Quasi-polynomial Hitting Sets for Circuits with Restricted Parse Trees

Ramprasad Saptharishi[*] Anamay Tengse[†]

August 19, 2021

Abstract

We study the class of non-commutative *Unambiguous circuits or Unique-Parse-Tree (UPT) circuits*, and a related model of *Few-Parse-Trees (FewPT)* circuits (which were recently introduced by Lagarde, Malod and Perifel [LMP19] and Lagarde, Limaye and Srinivasan [LLS19]) and give the following constructions:

- An explicit hitting set of *quasipolynomial* size for UPT circuits,
- An explicit hitting set of *quasipolynomial* size for FewPT circuits (circuits with constantly many parse tree shapes),
- An explicit hitting set of *polynomial* size for UPT circuits (of known parse tree shape), when a parameter of *preimage-width* is bounded by a constant.

The above three results are extensions of the results of [AGKS15], [GKST17] and [GKS17] to the setting of UPT circuits, and hence also generalize their results in the commutative world from *read-once oblivious algebraic branching programs (ROABPs)* to *UPT-set-multilinear* circuits.

The main idea is to study *shufflings* of non-commutative polynomials, which can then be used to prove suitable depth reduction results for UPT circuits and thereby allow a careful translation of the ideas in [AGKS15], [GKST17] and [GKS17].

1 Introduction

The field of algebraic complexity deals with classifying multivariate polynomials based on their hardness. Typically, the complexity of a polynomial is measured by the size of the smallest circuit computing it (an arithmetic circuit is a directed acyclic graph made up of internal nodes that are labeled with $+$ or $\times$ and leaves labelled with variables or constants from the field). The central question in this field is to construct

[*]ramprasad@tifr.res.in. School of Technology and Computer Science, Tata Institute of Fundamental Research, Mumbai, India. Research supported by Ramanujan Fellowship of DST.

[†]anamay.tengse@gmail.com. School of Technology and Computer Science, Tata Institute of Fundamental Research, Mumbai, India. Research supported by a fellowship of the DAE, Government of India.

an explicit family of polynomials ($\{\mathrm{Perm}_n\}$ is the top candidate) that requires large arithmetic circuits to compute it. This is also called the "VP vs VNP" question (named after Valiant [Val79]), and thought of as an algebraic analogue of the "P vs NP" question.

So far, the best lower bound we have for general arithmetic circuits computing an n-variate degree d polynomial is a barely super-linear $\Omega(n \log d)$ lower bound by Baur and Strassen [BS83]. Recent research has focused on proving lower bounds for restricted classes of circuits, either by bounding the depth of such circuits or by focusing on other syntactic restrictions. One such syntactic restriction is to consider *non-commutative circuits*, where we assume that the underlying variables $x_1, \ldots, x_n$ do not commute. In the non-commutative model, there is an inherent order in which elements are multiplied and this adds restrictions on the way monomials can be computed ($xy \neq yx$ here and hence $x^2 + 2xy + y^2 \neq (x+y)^2 = x^2 + xy + yx + y^2$). It is therefore natural to expect that it should be easier to prove lower bounds in this model.

Nisan [Nis91] introduced the non-commutative model, specifically the non-commutative algebraic branching programs (ABP). In his seminal paper, he showed that the non-commutative versions of the determinant and permanent polynomials (among others) require exponential sized non-commutative ABPs to compute them. In fact, using his technique, one could even reconstruct the smallest non-commutative ABP given just oracle access to that polynomial (cf. [KS06])! Although we have exponential lower bounds for non-commutative ABPs, we do not have any non-trivial lower bounds for non-commutative circuits. Hrubeš, Wigderson and Yehudayoff [HWY10] presented an approach via *sum-of-squares* lower bounds but we do not have any non-trivial lower bounds for the class of general non-commutative circuits.

Limaye, Malod and Srinivasan [LMS16] extended Nisan's lower bound to non-commutative skew circuits, which are circuits where every multiplication gate has at most one child that is a non-leaf. Lagarde, Malod and Perifel [LMP19] initiated the study of non-commutative *unambiguous circuits*, or *Unique Parse Tree (UPT)* circuits. These circuits, and generalizations are the main models of study in this paper.

Arvind and Raja [AR16] also studied lower bounds for various subclasses of commutative set-multilinear circuits. Some of the models they study also include analogues of UPT and FewPT circuits. They also proved lower bounds for UPT and FewPT set-multilinear circuits, and also for other subclasses of set-multilinear circuits called *narrow* set-multilinear circuits, *interval* set-multilinear circuits, the latter of which assumes the sum-of-squares conjecture of Hrubeš, Wigderson and Yehudayoff [HWY10].

1.1 The model of study

A parse tree of a circuit is obtained by starting at the root, and at every + gate choosing exactly one child, and at every × gate choosing all its children (formally defined in Definition 2.1). Informally, a parse tree of a circuit is basically a *certificate* of computation of a monomial in a circuit. Lagarde, Malod and Perifel [LMP19] introduced a subclass of non-commutative circuits called *Unique Parse Tree (UPT) circuits* or *unambiguous circuits* where all parse trees of the circuit have the same shape (formally defined in Definition 2.2). The class of non-commutative UPT circuits subsumes the class of non-commutative ABPs as any ABP can be expressed as a left-skew circuit. A related model of *set-depth-Δ formulas* was studied by Agrawal, Saha and Saxena [ASS13] that is a subclass of UPT circuits where the underlying parse trees are extremely regular[1].

Lagarde, Malod and Perifel [LMP19] extended the techniques of Nisan [Nis91] to give exponential lower bounds for UPT circuits. Subsequently, Lagarde, Limaye and Srinivasan [LLS19] extended the lower bounds to the class of circuits with parse trees of not-too-many shapes (at most $2^{o(n)}$ shapes).

In Figure 1, (a) is an example of a UPT circuit with (b) being the underlying parse tree shape; (c) is an example of a circuit with two distinct parse tree shapes.

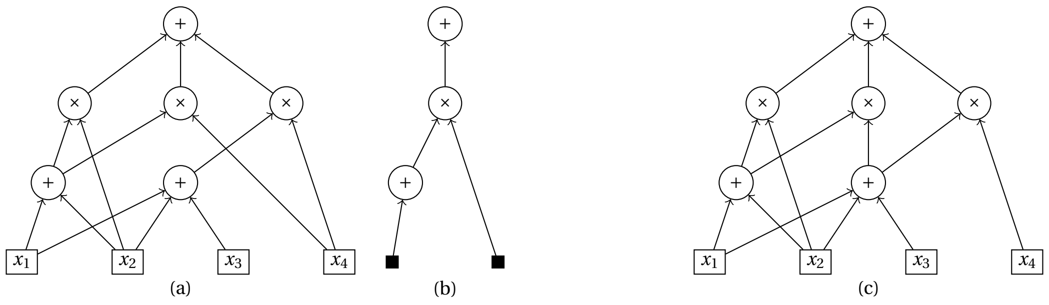


Figure 1: Examples of circuits with restricted parse trees

1.2 Polynomial identity testing

A *Polynomial Identity Test (PIT)* is an algorithm that, given a circuit as input, checks if the circuit is computing the zero polynomial or not. The standard Ore-DeMillo-Lipton-Schwartz-Zippel lemma [Ore22, DL78, Sch80, Zip79] provides a simple randomized algorithm but the goal is to construct an efficient deterministic PIT. A stronger test is what is called a *black-box PIT* where we are only provided evaluation access to the circuit. Hence, a black-box PIT is essentially equivalent to constructing a *hitting set* i.e., a set of points (or matrices, in the case of non-commutative polynomials) $\mathcal{H}$ such that every non-zero polynomial from the class of interest is guaranteed to evaluate to a nonzero value on some element $\mathbf{a} \in \mathcal{H}$. PITs that use the structure of the circuit are called *white-box* PITs.

