tree with a distinguished leaf labelled by a special symbol called the ***hole*** and written $\Box$. We let $\mathbf{Context}(X)$ denote the set of contexts whose leaves are labelled by variables in X. Given a context c and a tree t we construct a new tree $c[t]$ by substituting the hole of c by t. This operation is defined in both ordered and unordered settings.

Hankel Matrices

Let f be a non-associative polynomial. The ***Hankel matrix*** H_f of f is the matrix whose rows are indexed by contexts and columns by monomials and such that the value of H_f at row c and column t is the coefficient of the monomial $c[t]$ in f.

Arithmetic Circuits

An (arithmetic) ***circuit*** is a directed acyclic graph such that the vertices are of three types:

- input gates: they have in-degree 0 and are labelled by variables in X,
- addition gates: they have arbitrary in-degree, an output value in K, and a weight $w(a) \in K$ on each incoming arc a,
- multiplication gates: they have in-degree 2, and we distinguish between the left child and the right child.

Each gate v in the circuit computes a polynomial f_v which we define by induction.

- An input gate labelled by a variable $x \in X$ computes the polynomial x.
- An addition gate v with n arcs incoming from gates $v_1, \dots, v_n$ and with weights $\alpha_1, \dots, \alpha_n$, computes the polynomial $\alpha_1 f_{v_1} + \cdots + \alpha_n f_{v_n}$.
- A multiplication gate with left child u and right child v computes the polynomial $f_u f_v$.

The circuit itself computes a polynomial given by the sum over all addition gates of the output value times the polynomial computed by the gate. Note that it is slightly unusual that all addition gates contribute to the circuit; one can easily reduce to the classical case where there is a unique output addition gate by adding an extra gate.

To define the size of a circuit we make a syntactic assumption: each arc is either coming from, or going to (but not both), an addition gate. This is a small assumption which can be lifted at the price of a linear blow-up. The ***size*** of a circuit $\mathcal{C}$ is denoted $|\mathcal{C}|$ and defined to be its number of addition gates. Note that this is how the size of ABPs is defined, it will be a convenient definition here since our characterization result captures the exact size of the smallest circuit computing a given polynomial.

Note that the definitions we gave above do not depend on which of the four settings we consider: commutative or non-commutative, associative or non-assocative.

Consider the circuit on the left hand side of Figure 2: it computes the polynomial $7y^2 + 2xy + yx$, which in the commutative setting is equal to $7y^2 + 3xy$.

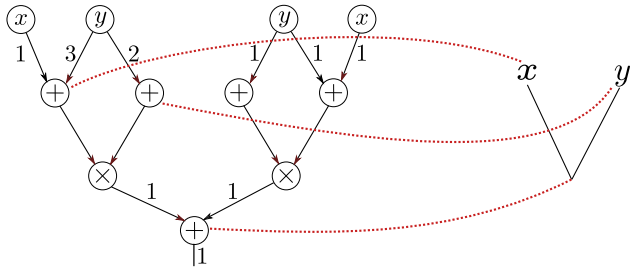

Figure 2 On the left hand side a circuit computing the polynomial $7y^2 + 2xy + yx$, which in the commutative setting is equal to $7y^2 + 3xy$. The only addition gate with a non-zero output value is at the bottom, its output value is 1. On the right hand side the monomial xy, seen as non-associative. The dashed red arrow show one run of the circuit over this monomial.

2.2 The Characterization

This section aims at proving the characterization stated in Theorem 5. It extends Nisan's characterization of non-commutative ABPs to general circuits in the non-associative setting. The result holds for both commutative and non-commutative settings, the proof being the same up to cosmetic changes.

The key step to go from ABPs to general circuits is the following: the polynomial computed by an ABP is the sum over the *paths* of the underlying graph, whereas in a general circuit the sum is over *trees*. We formalize this in the next definition by introducing *runs* of a circuit. The definition is given in the non-commutative setting but easily adapts to the commutative setting as explained in Remark 3.

▶ **Definition 2.** *Let $\mathcal{C}$ be a circuit and $V_\oplus$ denote its set of addition gates. Let $t \in Tree(X)$ be a monomial. A **run of $\mathcal{C}$ over t** is a map ρ from nodes of t to $V_\oplus$ such that*

- **(i)** *A leaf of t with label $x \in X$ is mapped to a gate with a non-zero edge incoming from an input gate labelled by x.*
- **(ii)** *If n is a node of t with left child n_1 and right child n_2, then $\rho(n)$ has a non-zero edge incoming from a multiplication gate with left child $\rho(n_1)$ and right child $\rho(n_2)$.*
- **(iii)** *The root of t is mapped to a gate with non-zero output value.*

*The **value** $val(\rho)$ of ρ is a non-zero element in K defined as the product of the weights of the edges mentioned in items (i) and (ii) together with the output value of $\rho(r)$, r being the root of t.*

We write by a small abuse of notation $\rho : t \to V_\oplus$ for runs of $\mathcal{C}$ over t.

We refer to Figure 2 for an example of a run over the monomial xy. The value of the run is 2.

▶ Remark 3. In the commutative setting we simply replace item (ii) by: "if n is a node of t with children n_1, n_2, then $\rho(n)$ has a non-zero edge incoming from a multiplication gate with children $\rho(n_1), \rho(n_2)$".