The task of constructing efficient PITs is intimately connected to the task of proving lower bounds [HS80, KI04, Agr05]. Once we have a lower bound for a class $\mathcal{C}$, it is natural to ask if we can also construct efficient PITs for that class. Raz and Shpilka [RS05] gave the first deterministic polynomial time white-box PIT for the class of non-commutative ABPs. Forbes and Shpilka [FS13] gave a quasipolynomial ($n^{O(\log n)}$) size hitting set for non-commutative ABPs. This was achieved by studying a natural commutative analogue of non-commutative ABPs, and this was the class of *Read-Once Oblivious Algebraic Branching Programs (ROABPs)* where the variables are read in a "known order".

[1] the formula is levelled, and all nodes at a level have the same fan-in

The class of ROABPs is interesting in its own right owing to the connection with the "RL vs L" question. In fact, much of the hitting set constructions for ROABPs has been inspired by Nisan's [Nis92] pseudorandom generator for RL (which has seed length $O(\log^2 n)$). As mentioned earlier, Forbes and Shpilka gave a hitting set of size $n^{O(\log n)}$ for polynomial sized ROABPs when the order in which variables are read was known. Agrawal, Gurjar, Korwar and Saxena [AGKS15] presented a different hitting set for the class of commutative ROABPs that did not need the knowledge of the order in which the variables were read. Subsequently, Gurjar, Korwar, Saxena and Thierauf [GKST17] studied polynomials that can be computed as a sum of constantly many ROABPs (of possibly different orders) and presented a polynomial time white-box PIT, and also a quasipolynomial time black-box PIT for this class.

Lagarde, Malod and Perifel [LMP19], besides presenting lower bounds for non-commutative UPT circuits, also gave a polynomial time white-box PIT for this class. This was extended by Lagarde, Limaye and Srinivasan [LLS19] to a white-box algorithm for non-commutative circuits with constantly many parse tree shapes (analogous to the result of [GKST17]). The question of constructing black-box PITs was left open by them, and we answer this in our paper.

1.3 Our results

Polynomial Identity Testing

Our main results are hitting sets for the class of polynomials computed by UPT circuits and related classes.

Theorem 1.1 (Hitting sets for UPT circuits)**.** *There is an explicit hitting set $\mathcal{H}_{d,n,s}$ of at most $(snd)^{O(\log d)}$ size for the class of degree d n-variate homogeneous non-commutative polynomials in $\mathbb{F}\langle x_1,\ldots,x_n\rangle$ that are computed by UPT circuits of size at most s.*

This result builds on the technique of *basis isolating weight assignments* introduced by [AGKS15] for constructing hitting sets for ROABPs. Furthermore, we can also extend the hitting set to the class of non-commutative circuits that have *few shapes* (analogous to [GKST17]'s hitting set for sum of few ROABPs).

Theorem 1.2 (Hitting sets for circuits with few parse tree shapes)**.** *There is an explicit hitting set $\mathcal{H}_{d,n,s,k}$ of size at most $(s^{2^k} nd)^{O(\log d)}$ for the class of n-variate degree d homogeneous non-commutative polynomials in $\mathbb{F}\langle x_1,\ldots,x_n\rangle$ that are computed by non-commutative circuits of size at most s consisting of parse trees of at most k shapes.*

Both the above theorems are fully black-box in the sense that it is not required to know the underlying shape(s). For the case of non-commutative ABPs (and more generally, ROABPs in a known order), Gurjar, Korwar and Saxena [GKS17] presented a more efficient hitting set when the width of the ABP is small. For UPT circuits, there is a natural notion of *preimage-width* of a UPT circuit (formally defined in Definition 2.3) that corresponds to the notion of width of an ABP. We show an analogue of the hitting set of Gurjar, Korwar and Saxena for the class of UPT circuits of small *preimage-width* if the underlying shape of the parse trees is known.

Theorem 1.3 (Hitting sets for known-shape low-width UPT circuits)**.** *Let $\mathscr{C}_{n,d,T,w}$ be the class of n-variate degree d non-commutative polynomials that are computable by UPT circuits of preimage-width at most w and underlying parse-tree shape as T. Over any field of zero or large characteristic, there is an explicit hitting set $\mathscr{H}_{n,d,T,w}$ of size $w^{O(\log d)}\operatorname{poly}(nd)$ for $\mathscr{C}_{n,d,T,w}$.*

These hitting sets also translate to the natural commutative analogues of *UPT set-multilinear circuits* etc. (formally defined in Definition 5.1).

Structural results

If f is a non-commutative polynomial of degree d and if $\sigma \in S_d$ is a permutation on d letters, we define the *shuffling* of f by σ (denoted by $\Delta_\sigma(f)$) as the natural operation of permuting each *word* of f according to σ.

The three PIT statements stated above begin with the following depth reduction statement about UPT circuits.

Theorem 1.4 (Depth reduction for UPT circuits)**.** *Let f be an n-variate degree d polynomial that is computable by a UPT circuit of preimage-width w. Then, there is some $\sigma \in S_d$ such that $\Delta_\sigma(f)$ can be computed by a UPT circuit of $O(\log d)$ depth and preimage-width at most $O(w^2)$.*

The above theorem implies that $\Delta_\sigma(f)$ is computable by an ABP of quasipolynomial size. We also show that this blow-up of quasipolynomial size is tight.

Theorem 1.5 (Separating UPT circuits and ABPs, under shuffling)**.** *There is an explicit n-variate degree d non-commutative polynomial f that is computable by UPT circuits of preimage-width $w = \operatorname{poly}(n,d)$ such that for every $\sigma \in S_d$, the polynomial $\Delta_\sigma(f)$ requires non-commutative ABPs of size $(nd)^{\Omega(\log nd)}$ to compute it.*

We also extend the lower bound of [LMP19] to give a polynomial computed by a *skew circuit* that requires exponential sized UPT circuits under any shuffling. Details are in Section 8.

1.4 Proof ideas

As mentioned, the starting point of all these results is the depth reduction. From a result of Nisan [Nis91], the palindrome polynomial Pal_d is known to require ABPs of size $2^{\Omega(d)}$ even though it can be computed by a polynomial sized UPT circuit. Therefore, Pal_d cannot be computed by a circuit of depth $o(d/\log d)$. The key insight here is that even though Pal_d cannot be computed by small depth non-commutative circuits, a shuffling of the palindrome is

$$\sum_{w_1,\ldots,w_d \in [n]} x_{w_1} x_{w_1} x_{w_2} x_{w_2} \cdots x_{w_d} x_{w_d} = \prod_{i=1}^{d} \left(x_1 x_1 + \cdots + x_n x_n\right),$$

which is of course computable by an $O(\log d)$ depth UPT formula even. Hence we attempt to reduce the depth under a suitable shuffling.

In order to establish the depth reduction (Theorem 1.4) we follow the strategy of Valiant, Skyum, Berkowitz and Rackoff [VSBR83] and Allender, Jiao, Mahajan and Vinay [AJMV98] but make use of the UPT structure (work with different *frontier nodes* and *gate quotients*) based on the underlying shape of the parse trees. It was pointed out to us that the key ideas in our proof of depth reduction were used by Arvind and Raja ([AR16]) for a commutative analogue of UPT circuits.

This depth reduction immediately yields that there is a quasipolynomial sized ABP computing a shuffling of f. We show that this blow-up is tight (Theorem 1.5) by essentially following the proof of Hrubeš and Yehudayoff [HY16] to separate monotone ABPs and monotone circuits in the commutative world.

In order to obtain hitting sets for UPT circuits, one could potentially just use the fact that there is a quasipolynomial sized ABP computing a shuffling of f and just use the known hitting sets for non-commutative ABPs [FS13] to obtain a hitting set of $\mathrm{poly}(ndw)^{O(\log^2 d)}$. However, we directly work with the UPT circuit and lift the technique of *basis isolating weight assignments* of Agrawal, Gurjar, Korwar and Saxena [AGKS15] to this more general setting to obtain Theorem 1.1. Theorem 1.3 is a straightforward generalization of the ideas of Gurjar, Korwar and Saxena [GKS17] once we observe that the depth reduction keeps the preimage-width small.

Theorem 1.2 essentially follows the same ideas of Gurjar, Korwar, Saxena and Thierauf [GKST17]. The techniques of [GKST17] are general enough that once a circuit class has a *characterizing set of dependencies* and a *basis isolating weight assignment*, there is a natural method to lift the techniques to work with the sum of few elements from this class. [GKST17] use this for ROABPs and we use this for UPT circuits.

To summarize, once we obtain the depth reduction, much of the results in this paper is a careful translation of prior work of [HY16], [AGKS15], [GKST17], [GKS17] to the setting of UPT (or FewPT) circuits. Consequently, this also generalizes the hitting sets of [AGKS15, GKST17, GKS17] from ROABPs to *UPT (or FewPT) set-multilinear* circuits. Such a generalization was unknown prior to this work.

2 Preliminaries

2.1 Notation

- We use $\mathbb{F}\langle x_1,\ldots,x_n\rangle$ to refer to the ring of polynomials in non-commuting variables $\{x_1,\ldots,x_n\}$. For a parameter d, we use $\mathbb{F}\langle x_1,\ldots,x_n\rangle_{\deg=d}$ to refer to the set of polynomials in $\mathbb{F}\langle x_1,\ldots,x_n\rangle$ that are homogeneous and of degree d. Similarly, $\mathbb{F}\langle x_1,\ldots,x_n\rangle_{\deg\leq d}$ refers to the set of polynomials of degree at most d.
- We use boldface letters $\mathbf{x}$ and $\mathbf{y}$ to denote sets of variables (the number of variables would be clear from context). We shall also use $[d]$ to refer to the set $\{1,2,\ldots,d\}$.
- The paper would sometime shift between the commutative and the non-commutative domains. We use $\mathbf{x}$ whenever we are talking about non-commutative variables, and $\mathbf{y}$, $\mathbf{z}$ for variables in the

commutative domain.

2.2 Basic definitions

UPT and FewPT circuits

Definition 2.1 (Parse trees)**.** *A parse tree T of a circuit C is a tree obtained as follows:*

- *the root of C is the root of T,*
- *if $v \in T$ is a $\times$ gate, then all the children in C are the children of v in T in the same order,*
- *if $v \in T$ is a $+$ gate, then exactly one child of v in C is a child of v in T.*

The value of the parse tree T, denoted by $[T]$, is just the product of the leaf labels in T. $\Diamond$

Intuitively, a parse tree is a *certificate* that a monomial was produced in the computation of C (though it could potentially be canceled by other parse trees computing the same monomial). Therefore, if f is the polynomial computed by C, then

$$f = \sum_{T \text{ is a parse tree}} [T].$$

Definition 2.2. *(UPT and FewPT circuits) A circuit C computing a homogeneous polynomial is said to be a* Unique Parse Tree (UPT) circuit *if all parse trees of C have the same shape (that is, they are identical except perhaps for the gate names).*

A circuit C that computes a homogeneous polynomial is said to be a FewPT(k) circuit if the parse trees of C have at most k distinct shapes. $\Diamond$

Definition 2.3 (Preimage-width)**.** *Suppose C is a UPT circuit and say T is the shape of the underlying parse trees. For a node $\tau \in T$ and a gate $g \in C$, we shall say that g is a* preimage *of τ, denoted by $g \sim \tau$, if and only if there is some parse tree T' of C where the gate g appears in position τ.*

The preimage-width *of a UPT circuit C is the largest size of preimages of any node $\tau \in T$. That is,*

$$\text{preimage-width}(C) = \max_{\tau \in T} \left|\{g \in C : g \sim \tau\}\right|. \qquad \Diamond$$

It is clear that if C is a UPT circuit of preimage-width w computing a homogeneous degree d polynomial, then the size of C is at most dw. The preimage-width of a UPT circuit is a more useful measure to study than the size of the circuit. A simple concrete example of this is that the standard conversion of homogeneous ABPs to homogeneous circuits in fact yields UPT circuits. Furthermore, the width of the ABP is directly related to the preimage-width of the resulting UPT circuit.

Observation 2.4. *If f is computable by a width w homogeneous algebraic branching program, then f can be equivalently computed by UPT circuits of preimage-width w^2.* $\square$

Infixions

Definition 2.5 (Infixion)**.** *For any $d_1, d_2 \geq 0$ and p satisfying $0 \leq p \leq d_2$, define $\times_p$ as the unique bilinear map $\times_p : \mathbb{F}\langle \mathbf{x} \rangle_{\deg = d_1} \times \mathbb{F}\langle \mathbf{x} \rangle_{\deg = d_2} \to \mathbb{F}\langle \mathbf{x} \rangle_{\deg = d_1 + d_2}$ that satisfies*

$$x_{w_1} \cdots x_{w_{d_1}} \times_p x_{v_1} \cdots x_{v_{d_2}} = x_{v_1} \cdots x_{v_p} x_{w_1} \cdots x_{w_{d_1}} x_{v_{p+1}} \cdots x_{v_{d_2}}.$$

*We shall refer to this operation as a p-*infixion [2]. ◊

For instance, the usual multiplication (or concatenation) operation is just $\times_0$.