A run of $\mathcal{C}$ over a monomial t additively contributes to the coefficient of t in the polynomial computed by $\mathcal{C}$, leading to the following lemma.

▶ **Lemma 4.** *Let $\mathcal{C}$ be a circuit computing the non-associative polynomial $f : Tree(X) \to K$. Then the coefficient $f(t)$ of a monomial $t \in Tree(X)$ in f is equal to*

$$\sum_{\rho : t \to V_\oplus} val(\rho).$$

We may now state and prove our cornerstone result, which holds in both the commutative and non-commutative settings.

▶ **Theorem 5.** *Let $f : Tree(X) \to K$ be a non-associative polynomial, H_f be its Hankel matrix, and $\mathcal{C}$ be a circuit computing f. Then $|\mathcal{C}| \geq rank(H_f)$. Moreover, if f is homogeneous this bound is tight, meaning there exists a circuit $\mathcal{C}$ computing f of size $rank(H_f)$.*

An interesting feature of this theorem is that the upper bound is effective: given a homogenous polynomial one can construct a circuit computing this polynomial of size $\mathrm{rank}(H_f)$.

Here, we only prove the lower bound as the upper bound is not used in the rest of the paper. The proof of the lower bound follows the same lines as Nisan's original proof for non-commutative ABPs [19].

Proof. Let $\mathcal{C}$ be a circuit computing the non-associative polynomial $f : \mathrm{Tree}(X) \to K$. Let $V_\oplus$ denote the set of addition gates of $\mathcal{C}$. To bound the rank of the Hankel matrix H_f by $|\mathcal{C}| = |V_\oplus|$ we show that H_f can be written as the sum of $|V_\oplus|$ matrices each of rank at most 1.

For each $v \in V_\oplus$ we define two circuits which decompose the computations around v. Let $\mathcal{C}_1^v$ be the restriction of $\mathcal{C}$ to descendants of v, and $\mathcal{C}_2^v$ be a copy of $\mathcal{C}$ with just an extra input gate labelled by a fresh variable $\square \notin X$ with a single outgoing edge with weight 1 going to v.

We let $f^v : \mathrm{Tree}(X) \to K$ denote the polynomial computed by $\mathcal{C}_1^v$ and $g^v : \mathrm{Context}(X) \to K$ denote the restriction of the polynomial computed by $\mathcal{C}_2^v$ to $\mathrm{Context}(X) \subseteq \mathrm{Tree}(X \sqcup \{\square\})$.

We show the equality

$$H_f(c,t) = \sum_{v \in V_\oplus} f^v(t) g^v(c).$$

Fix a monomial $t \in \mathrm{Tree}(X)$ and a context $c \in \mathrm{Context}(X)$. We let $n_\square$ denote the leaf of c labelled by $\square$, which is also the root of t and the node to which t is substituted with in $c[t]$. Relying on Lemma 4, we calculate the coefficient $f(c[t])$ of $c[t]$ in f.

$$\begin{aligned} f(c[t]) &= \sum_{\rho : c[t] \to V_\oplus} \mathrm{val}(\rho) = \sum_{v \in V_\oplus} \sum_{\substack{\rho : c[t] \to V_\oplus \\ \rho(n_\square) = v}} \mathrm{val}(\rho) = \sum_{v \in V_\oplus} \sum_{\substack{\rho_1^v : t \to V_\oplus \\ \rho_1^v(n_\square) = v}} \sum_{\substack{\rho_2^v : c \to V_\oplus \\ \rho_2^v(n_\square) = v}} \mathrm{val}(\rho_1^v)\mathrm{val}(\rho_2^v) \\ &= \sum_{v \in V_\oplus} \sum_{\substack{\rho_1^v : t \to V_\oplus \\ \rho_1^v(n_\square) = v}} \mathrm{val}(\rho_1^v) \sum_{\substack{\rho_2^v : c \to V_\oplus \\ \rho_2^v(n_\square) = v}} \mathrm{val}(\rho_2^v) = \sum_{v \in V_\oplus} f^v(t) g^v(c). \end{aligned}$$

Let $M_v \in K^{\mathrm{Tree}(X) \times \mathrm{Context}(X)}$ be the matrix given by $M_v(t,c) = f^v(t)g^v(c)$: its rank is at most one as M_v is the product of a column vector by a row vector. The previous equality reads in matrix form $H_f = \sum_{v \in V_\oplus} M_v$. Hence, we obtain the announced lower bound using rank subadditivity:

$$\mathrm{rank}\,(H_f) = \mathrm{rank}\left(\sum_{v \in V_\oplus} M_v\right) \leq \sum_{v \in V_\oplus} \mathrm{rank}\,(M_v) \leq |V_\oplus| = |\mathcal{C}|. \qquad \blacktriangleleft$$

The remainder of this paper consists in applying Theorem 5 to obtain lower bounds in various cases. To this end we need a better understanding of the Hankel matrix: in Section 3 we introduce a few concepts and develop decomposition theorems for the Hankel matrix.

3 Decomposing the Hankel Matrix

Our decomposition of the Hankel matrix relies on the notion of parse trees and partial derivative matrices, which we formally introduce now.

3.1 Parse Trees

With any monomial $t \in \mathrm{Tree}(X)$ we associate its ***shape*** $\mathrm{shape}(t) \in \mathrm{Tree}$ as the tree obtained from t by removing the labels at the leaves.

▶ **Definition 6.** *Let $\mathcal{C}$ be a circuit computing a non-commutative non-associative polynomial f. A **parse tree** of $\mathcal{C}$ is any shape $s \in Tree$ for which there exists a monomial $t \in Tree(X)$ whose coefficient in f is non-zero and such that $s = shape(t)$. We let $PT(\mathcal{C}) = \{shape(t) \mid f(t)\ non\text{-}zero\}$.*

3.2 Partial Derivative Matrices

We now introduce a well known tool for proving circuit lower bounds, namely, partial derivative matrices. For $A \subseteq [d]$ of size i , $u \in X^{d-i}$, and $v \in X^i$, we define the monomial $u \otimes_A v \in X^d$: it is obtained by interleaving u and v with u taking the positions indexed by $[d] \setminus A$ and v the positions indexed by A. For instance $x_1x_2 \otimes_{\{2,4\}} y_1y_2 = x_1y_1x_2y_2$.

▶ **Definition 7.** *Let f be a homogeneous non-commutative associative polynomial. Let $A \subseteq [d]$ be a set of positions of size i.*

*The **partial derivative matrix** $M_A(f)$ of f with respect to A is defined as follows: the rows are indexed by $u \in X^{d-i}$ and the columns by $v \in X^i$, and the value of $M_A(f)(u,v)$ is the coefficient of the monomial $u \otimes_A v$ in f.*

▶ **Example 8.** Let $f = xyxy + 3xxyy + 2xxxy + 5yyyy$ and $A = \{2,4\}$. Then $M_A(f)$ is given below.

	$\_x\_x$	$\_x\_y$	$\_y\_x$	$\_y\_y$
$x\_x\_$	0	2	0	1
$y\_x\_$	0	0	0	0
$x\_y\_$	0	3	0	0
$y\_y\_$	0	0	0	5

We define a distance $\mathrm{dist} : \mathcal{P}([d]) \times \mathcal{P}([d]) \to \mathbb{N}$ on subsets of $[d]$ by letting $\mathrm{dist}(A,B)$ be the minimal number of additions and deletions of elements of $[d]$ to go from A to B, assuming that complementing is for free. Formally, $\mathrm{dist}(A,B) = \min\{|\Delta(A,B)|, |\Delta(A^{\mathrm{C}},B)|\}$, where $\Delta(A,B) = (A \setminus B) \cup (B \setminus A)$ is the symmetric difference between A and B.

The following lemma (see e.g., [17]) informally says that, if A and B are close to each other, then the ranks of the corresponding partial derivative matrices are close to each other as well.

▶ **Lemma 9.** *Let f be a homogeneous non-commutative associative polynomial of degree d with n variables. Then, for any subsets $A, B \subseteq [d]$, $\mathrm{rank}(M_A(f)) \leq n^{dist(A,B)} \mathrm{rank}(M_B(f))$.*

At this point, we have the material in hands to describe a precise characterization of the size of the smallest Unique Parse Tree circuit which computes a given polynomial. We take this short detour before moving on to our core lower bound results.

3.3 Characterization of smallest Unique Parse Tree Circuit

Unique Parse Tree (UPT) circuits are non-commutative associative circuits with a unique parse tree. They were first introduced in [16]. Our techniques allow a slight improvement and a better understanding of their results. We obtain a small improvement since the original result requires a normal form which can lead to an exponential blow-up.

Given a shape $s \in \mathrm{Tree}$ of size d, i.e., with d leaves and a node v of s, we let s_v denote the subtree of s rooted in v, and $I_v \subseteq [d]$ denote the interval of positions of the leaves of s_v in s. We say that $s' \in \mathrm{Tree}$ is a subshape of s if $s' = s_v$ for some v, and that I is spanned by s if $I = I_v$ for some v.

Let $f : X^d \to K$ be a homogeneous non-commutative associative polynomial of degree d, let $s \in \mathrm{Tree}$ be a shape of size d, and let s' be a subshape of s such that $v_1, \ldots, v_p$ are all the nodes v of s such that $s' = s_v$. We define

$$M_{s'} = \begin{bmatrix} M_{I_{v_1}}(f) \\ M_{I_{v_2}}(f) \\ \vdots \\ M_{I_{v_p}}(f) \end{bmatrix}.$$

▶ **Theorem 10.** *Let $f : X^d \to K$ be a homogeneous non-commutative associative polynomial and let $s \in Tree$ be a shape of size d. Then the smallest UPT circuit with shape s computing f has size exactly*

$$\sum_{s' \text{ subshape of } s} rank(M_{s'}) .$$

Proof. Let $\mathcal{C}$ be a UPT circuit with shape s computing f. We let $\tilde{f}$ denote the non-associative polynomial computed by $\mathcal{C}$. Since $\mathcal{C}$ is UPT with shape s, $\tilde{f}$ is the *unique* non-associative polynomial which is non-zero only on trees with shape s and projects to f, i.e., $\tilde{f}(t) = f(u)$ if $\text{shape}(t) = s$ and t is labelled by u, and $\tilde{f}(t) = 0$ otherwise.