Shuffling of a polynomial

Definition 2.6 (Shuffling of a non-commutative polynomial)**.** *Let $P_d(x_1, \ldots, x_n) \in \mathbb{F}\langle x_1, \ldots, x_n \rangle_{\deg = d}$ be a homogeneous degree d non-commutative polynomial. Given any permutation $\sigma \in S_d$ over d-letters, we can define the* shuffling of P_d via σ *as the unique linear map $\Delta_\sigma : \mathbb{F}\langle x_1, \ldots, x_n \rangle_{\deg = d} \to \mathbb{F}\langle x_1, \ldots, x_n \rangle_{\deg = d}$ that is obtained by linearly extending*

$$\Delta_\sigma(x_{w_1} \cdots x_{w_d}) = x_{w_{\sigma(1)}} \cdots x_{w_{\sigma(d)}}.$$ ◊

2.3 Basic lemmas

Canonical UPT circuits, and types of gates

We shall say that a UPT circuit C with underlying parse tree shape T is *canonical* if for every gate $g \in C$ there is some node $\tau \in T$ such that every parse tree of C involving g has g only in position τ. In other words, every gate of the circuit has a unique type associated with it.

Lemma 2.7 ([LMP19])**.** *Suppose if $f \in \mathbb{F}\langle x_1, \ldots, x_n \rangle$ is a homogeneous, degree d, non-commutative polynomial computed by a non-commutative UPT circuit of preimage-width w. Then, f can be equivalently computed by a canonical UPT circuit of preimage-width w as well.*

For a canonical UPT circuit where the parse trees have shape T, we shall say that g has type τ if $\tau \in T$ is the unique node in T such that $g \sim \tau$.

Fix a $\tau \in T$ and let i be the number of leaves of the subtree rooted at τ, and let p be the number of leaves to the left of τ in the inorder traversal of T. We shall then say that τ (or a gate $g \in C$ of type τ) has *position-type* (i, p). The following lemma allows us to write the polynomial computed by the circuit as a small sum of $\times_p$-products.

Lemma 2.8 ([LMP19])**.** *Let f be a polynomial computed by a canonical UPT circuit C of preimage-width w and say T is the shape of the underlying parse trees. If $\tau \in T$ with position-type (i, p), then we can write*

[2]An infix is defined as "a formative element inserted in a word". E.g. The plural of 'spoonful' is 'spoonsful', obtained by adding the infix 's'.

f as

$$f(\mathbf{x}) = \sum_{r=1}^{w} g_r(\mathbf{x}) \times_p h_r(\mathbf{x}),$$

where $\deg g_r = i$ *and* $\deg h_r = \deg(f) - i$ *for all* $r = 1, \ldots, w$*.*

3 Depth reduction for UPT circuits

This section shall address Theorem 1.4, which we recall below.

Theorem 1.4 (Depth reduction for UPT circuits)**.** *Let f be an n-variate degree d polynomial that is computable by a UPT circuit of preimage-width w. Then, there is some $\sigma \in S_d$ such that $\Delta_\sigma(f)$ can be computed by a UPT circuit of $O(\log d)$ depth and preimage-width at most $O(w^2)$.*

It was pointed out to us that a very similar depth reduction was also proved by Arvind and Raja [AR16]. They showed that a commutative UPT set-multilinear circuit can be depth-reduced to a corresponding quasi-polynomial sized $O(\log d)$ depth UPT set-multilinear formula via Hyafil's [Hya79] depth reduction. Using techniques similar to [VSBR83], one can obtain a polynomial sized circuit of depth $O(\log d)$ while maintaining the UPT property. Though this can be inferred from the results in [AR16], we state and prove it in the form needed for the non-commutative setting.

3.1 UPT infixion-circuits

To prove the depth reduction, we will move to an intermediate model of *UPT infixion-circuits.*

Definition 3.1 (UPT infixion-circuits)**.** *The class of* UPT infixion-circuits *is a generalization of homogeneous non-commutative circuits in that the internal gates are $+$ gates and $\times_p$ gates instead of the usual $+$ and $\times$ gates. We shall also say that the circuit is* semi-unbounded *if all $\times_p$ gates have fan-in bounded by* 2 *(with no restriction on $+$ gates).*

A parse tree *for an infixion-circuit is similar to parse trees in a general non-commutative circuit but the internal nodes of the parse tree are labelled by $+$ and $\times_p$ (with the p specified at each gate).*

We shall say that an infixion-circuit C is UPT if every parse tree is of the same shape. That is, two parse trees in C can differ only in the gate names. ◊

To prove Theorem 1.4, we shall first depth reduce the circuit to obtain an infixion-circuit computing f of $O(\log d)$ depth. Then, we will convert that to a UPT circuit that computes a shuffling of f.

Lemma 3.2 (Depth reducing to infixion-circuits)**.** *Let $f \in \mathbb{F}\langle \mathbf{x} \rangle$ be a homogeneous degree d polynomial that is computable by a UPT circuit of preimage-width s. Then, f can be equivalently be computed by a semi-unbounded UPT infixion-circuit of preimage-width $O(s^2)$ and depth $O(\log d)$.*

Proof. Let C be the UPT circuit computing $f(x_1,\dots,x_n)$ and say T is the shape of the parse trees of C. For any node $\tau \in T$, let $\mathscr{F}_\tau$ be the set of all gates in C whose position in T is τ. For two gates $u, v \in C$, we shall say that $u \succeq v$ if the place of u in T is an ancestor of the place of v in T. We shall abuse notation and use $u \succeq \tau$ to mean that u's position in T is an ancestor of $\tau \in T$. For a gate $u \in C$, let $[u]$ refer to the polynomial computed at that gate. Similar to [VSBR83, AJMV98], we define inductively the following notion of a *gate quotient* for any pair of gates $u, v \in C$:

$$[u:v] = \begin{cases} 0 & \text{if } u \not\succeq v, \\ 1 & \text{if } u = v, \\ [u_1:v] + [u_2:v] & \text{if } u = u_1 + u_2, \\ [u_1:v]\cdot[u_2] & \text{if } u = u_1 \times u_2 \text{ and } u_1 \succeq v, \\ [u_1]\cdot[u_2:v] & \text{if } u = u_1 \times u_2 \text{ and } u_2 \succeq v. \end{cases}$$

Claim 3.3. *For any $u \in C$, if $\tau \in T$ such that $u \succeq \tau$, then*

$$[u] = \sum_{\substack{w \in C \\ w \sim \tau}} [w] \times_p [u:w] \tag{3.4}$$

for a suitable p depending just on τ and the type of u. Furthermore, suppose $u, v \in C$ with v being a multiplication gate and if $\tau \in T$ such that $u \succeq \tau \succeq v$ then

$$[u:v] = \sum_{\substack{w \in C \\ w \sim \tau}} [w:v] \times_p [u:w]. \tag{3.5}$$

for a suitable p depending just on τ and the type of u and v.