In particular, the size of the smallest UPT circuit with shape s computing f is the same as the size of the smallest circuit computing $\tilde{f}$, which thanks to Theorem 5 is equal to the rank of the Hankel matrix $H_{\tilde{f}}$.

The Hankel matrix of $\tilde{f}$ may be non-zero only on columns indexed by trees whose shapes s' are subshapes of s, and on such columns, non-zero values are on rows corresponding to a context obtained from s by replacing an occurrence of s' by $\Box$. The corresponding blocks are precisely the matrices $M_{s'}$, and are placed in a diagonal fashion, hence the lower bound. ◀

Theorem 10 can be applied to concrete polynomials, for instance to the permanent of degree d.

▶ **Corollary 11.** *Let $s \in Tree$ be a shape. The smallest UPT circuit with shape s computing the permanent has size*

$$\sum_{v \text{ node of } s} \binom{d}{|I_v|},$$

where I_v is the set of leaves in the subtree rooted at v in s. In particular, this is always larger than $\binom{d}{d/3}$.

Applied to s being a left-comb, Corollary 11 yields that the smallest ABP computing the permanent has size $2^d + d$. Applied to s being a complete binary tree of depth $k = \log d$, the size of the smallest UPT is $\Theta\left(\frac{2^d}{d}\right)$, showing that this circuit is more efficient than any ABP.

3.4 General Roadmap

We now get to the technical core of the paper where we establish generic lower bounds theorems through a decomposition of the Hankel matrix, that we will later instantiate in Section 4 to concrete classes of circuits. We first restrict ourselves to the non-commutative setting. Our first decomposition, Theorem 12, seems to capture mostly previously known techniques. However, the second, more powerful decomposition, Theorem 13, takes advantage of the global shape of the Hankel matrix. Doing so allows to go beyond previous results only hinging around considering partial derivatives matrices which only turn out to be isolate slices of the Hankel matrix.

We later explain in Section 3.6 how to extend the study to the commutative case.

Let f be a (commutative or non-commutative) polynomial for which we want to prove lower bounds. Consider a circuit $\mathcal{C}$ which computes f, and let $\tilde{f}$ be the non-associative polynomial computed by $\mathcal{C}$. Our aim is, following Theorem 5, to give lower bounds on the rank of the Hankel matrix $H_{\tilde{f}}$. We know that the $\tilde{f}$ and f are equal up to associativity, which provides linear relations among the coefficients of $H_{\tilde{f}}$.

The bulk of the technical work is to reorganize the rows and columns of $H_{\tilde{f}}$ in order to decompose it into blocks which may be identified as partial derivative matrices with respect to some subsets $A_1, A_2, \cdots \subseteq [d]$, of some associative polynomials which depend on $\tilde{f}$ and sum to f. The number and choice of these subsets depend on the parse trees of the circuit $\mathcal{C}$.

Now, assume there exists a subset $A \subseteq [d]$ which is at distance at most δ to each A_i. Losing a factor of n^δ on the rank through the use of Lemma 9 we reduce the aforementioned blocks of $H_{\tilde{f}}$ to partial derivatives with respect to A. Such matrices can then be summed to recover the partial derivative matrix of f with respect to A, yielding in the lower bound a (dominating) factor of $\operatorname{rank}(M_A(f))$.

3.5 Generic Lower Bounds in the Non-commutative Setting

Following the general roadmap described above, we obtain a first generic lower bound result.

▶ **Theorem 12.** *Let $f : X^d \to K$ be a non-commutative homogeneous polynomial computed by a circuit $\mathcal{C}$. Let $A \subseteq [d]$ and $\delta \in \mathbb{N}$ such that all parse trees of $\mathcal{C}$ span an interval at distance at most δ from A. Then $\mathcal{C}$ has size at least $\operatorname{rank}(M_A(f))\, n^{-\delta} |PT(\mathcal{C})|^{-1}$.*

The crux to prove Theorem 12 is to identify for each parse tree s of $\mathcal{C}$ a block in $H_{\tilde{f}}$ containing the partial derivative matrix $M_{I(s)}(f_s)$ where f_s is the polynomial corresponding to the contribution of the parse tree s in the computation of f and $I(s)$ is an interval spanned by s.

However, we do not consider in this analysis how these blocks are located relative to each other. A more careful analysis of $H_{\tilde{f}}$ consists in grouping together all parse trees that lead to the same spanned interval. Aligning and then summing these blocks we remove the dependence in $|\mathrm{PT}(\mathcal{C})|$ and instead use d^2 which is the total number of possibly spanned intervals of $[d]$. This yields Theorem 13.

▶ **Theorem 13.** *Let f be a non-commutative homogeneous polynomial computed by a circuit $\mathcal{C}$. Let $A \subseteq [d]$ and $\delta \in \mathbb{N}$ such that all parse trees of $\mathcal{C}$ span an interval at distance at most δ from A. Then $\mathcal{C}$ has size at least $\operatorname{rank}(M_A(f))\, n^{-\delta} d^{-2}$.*

As we shall see in Section 4 the lower bounds we obtain using Theorem 12 match known results, while using Theorem 13 yields substantial improvements.

3.6 Extending to the Commutative Setting

We explain how to extend the notions of parse trees and the generic lower bound theorems to the commutative setting.

Let $X = X_1 \sqcup X_2 \sqcup \cdots \sqcup X_d$ be a partition of the variable set X. A monomial is ***set-multilinear*** with respect to the partition if it is the product of exactly one variable from each set X_i, and a polynomial is set-multilinear if all its monomials are.

The permanent and the determinant of degree d are set-multilinear with respect to the partition $X = X_1 \sqcup X_2 \sqcup \cdots \sqcup X_d$ where $X_i = \{x_{i,j}, j \in [d]\}$. The iterated matrix multiplication polynomial is another example of an important and well-studied set-multilinear polynomial. The partial derivative matrix also make sense in the realm of set-multilinear polynomials.

▶ **Definition 14.** *Let $X = X_1 \sqcup X_2 \sqcup \cdots \sqcup X_d$, f be a set-multilinear polynomial of degree d, and $A \subseteq [d]$ be a set of indices. The **partial derivative matrix $M_A(f)$** of f with respect to A is defined as follows: the rows are indexed by set-multilinear monomials g with respect to the partition $\bigsqcup_{i \notin A} X_i$ and the columns are indexed by set-multilinear monomials h with respect to the partition $\bigsqcup_{i \in A} X_i$. The value of $M_A(f)(g,h)$ is the coefficient of the monomial $g \cdot h$ in f.*

The notion of shape was defined by [3], and it slightly differs from the non-commutative case because we need to keep track of the indices of the variable sets given by the partition from which the variables belong. More precisely, given a partition of $X = X_1 \sqcup X_2 \sqcup \cdots \sqcup X_d$, we associate to any monomial $t \in \mathrm{Tree}(X)$ of degree d its ***shape*** $\mathrm{shape}(t) \in \mathrm{Tree}([d])$ defined as the tree obtained from t by replacing each label by its index in the partition. We let $\mathcal{T}_d \subseteq \mathrm{Tree}([d])$ denote the set of trees such that all elements of $[d]$ appear as a label of a leaf.

Let $\mathcal{C}$ be a commutative circuit. We let $\tilde{f}$ denote the commutative non-associative polynomial computed by C when it is seen as non-associative. A ***parse tree*** of $\mathcal{C}$ is any shape $s \in \mathcal{T}_d$ for which there exists a monomial $t \in \mathrm{Tree}(X)$ whose coefficient in $\tilde{f}$ is non-zero and such that $s = \mathrm{shape}(t)$. Hence, we let $\mathrm{PT}(\mathcal{C}) = \{\mathrm{shape}(t) \mid \tilde{f}(t) \text{ non-zero}\} \cap \mathcal{T}_d$.

Given a shape $s \in \mathrm{Tree}([d])$ with d leaves and a node v of s, we let s_v denote the subtree rooted at v and $A_v \subseteq [d]$ denote the set of labels appearing on the leaves of s_v.

Following the same roadmap as in the non-commutative setting we obtain the following counterpart of Theorem 12. We assume that the set of variables is partitioned into d parts of equal size n (this is a natural setting for polynomials such as the determinant, the permanent or the iterated matrix multiplication). In particular, it means that the polynomials we consider are of degree d and over nd variables.

▶ **Theorem 15.** *Let f be a set-multilinear polynomial computed by a circuit $\mathcal{C}$. Let $A \subseteq [d]$ and $\delta \in \mathbb{N}$ such that all parse trees of $\mathcal{C}$ span a subset at distance at most δ from A. Then $\mathcal{C}$ has size at least $\mathrm{rank}(M_A(f))\, n^{-\delta} |PT(\mathcal{C})|^{-1}$.*

A notable difference with the non-commutative setting is that now parse trees no longer span intervals of $[d]$ but subsets of $[d]$. As a consequence, the technique used to prove Theorem 13 groups together blocks corresponding to the same *subset* of $[d]$ and therefore the multiplicative factor is now 2^{-d} as there are 2^d such subsets.

▶ **Theorem 16.** *Let f be a set-multilinear polynomial computed by a circuit $\mathcal{C}$. Let $A \subseteq [d]$ and $\delta \in \mathbb{N}$ such that all parse trees of $\mathcal{C}$ span a subset at distance at most δ from A. Then $\mathcal{C}$ has size at least $\mathrm{rank}(M_A(f))\, n^{-\delta} 2^{-d}$.*

While in the non-commutative setting, Theorem 13 strengthens Theorem 12 (when d^2 is small), this is no longer the case in the commutative setting. Indeed, the maximal number of commutative parse trees being roughly $d!$ (the exact asymptotic is $\frac{\sqrt{2-\sqrt{2}}d^{d-1}}{e^d(\sqrt{2}-1)^{d+1}}$, see e.g., `https://oeis.org/A036774`), Theorem 15 and Theorem 16 are incomparable.