We'll defer this proof to later and first finish the proof of Lemma 3.2. With (3.4) and (3.5), we can construct the infixion-circuit C' for f just as in [VSBR83, AJMV98]. The circuit C' would have gates computing each $[u]$ and $[u:v]$ for nodes $u, v \in C$ with $u \succeq v$ and v being a multiplication gate. The wirings in C' is built by appropriate applications of (3.4) and (3.5).

Let $u \in C$ and say $\deg[u] = d_u$. The plan would be to set up the computation in C' so that using an $O(1)$ depth computation, we can compute $[u]$ using gates whose degrees are a constant factor smaller than d_u. Consider any parse tree rooted at u, and starting from u follow the *higher degree* child. Let τ be the last point on the path with degree $\geq d_u/2$ (degree of its children will be $< d_u/2$). Applying (3.4),

$$\begin{aligned} [u] &= \sum_{w \sim \tau} [w] \times_p [u:w] \\ &= \sum_{w \sim \tau} ([w_1] \times [w_2]) \times_p [u:w] && \text{where } w = w_1 \times w_2. \end{aligned}$$

Now observe that each of the terms on the RHS, $[u:w], [w_1], [w_2]$ have degree at most $d_u/2$, as we wanted. Furthermore, all coordinates of the tuple $([u:w],[w_1],[w_2])$ are all of the same *type* as we run over all $w \sim \tau$.

We now need to show how to compute $[u:v]$ for a pair $u \succ v$. Say $\deg[u] = d_u$ and $\deg[v] = d_v$. For this, start with some parse tree rooted at u and walk down the path leading to the place of v, and let τ be the last point on this path such that $\deg\tau \geq \frac{d_u+d_v}{2}$. Using (3.5),

$$\begin{aligned}[u:v] &= \sum_{w\sim\tau} [w:v] \times_p [u:w]\\ &= \sum_{w\sim\tau} ([w_1]\times[w_2:v]) \times_p [u:w]\end{aligned}$$

where $w = w_1 \times w_2$ and $w_2 \succeq v$ (the other possibility is identical). By the choice of τ, we have $\deg[u:w], \deg[w_2:v] \leq \frac{d_u-d_v}{2}$. However, the best bound we can give on $\deg[w_1]$ is $d_u - d_v$. Nevertheless, we can apply (3.4) again on $[w_1]$ by finding a suitable $\tau' \prec w_1$ satisfying $\deg\tau' \geq \frac{\deg w_1}{2}$ and write

$$\begin{aligned}[u:v] &= \sum_{w\sim\tau} ([w_1]\times[w_2:v]) \times_p [u:w]\\ &= \sum_{w\sim\tau}\left(\left(\sum_{w'\sim\tau'}[w'] \times_{p'} [w_1:w']\right)\times[w_2:v]\right)\times_p[u:w]\\ &= \sum_{w\sim\tau}\sum_{w'\sim\tau'}\Big(\big(([w_1']\times[w_2'])\times_{p'}[w_1:w']\big)\times[w_2:v]\Big)\times_p[u:w]\end{aligned}$$

By the choice of τ and τ', each of the *factors* on the RHS have degree at most $\frac{(d_u-d_v)}{2}$ as we wanted. Furthermore, once again, all of the summands consists of similarly typed factors.

This naturally yields an infixion-circuit computing f of depth $O(\log d)$ and size $\mathrm{poly}(s)$. Since all summands consist of similarly typed factors, it follows that the circuit is UPT as well. □

Proof of Claim 3.3. The proof is by induction. As a base case, suppose $u \sim \tau$. Then, $[u]$ is just the sum of the values of parse trees. Some of the parse trees use u. Of all nodes $w \in C$ such that $w \sim \tau$, only $[u:u] = 1$ and every other $[u:w] = 0$. Therefore, clearly $[u] = \sum_{w\sim\tau}[w]\cdot[u:w]$.

Now suppose $u \succ \tau$ and say we already know that $[u'] = \sum_{w\sim\tau}[w]\times_p[u':w]$ for every $u \succ u' \succeq \tau$. If $u = u_1 + u_2$, then

$$\begin{aligned}[u] &= [u_1]+[u_2]\\ &= \left(\sum_{w\sim\tau}[w]\times_p[u_1:w]\right)+\left(\sum_{w\sim\tau}[w]\times_p[u_2:w]\right)\\ &= \sum_{w\sim\tau}[w]\times_p([u_1:w]+[u_2:w])\\ &= \sum_{w\sim\tau}[w]\times_p[u:w].\end{aligned}$$

Similarly, suppose $[u] = [u_1] \times [u_2]$. We have two cases depending on whether $u_1 \succeq \tau$ or $u_2 \succeq \tau$.

If $u_1 \succeq \tau$, then

$$\begin{aligned}
[u] &= [u_1] \times [u_2] \\
&= \left(\sum_{w \sim \tau} [w] \times_p [u_1 : w] \right) \times [u_2] \\
&= \sum_{w \sim \tau} [w] \times_p ([u_1 : w] \times [u_2]) \\
&= \sum_{w \sim \tau} [w] \times_p [u : w].
\end{aligned}$$

If $u_2 \succeq \tau$, then

$$\begin{aligned}
[u] &= [u_1] \times [u_2] \\
&= [u_1] \times \left(\sum_{w \sim \tau} [w] \times_p [u_2 : w] \right) \\
&= \sum_{w \sim \tau} [w] \times_{p + \deg u_1} ([u_1] \times [u_2 : w]) \\
&= \sum_{w \sim \tau} [w] \times_{p + d_1} [u : w].
\end{aligned}$$

Essentially the same proof works for (3.5) as well. □

Lemma 3.6 (Infixion-circuits to circuits for a shuffling)**.** *Let $f \in \mathbb{F}\langle \mathbf{x} \rangle$ be a homogeneous degree d polynomial that is computable by a UPT infixion-circuit C' of size s. Consider the circuit C'' obtained by replacing all $\times_p$ (infixion) gates in C' by $\times$ gates. Then, C'' computes $\Delta_\sigma(f)$ for some $\sigma \in S_d$.*

Proof. We shall prove this by induction. We need a slightly stronger inductive hypothesis which is that the choice of permutation σ depends only on the shape of the parse trees in C'.