4 Applications

In this section we instantiate our generic lower bound theorems on concrete classes of circuits. We first show how the weaker version (Theorem 12) yields the best lower bounds to date for skew and small non-skew depth circuits. Extending these ideas we obtain exponential lower bounds for $\left(\frac{1}{2} - \varepsilon\right)$-unbalanced circuits, an extension of skew circuits which are just slightly unbalanced. We also adapt the proof to ε-balanced circuits, which are slightly balanced, then move on to our main results, which concern circuits with many parse trees.

High-ranked polynomials

The lower bounds we state below hold for any polynomial whose partial derivative matrices with respect to either a fixed subset A or all subsets have large rank. Such polynomials exist for all fields in both the commutative and non-commutative settings, and can be explicitly

constructed. For instance the so-called Nisan-Wigderson polynomial given in [14] (inspired by the notion of designs by Nisan and Wigderson [20]) has this property. It are given by

$$NW_{n,d} = \sum_{\substack{h \in \mathbb{F}_n[z] \\ \deg(h) \leq d/2}} \prod_{i=1}^{d} x_{i,h(i)},$$

where $\mathbb{F}_n[z]$ denotes univariate polynomials with coefficients in the finite field of prime order n. The fact that there exists a unique polynomial $h \in \mathbb{F}_n[z]$ of degree at most $d/2$ which takes $d/2$ given values at $d/2$ given positions exactly implies that the partial derivative matrix of $NW_{n,d}$ with respect to any $A \subseteq [d]$ of size $d/2$ is a permutation matrix. This is then easily extended to any $A \subseteq [d]$.

4.1 Skew, Slightly Unbalanced, Slightly Balanced and Small Non-Skew Depth Circuits

We show how using Theorem 12 yields exponential lower bounds for four classes of circuits in the non-commutative setting.

Skew Circuits

A circuit $\mathcal{C}$ is ***skew*** if all its parse trees are skew, meaning that each node has at least one of its children which is a leaf. As a direct application of Theorem 12, we obtain the following result.

▶ **Theorem 17.** *Let f be a homogeneous non-commutative polynomial such that $M_{[d/4+1,3d/4]}(f)$ has full rank $n^{d/2}$. Then any skew circuit computing f has size at least $2^{-d}n^{d/4}$.*

Slightly unbalanced circuits

A circuit $\mathcal{C}$ computing a homogeneous non-commutative polynomial of degree d is said to be ***α-unbalanced*** if every multiplication gate has at least one of its children which computes a polynomial of degree at most αd.

▶ **Theorem 18.** *Let f be a homogeneous non-commutative polynomial such that $M_{[d/4+1,3d/4]}(f)$ has full rank $n^{d/2}$. Then any $\left(\frac{1}{2}-\varepsilon\right)$-unbalanced circuit computing f has size at least $4^{-d}n^{\varepsilon d}$.*

This result improves over a previously known exponential lower bound on $\left(\frac{1}{5}\right)$-unbalanced circuits [17].

Slightly balanced circuits

A circuit $\mathcal{C}$ computing a homogeneous non-commutative polynomial of degree d is said to be ***α-balanced*** if every multiplication gate which computes a polynomial of degree k has both of its children which compute a polynomial of degree at least αk.

▶ **Theorem 19.** *Let f be a homogeneous non-commutative polynomial such that $M_{[1,d/2]}(f)$ has full rank $n^{d/2}$. Then any ε-balanced circuit computing f has size at least $4^{-d}n^{\varepsilon d}$.*

Small Non-skew Depth Circuits

A circuit $\mathcal{C}$ has ***non-skew depth k*** if all its parse trees are such that each path from the root to a leaf goes through at most k non-skew nodes, i.e., nodes for which the two children are inner nodes.

We obtain an alternative proof of the exponential lower bound of [17] on non-skew depth k circuits as an application of Theorem 12. In the statement below A refers to an explicit subset of $[d]$ that we do not define here.

▶ **Theorem 20.** *Let f be a homogeneous non-commutative polynomial of degree $d = 12kp$ such that $M_A(f)$ has full rank $n^{d/2}$. Then any circuit of non-skew depth k computing f has size at least $4^{-d}n^{p/3} = 4^{-d}n^{d/36k}$.*

4.2 Circuits with Many Parse Trees

We focus on ***k-PT circuits*** which are circuits with at most k different parse trees.

The Non-commutative Setting

Lagarde, Limaye, and Srinivasan [15] obtained a superpolynomial lower bound for superpolynomial k (up to $k = 2^{d^{\frac{1}{3}-\varepsilon}}$). We first show how to obtain the same result using Theorem 12.

For $s \in \mathrm{Tree}_d$ and $A \subseteq [d]$, we define $\mathrm{dist}(A, s) = \min\{\mathrm{dist}(A, I) \mid I \text{ spanned by } s\}$. The following lemma is a subtle probabilistic analysis ensuring the existence of a subset which is close enough to all k parse trees.

▶ **Lemma 21** (adapted from Claim 15 in [15])**.** *Let $s \in Tree_d$ be a shape with d leaves, and $\delta \leq \sqrt{d}$. Then*

$$\Pr_{A\sim\mathcal{U}\left(\binom{[d]}{d/2}\right)}\left[dist(A, s) > d/2 - \delta\right] \leq 2^{-\alpha d/\delta^2},$$

where α is some positive constant and $\mathcal{U}\left(\binom{[d]}{d/2}\right)$ the uniform distribution of subsets of d of size $d/2$.

Proof sketch. Following [15], we find a sequence of $r = \Omega(d/\delta^2)$ nodes of s which all span distant enough subtrees. We then obtain the bound by splitting the previous event into r essentially independent events. ◀

From there, the lower bound is obtained using Theorem 12 and a fine tuning of the parameters.

▶ **Theorem 22.** *Let f be a homogeneous non-commutative polynomial such that for all $A \subseteq [d]$ $M_A(f)$ has full rank. Let $k = 2^{d^{1/3-\varepsilon}}$ and $\varepsilon > 0$. Then for large enough d, any k-PT circuit computing f has size at least $2^{d^{1/3}(\log n - d^{-\varepsilon})}$.*

Proof. Let $\mathcal{C}$ be a k-PT circuit computing f, and $\delta = d^{1/3} \leq \sqrt{d}$. We first show that there exists a subset $A \subseteq [d]$ which is close to all parse trees in $\mathcal{C}$. Indeed, a union bound and Lemma 21 yield

$$\begin{aligned}\Pr_{A\sim\mathcal{U}\left(\binom{[d]}{d/2}\right)}\left[\exists s \in \mathrm{PT}(\mathcal{C}), \mathrm{dist}(A, s) > d/2 - \delta\right] &\leq \sum_{s\in\mathrm{PT}(\mathcal{C})} \Pr_{A\sim\mathcal{U}\left(\binom{[d]}{d/2}\right)}\left[\mathrm{dist}(A, s) > d/2 - \delta\right] \\ &\leq k2^{-\alpha d/\delta^2} = 2^{d^{1/3-\varepsilon} - \alpha d^{1/3}} < 1,\end{aligned}$$

for large enough d. We now pick a subset $A \subseteq [d]$ of size $d/2$ such that for all $s \in \mathrm{PT}(\mathcal{C}), \mathrm{dist}(A, s) \leq d/2 - \delta$, that is, any $s \in \mathrm{PT}(\mathcal{C})$ spans an interval $I(s)$ at distance at most $d/2 - \delta$ from A. Finally, we apply Theorem 12 to obtain

$$|\mathcal{C}| \geq \mathrm{rank}(M_A(f))\, n^{-(d/2-\delta)} k^{-1} = n^{d/2} n^{-(d/2-d^{1/3})} 2^{-d^{1/3-\varepsilon}} = 2^{d^{1/3}(\log n - d^{-\varepsilon})}.$$ ◀

We may improve the previous bound by applying Theorem 13 instead of Theorem 12. Indeed, since Theorem 13 gets rid of the factor k^{-1} in the lower bound, picking a much smaller δ ($\delta = d^{\varepsilon/3}$ instead of $d^{1/3}$) still leads to a superpolynomial lower bound, while allowing for more parse trees.

▶ **Theorem 23.** *Let f be a homogeneous non-commutative polynomial such that for all $A \subseteq [d]$ $M_A(f)$ has full rank. Let $k = 2^{d^{1-\varepsilon}}$ and $\varepsilon > 0$. Then for large enough d, any k-PT circuit computing f has size at least $n^{d^{\varepsilon/4}} d^{-2}$.*

The bound $2^{d^{1-\varepsilon}}$ on the number of parse trees is to be compared to the total number of shapes of size d which is $\frac{1}{d}\binom{2(d-1)}{d-1} \sim \frac{4^d}{d^{3/2}\sqrt{\pi}} \leq 2^{2d}$. As explained in the introduction this means that we obtain superpolynomial lower bounds for any class of circuits which has a small defect in the exponent of the total number of parse trees.

The Commutative Setting

Arvind and Raja [3] showed a superpolynomial lower bound for sublinear k (up to $k = d^{1/2-\varepsilon}$). We improve this to superpolynomial k (up to $k = 2^{d^{1-\varepsilon}}$).

Indeed, in the commutative setting, Lemma 21 holds as such (with a shape being an element of $\mathcal{T}_d$, that is, a commutative parse tree of size d). However, the generic lower bound theorems, namely Theorem 15 and Theorem 16, are not exactly the same, so we obtain slightly different results. In particular, the two results we obtain are incomparable. Applying Theorem 15 leads to Theorem 24, whereas Theorem 16 leads to Theorem 25.

▶ **Theorem 24.** *Let f be a set-multilinear commutative polynomial such that for all $A \subseteq [d]$, the matrix $M_A(f)$ has full rank. Let $k = 2^{d^{1/3-\varepsilon}}$ and $\varepsilon > 0$. Then for large enough d, any k-PT circuit computing f has size at least $2^{d^{1/3}(\log n - d^{-\varepsilon})}$.*

▶ **Theorem 25.** *Let f be a set-multilinear commutative polynomial such that for all $A \subseteq [d]$, the matrix $M_A(f)$ has full rank. Let $k = 2^{d^{1-\varepsilon}}$ and $\varepsilon > 0$. Then for large enough d, any k-PT circuit computing f has size at least $n^{d^{\varepsilon/4}} 2^{-d}$.*