Say u is the root of C'. Suppose u is a $+$ gate and say $u = u_1 + u_2 + \cdots + u_r$. If $u' = u'_1 + \cdots + u'_r$ is the resulting computation in C'' then by the inductive hypothesis, we know that there is a $\sigma \in S_d$ such that $[u'_i] = \Delta_\sigma([u_i])$. Therefore,

$$[u'] = \sum_{i=1}^{r} \Delta_\sigma([u_i]) = \Delta_\sigma([u]).$$

Suppose $u = u_1 \times_p u_2$ with $\deg[u_1] = d_1$ and $\deg[u_2] = d_2$. Say $u_1 = \sum_{\alpha \in [n]^{d_1}} a_\alpha x_\alpha$ and $\sum_{\beta \in [n]^{d_2}} b_\beta x_\beta$. Then, $[u] = \sum_{\alpha,\beta} a_\alpha b_\beta \cdot x_\alpha \times_p x_\beta$. If u', u'_1 and u'_2 is the resulting computation in C'', then

$$\begin{aligned}
[u'] &= [u'_1] \times [u'_2] \\
&= \Delta_{\sigma_1}([u_1]) \times \Delta_{\sigma_2}([u_2]) && \text{for some } \sigma_1 \in S_{d_1}, \sigma_2 \in S_{d_2}, \\
&= \sum_{\alpha,\beta} a_\alpha b_\beta \cdot (\Delta_{\sigma_1}(x_\alpha) \times \Delta_{\sigma_2}(x_\beta)) \\
&= \sum_{\alpha,\beta} a_\alpha b_\beta \cdot \Delta_\sigma(x_\alpha \times_p x_\beta) && \text{for some } \sigma \in S_d, \\
&= \Delta_\sigma([u])
\end{aligned}$$
□

Together, Lemma 3.2 and Lemma 3.6 yield Theorem 1.4. □(Theorem 1.4)

The following corollary is immediate from the fact that any circuit of depth D and size s can be computed by a formula of size $s^{O(d)}$ and hence an ABP of size $s^{O(d)}$.

Corollary 3.7. *If $f \in \mathbb{F}\langle \mathbf{x} \rangle$ is a homogeneous degree d polynomial that is computable by a UPT circuit of size s, then there is some $\sigma \in S_d$ such that $\Delta_\sigma(f)$ is computable by a non-commutative algebraic branching program of size $s^{O(\log d)}$.*

Furthermore, the shuffling σ that permits this can also be efficiently computed given the underlying shape for the circuit computing f.

3.2 UPT circuits of constant width

For a UPT circuit C, we shall say that its *width* is w if for every node τ in the shape T, there are at most w gates of C that have type τ. The following observation is evident from the proof of the above depth reduction.

Observation 3.8. *If C is a UPT circuit of width w, then the depth reduced circuit C' as obtained in Theorem 1.4 has width $O(w^2)$.*

This observation would allow us to yield a more efficient hitting set for the class of *small width known shape* UPT circuits. Details are present in Section 9.2.

4 Separating ROABPs and UPT circuits

Theorem 1.5 (Separating UPT circuits and ABPs, under shuffling)**.** *There is an explicit n-variate degree d non-commutative polynomial f that is computable by UPT circuits of preimage-width $w = \mathrm{poly}(n, d)$ such that for every $\sigma \in S_d$, the polynomial $\Delta_\sigma(f)$ requires non-commutative ABPs of size $(nd)^{\Omega(\log nd)}$ to compute it.*

The polynomial and the proof technique described here were introduced by Hrubeš and Yehudayoff [HY16] to separate monotone circuits and monotone ABPs in the commutative regime. The polynomial described here is a non-commutative analogue of the polynomial used by [HY16]. Much of the proof is also the argument of [HY16] tailored to the non-commutative setting.

4.1 The polynomial

Let T_d denote the complete binary tree of depth d (with 2^d leaves) and let $D = 2^{d+1}-1$ refer to the number of nodes in T_d. We shall say that a colouring $\gamma : T_d \to \mathbb{Z}_m$ is *legal* if for every node $u \in T$, if v and w are the children of u then $\gamma(u) = \gamma(v) + \gamma(w) \bmod m$.

Let $v_1, \ldots, v_D$ be the vertices of T_d listed in an *in-order* manner (left-subtree listed inductively, then the root, and then the right-subtree listed inductively). We now define the non-commutative polynomial $P_d(x_1, \ldots, x_m) \in \mathbb{F}\langle x_1, \ldots, x_m \rangle$ of degree $D = 2^{d+1} - 1$ as

$$P_d(x_1, \ldots, x_m) = \sum_{\substack{\gamma \in [m]^D \\ \gamma \text{ is legal}}} x_{\gamma(v_1)} x_{\gamma(v_2)} \cdots x_{\gamma(v_D)}. \tag{4.1}$$

Lemma 4.2 (Upper bound)**.** *For every $m, d > 0$, the polynomial $P_d(y_1, \ldots, y_m)$ can be computed by a non-commutative UPT circuit of size $O(m^2 d)$ and preimage-width $O(m^2)$.*

(Refer to Section 7 for a proof).

Theorem 4.3 (Lower bound)**.** *For every permutation $\sigma \in S_D$, any non-commutative ABP computing the polynomial $\Delta_\sigma(P_d)$ has width $m^{\Omega(d)}$.*

Hence for $d = \log m$, we have that $P_d(x_1, \ldots, x_m)$ is computable by a UPT circuit of size $O(m^2 \log m)$ but for every $\sigma \in S_D$ the above theorem tells us that $\Delta_\sigma(P_d)$ requires ABPs of width $m^{\Omega(\log m)}$ to compute it. The lower bound follows on exactly same lines as the [HY16]. A proof is present in Section 7.

5 Hitting sets for non-commutative models

Commutative brethren of non-commutative models

This reduction to an appropriate commutative case was used by Forbes and Shpilka [FS13] to reduce constructing hitting sets for non-commutative ABPs to hitting sets for commutative ROABPs (more precisely, to set-multilinear ABPs). They studied the image of the non-commutative polynomial under the map $\Psi : \mathbb{F}\langle \mathbf{x} \rangle_{\deg=d} \to \mathbb{F}[y_{1,1}, \ldots, y_{d,n}]$ which is the unique $\mathbb{F}$-linear map given by $\Psi : x_{w_1} \cdots x_{w_d} \mapsto y_{1,w_1} \cdots y_{d,w_d}$.

For the model of non-commutative UPT circuits, the appropriate commutative model is a restriction of set-multilinear circuits that we call UPT set-multilinear (UPT-SML) circuits.

Definition 5.1 (Set-multilinear circuits)**.** *Let $\mathbf{y} = \mathbf{y}_1 \sqcup \cdots \sqcup \mathbf{y}_d$ be a partition of the variables. A circuit C computing a polynomial $f \in \mathbb{F}[\mathbf{y}]$ is said to be a* set-multilinear circuit *with respect to the above partition if:*

- *each gate $g \in C$ is labelled by a subset $S_g \subseteq [d]$ and g computes a polynomial over variables $\bigcup_{i \in S_g} \mathbf{y}_i$ where every monomial of $[g]$ is divisible by exactly one variable in $\mathbf{y}_i$ for each $i \in S_g$,*
- *if g is a $+$ gate, then the subset that labels g also labels each of its children,*
- *if g is a $\times$ gate with g_1 and g_2 being its children, then the subsets S_{g_1} and S_{g_2} labelling g_1 and g_2 respectively is a partition of S_g. That is, $S_g = S_{g_1} \sqcup S_{g_2}$.* $\Diamond$

We shall say the circuit C is UPT set-multilinear *if every parse tree of C is of the same shape and identically labelled. That is, if g and g' are $\times$ gates labelled by a set $S \subseteq [d]$, and if $g = g_1 \times g_2$ with S_1 and S_2 labelling g_1 and g_2, then the children of g' are also labelled by S_1 and S_2 respectively.*

We shall say the set-multilinear circuit C is FewPT(k) set-multilinear *if the circuit consists of parse trees of at most k different shapes.*

A natural generalization that will be useful later is a *multi-output UPT set-multilinear* circuit, which is a UPT set-multilinear circuit that potentially has multiple output gates, which are all labelled with the same subset.