References

1. Manindra Agrawal, Rohit Gurjar, Arpita Korwar, and Nitin Saxena. Hitting-sets for ROABP and sum of set-multilinear circuits. *SIAM Journal on Computing*, 44(3):669–697, 2015.
2. Eric Allender, Jia Jiao, Meena Mahajan, and V. Vinay. Non-commutative arithmetic circuits: Depth reduction and size lower bounds. *Theoretical Computer Science*, 209(1-2):47–86, 1998.
3. Vikraman Arvind and S. Raja. Some lower bound results for set-multilinear arithmetic computations. *Chicago Journal of Theoretical Computer Science*, 2016, 2016.
4. Walter Baur and Volker Strassen. The complexity of partial derivatives. *Theoretical Computer Science*, 22:317–330, 1983.
5. Symeon Bozapalidis and Olympia Louscou-Bozapalidou. The rank of a formal tree power series. *Theoretical Computer Science*, 27:211–215, 1983.
6. Marco L. Carmosino, Russell Impagliazzo, Shachar Lovett, and Ivan Mihajlin. Hardness amplification for non-commutative arithmetic circuits. In *Proceedings of the 33rd Computational Complexity Conference (CCC 2018)*, volume 102 of *LIPIcs*, pages 12:1–12:16. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2018.

7 Zeev Dvir, Guillaume Malod, Sylvain Perifel, and Amir Yehudayoff. Separating multilinear branching programs and formulas. In *Proceedings of the 44th Symposium on Theory of Computing Conference (STOC 2012)*, pages 615–624. ACM, 2012.

8 Nathanaël Fijalkow, Guillaume Lagarde, and Pierre Ohlmann. Tight bounds using hankel matrix for arithmetic circuits with unique parse trees. *Electronic Colloquium on Computational Complexity (ECCC)*, 25:38, 2018. URL: `https://eccc.weizmann.ac.il/report/2018/038`.

9 Michel Fliess. Matrices de Hankel. *Journal de Mathématiques Pures et Appliquées*, 53:197–222, 1974.

10 Michael A. Forbes, Ramprasad Saptharishi, and Amir Shpilka. Hitting sets for multilinear read-once algebraic branching programs, in any order. In *Proceedings of the 46th Symposium on Theory of Computing, (STOC 2014)*, pages 867–875. ACM, 2014.

11 Rohit Gurjar, Arpita Korwar, Nitin Saxena, and Thomas Thierauf. Deterministic identity testing for sum of read-once oblivious arithmetic branching programs. *Computational Complexity*, 26(4):835–880, 2017.

12 Pavel Hrubeš, Avi Wigderson, and Amir Yehudayoff. Non-commutative circuits and the sum-of-squares problem. *Journal of the American Mathematical Society*, 24(3):871–898, 2011.

13 Valentine Kabanets and Russell Impagliazzo. Derandomizing polynomial identity tests means proving circuit lower bounds. In *Proceedings of the 35th Annual ACM Symposium on Theory of Computing (STOC 2003)*, pages 355–364. ACM, 2003.

14 Neeraj Kayal, Chandan Saha, and Ramprasad Saptharishi. A super-polynomial lower bound for regular arithmetic formulas. In *Proceedings of the 46th Symposium on Theory of Computing, (STOC 2014)*, pages 146–153. ACM, 2014.

15 Guillaume Lagarde, Nutan Limaye, and Srikanth Srinivasan. Lower bounds and PIT for non-commutative arithmetic circuits with restricted parse trees. *Computational Complexity*, pages 1–72, 2018. https://doi.org/10.1007/s00037-018-0171-9.

16 Guillaume Lagarde, Guillaume Malod, and Sylvain Perifel. Non-commutative computations: lower bounds and polynomial identity testing. *Electronic Colloquium on Computational Complexity (ECCC)*, 23:94, 2016.

17 Nutan Limaye, Guillaume Malod, and Srikanth Srinivasan. Lower bounds for non-commutative skew circuits. *Theory of Computing*, 12(1):1–38, 2016.

18 Guillaume Malod and Natacha Portier. Characterizing Valiant's algebraic complexity classes. *Journal of Complexity*, 24(1):16–38, 2008.

19 Noam Nisan. Lower bounds for non-commutative computation (extended abstract). In *Proceedings of the 23rd Symposium on Theory of Computing (STOC 1991)*, pages 410–418. ACM, 1991.

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

21 C. Ramya and B. V. Raghavendra Rao. Lower bounds for special cases of syntactic multilinear abps. In *Proceedings of the 24th International Computing and Combinatorics Conference (COCOON 2018)*, volume 10976 of *Lecture Notes in Computer Science*, pages 701–712. Springer, 2018.

22 Ran Raz and Amir Shpilka. Deterministic polynomial identity testing in non-commutative models. *Computational Complexity*, 14(1):1–19, 2005.

23 Ramprasad Saptharishi and Anamay Tengse. Quasi-polynomial hitting sets for circuits with restricted parse trees. *Electronic Colloquium on Computational Complexity (ECCC)*, 24:135, 2017.

24 Seinosuke Toda. Classes of arithmetic circuits capturing the complexity of computing the determinant. *IEICE Trans. Inf. Systems*, E75-D(1):116–124, 1992.

25 Leslie G. Valiant. The complexity of computing the permanent. *Theoretical Computer Science*, 8:189–201, 1979.