Forbes and Shpilka [FS13] showed that constructing hitting sets for these commutative models suffices for the corresponding non-commutative models by a simple reduction (details in Section 9.1). We shall therefore focus on these commutative models for the hitting set constructions. And since we have already seen that such circuits can be depth reduced[3] to $O(\log d)$ depth, it suffices to construct a hitting set for $O(\log d)$-depth UPT and FewPT set-multilinear circuits.

5.1 Preliminaries for PIT

Weight assignments and basis isolation

To construct hitting sets for ROABPs, Agrawal, Gurjar, Korwar and Saxena [AGKS15] defined the notion of *basis isolating weight assignments* for associated vector spaces of polynomials. The description presented here is an adaptation of the approach of [AGKS15] to set-multilinear circuits of small depth.

Definition 5.2 (Basis Isolating Weight Assignment (BIWA))**.** *A* weight assignment *is a function* $\mathrm{wt} : \mathbf{y} \to [M]^k$*, for some positive integer* M*, that can then be extended to all multilinear monomials over* $\mathbf{y}$ *via*

$$\mathrm{wt}\left(\prod_{i\in S} y_i\right) = \sum_{i\in S}^{n} \mathrm{wt}(y_i). \qquad \Diamond$$

Let V *be a vector space of polynomials in* $\mathbb{F}[\mathbf{y}]$*, which can also be thought of as a matrix with a generating set of polynomials listed out as rows (with each column being indexed by a monomial in* $\mathbf{y}$*).*

Such a weight assignment wt *is said to be a* basis isolating weight assignment *for* V *if there exists a basis of its column space, indexed by* $B \subseteq \mathrm{Mons}(\mathbf{y})$*, such that*

1. *if* $m_1, m_2 \in B$ *and* $m_1 \neq m_2$*, then* $\mathrm{wt}(m_1) \neq \mathrm{wt}(m_2)$*,*
2. *for every* $m \notin B$*,*

$$V_m \in \mathrm{span}\left\{V_{m'} \;:\; m' \in B\,,\, \mathrm{wt}(m') \prec \mathrm{wt}(m)\right\}$$

where by V_m *we mean the column of* V *indexed by the monomial* m *and* $\prec$ *is the lexicographic ordering on* $M^k \subset \mathbb{N}^k$*.*

Lemma 5.3 ([AGKS15])**.** *Let* V *be a vector space of polynomials in* $\mathbb{F}[\mathbf{y}]$ *and say* $f \in V$*. If* $\mathrm{wt} : \mathbf{y} \to [M]^k$ *is a BIWA for* V*, then if* $\mathbf{t} = \{t_1, \ldots, t_k\}$

$$f(y_1, \ldots, y_n) \neq 0 \Longleftrightarrow f(\mathbf{t}^{\mathrm{wt}(y_1)}, \cdots, \mathbf{t}^{\mathrm{wt}(y_n)}) \neq 0$$

(where $\mathbf{t}^{(\alpha_1,\ldots,\alpha_k)}$ *is short-hand for* $t_1^{\alpha_1} \cdots t_k^{\alpha_k}$ *).*

If $f \neq 0$ and $\deg(f) \leq d$, then $f(\mathbf{t}^{\mathrm{wt}(y_1)}, \ldots, \mathbf{t}^{\mathrm{wt}(y_n)})$ is a non-zero k-variate polynomial of degree at most dM. Hence, the polynomial identity lemma [Ore22, DL78, Zip79, Sch80] would present a $(dM+1)^k$ sized

[3] the shuffling just reorders the partition of the set-multilinear circuit

hitting set.

Definition 5.4 (Separating small sets of monomials)**.** *Let S be an arbitrary set of monomials over* $\mathbf{y}$*. We shall say that a weight assignment* $\mathrm{wt} : \mathbf{y} \to \mathbb{N}$ separates S *if for every distinct* $m, m' \in S$ *we have* $\mathrm{wt}(m) \neq \mathrm{wt}(m')$. $\lozenge$

Lemma 5.5 ([AB03])**.** *Let S be an arbitrary set of r multilinear monomials of degree at most d over variables* $\mathbf{y} = \{y_{ij} : i \in [d], j \in [n]\}$*. For a prime p, let $w_p : \mathbf{y} \to \mathbb{N}$ be a weight assignment given by*

$$w_p(y_{i,j}) = 2^{(i-1)n+(j-1)} \bmod p.$$

Then for all but at most $\binom{r}{2} \cdot n^2$ primes p, the weight assignment w_p separates S.

BIWAs for subspaces and products

Agrawal, Gurjar, Korwar and Saxena [AGKS15] constructed BIWAs for polynomials computed by ROABPs. The following two lemmas are slight abstractions of the key ideas in [AGKS15], so that they can also be applied in our setting. For the sake of completeness, the proofs are provided in Section 9.1.

Lemma 5.6 (BIWA for subspaces)**.** *Say V is a vector space of polynomials and suppose* wt *is a BIWA for V. Then, if V' is a subspace of V, then* wt *is a BIWA for V' as well.*

Lemma 5.7 (BIWA for variable disjoint products)**.** *Say $V_1 \subseteq \mathbb{F}[\mathbf{y}]$ and $V_2 \subseteq \mathbb{F}[\mathbf{z}]$ are two vector spaces of polynomials over disjoint sets of variables, and of dimension at most s. Suppose*

$$\mathrm{wt}_1 : \mathbf{y} \to \mathbb{N}^k$$
$$\mathrm{wt}_2 : \mathbf{z} \to \mathbb{N}^k$$

are BIWAs for V_1 and V_2 isolating bases B_1 and B_2 respectively. If $w : \mathbf{y} \cup \mathbf{z} \to \mathbb{N}$ is a weight assignment that separates $B_1 \cdot B_2 = \{m_1 m_2 : m_1 \in B_1, m_2 \in B_2\}$*. Then the weight assignment defined by*

$$\mathrm{wt} : \mathbf{y} \cup \mathbf{z} \to \mathbb{N}^{k+1}$$
$$\mathrm{wt} : y_i \mapsto (\mathrm{wt}_1(y_i), w(y_i)) \quad \text{for all } y_i \in \mathbf{y},$$
$$\mathrm{wt} : z_i \mapsto (\mathrm{wt}_2(z_i), w(z_i)) \quad \text{for all } z_i \in \mathbf{z},$$

is a BIWA for $V = V_1 \cdot V_2 = \mathrm{span}\{f \cdot g : f \in V_1, g \in V_2\}$.

5.2 Hitting sets for UPT set-multilinear circuits

Theorem 5.8 (Hitting sets for UPT set-multilinear circuits)**.** *Let $\mathscr{C}$ be the class of n-variate degree d set-multilinear polynomials (with respect to* $\mathbf{y} = \mathbf{y}_1 \sqcup \cdots \sqcup \mathbf{y}_d$*) that are computable by UPT set-multilinear* *circuits of preimage-width w and depth r. Then, for $M = \left(\binom{w}{2} n^2 d + 1\right)^2$, the set*

$$\mathscr{H} = \left\{ (b_{11}, \ldots, b_{dn}) \ : \ \mathbf{p} \in [M]^r \ , \ a_k \in A \ , \ b_{ij} = \prod_{k=1}^{r+1} a_k^{2^{(i-1)n+(j-1)} \bmod p_i} \right\}$$

is a hitting set for $\mathscr{C}$ of size $\operatorname{poly}(ndw)^r$.

The proof of this theorem is obtained by constructing what is called a *basis isolating weight assignment* for polynomials simultaneously computed by a multi-output UPT-SML circuit, heavily borrowing from the ideas in [AGKS15].

Proof. Suppose $f(\mathbf{y})$ is a polynomial that is computable by a UPT set-multilinear circuit C with respect to $\mathbf{y} = \mathbf{y}_1 \sqcup \cdots \sqcup \mathbf{y}_d$ and say C is of preimage-width size w and depth r.

Since C is a UPT set-multilinear circuit, let T be the shape of the parse tree. For each $\tau \in T$, we define the vector space

$$V_\tau = \operatorname{span}\left\{ [g] \ : \ g \in C \, , \ g \sim \tau \right\}.$$

The following claim relates the vector space corresponding to nodes in T to the vector spaces corresponding to the children.

Claim 5.9. *If $\tau \in T$ labels a $+$ gate and if τ' is the unique child of τ, then $V_\tau \subseteq V_{\tau'}$.*

If $\tau \in T$ labels a $\times$ gate and has children τ_1 and τ_2, then V_τ is a subspace of $V_{\tau_1} \cdot V_{\tau_2}$.

Proof. Suppose $\tau \in T$ labels a $+$ gate and say τ' is the unique child of τ in T. Pick an arbitrary $g \in C$ such that $g \sim \tau$. If $[g] = [g_1] + \cdots + [g_s]$, then each $g_i \sim \tau'$. Therefore, $[g_i] \in V_{\tau'}$ and $[g] = [g_1] + \cdots + [g_s]$ implies that $[g] \in V_{\tau'}$. Since the choice of g was an arbitrary gate of type τ, it follows that V_τ is a subspace of $V_{\tau'}$.

Say τ labels a $\times$ gate, and say τ_1 and τ_2 are the children of τ. Pick an arbitrary gate $g \in C$ with $g \sim \tau$. If $[g] = [g_1] \times [g_2]$ then $g_1 \sim \tau_1$ and $g_2 \sim \tau_2$. But that implies that $[g_1] \in V_{\tau_1}$ and $[g_2] \in V_{\tau_2}$ and therefore $[g] \in V_{\tau_1} \cdot V_{\tau_2}$. Once again, since the choice of g was arbitrary, we get V_τ is a subspace of $V_{\tau_1} \cdot V_{\tau_2}$. □ (Claim 5.9)

Define the *multiplication height* of any gate g, denoted by $\left|g\right|_\times$, as the largest number of $\times$ gates encountered on a path from g to a leaf. Starting with the leaves, we shall build towards a BIWA for V_{root}, which by Lemma 5.3 also yields a hitting set.

Let P be the set of the first $(dn^2\binom{w}{2} + 1)$ primes. For each $0 \le k \le r$ and $\mathbf{p} = (p_1, \ldots, p_k) \in P^k$, define the function

$$\Omega_{\mathbf{p}}^{(k)} : \mathbf{y} \to \mathbb{N}^{k+1}$$
$$\Omega_{\mathbf{p}}^{(k)} : y_{ij} \mapsto (j, 2^{(i-1)n+(j-1)} \bmod p_1, \ldots, 2^{(i-1)n+(j-1)} \bmod p_k).$$

The plan is to use $\Omega_{\mathbf{p}}^{(k)}$ to build BIWAs for each V_τ. For a $\tau \in T$ with $|\tau|_\times = k$, let $S_\tau \subseteq [d]$ be the subset of indices labelling τ. Define $\mathrm{wt}_{\mathbf{p}}^{(\tau)}$ to be the restriction of $\Omega_{\mathbf{p}}^{(k)}$ to $\cup_{i \in S_\tau} \mathbf{y}_i$:

$$\begin{aligned}\mathrm{wt}_{\mathbf{p}}^{(\tau)} &: \bigcup_{i \in S_\tau} \mathbf{y}_i \to \mathbb{N}^{k+1}\\ \mathrm{wt}_{\mathbf{p}}^{(\tau)}(y_{ij}) &= \Omega_{\mathbf{p}}^{(k)}(y_{ij}).\end{aligned}$$

We shall prove, by induction, that for each $0 \le k \le r$ there is a $\mathbf{p} \in P^k$ such that for every $\tau \in T$ with $|\tau|_\times \le k$, the weight assignment $\mathrm{wt}_{\mathbf{p}}^{(k)}$ is a BIWA for V_τ.

If τ was a leaf of T, then any such node just computes a variable. Clearly, $\mathrm{wt}_{\mathbf{p}}^{(\tau)} : (y_{ij}) \mapsto j$ is a BIWA as it gives distinct weights to all variables of a partition. Hence, $\mathrm{wt}_{\mathbf{p}}^{(\tau)}$ is a BIWA for all V_τ whenever τ is a leaf.

If τ is not a leaf but $|\tau|_\times = 0$, then neither τ nor its descendants are $\times$ gates. Hence, the subtree at τ has a unique leaf ℓ and all the nodes along this path are $+$ gates. By Claim 5.9, V_τ is a subspace of V_ℓ and hence, by Lemma 5.6, $\mathrm{wt}_{\mathbf{p}}^{(\tau)} = \mathrm{wt}_{\mathbf{p}}^{(\ell)}$ is a BIWA for V_τ. That finishes the base case of $k = 0$.

Suppose we have proved the claim up to $k-1$. Let T_k be the set of all nodes of multiplication height at most k that are $\times$ gates. By the inductive hypothesis, there exists $\mathbf{p} \in P^{k-1}$ such that $\mathrm{wt}_{\mathbf{p}}^{(\tau')}$ is BIWA for all $V_{\tau'}$ with $\left|\tau'\right|_\times < k$. Fix such a $\mathbf{p}$. For each $\tau \in T_k$, its children τ_1, τ_2 must have multiplication height at most $k-1$. Since C is set-multilinear, the subset of indices that label τ_1 and τ_2 must be disjoint. Say S_1 and S_2 are the subsets of indices labelling τ_1 and τ_2 respectively.

Hence, by Claim 5.9, V_τ is a subspace of $V_{\tau_1} \cdot V_{\tau_2}$. By our inductive hypothesis, we know that $\mathrm{wt}_{\mathbf{p}}^{(\tau_1)}$ and $\mathrm{wt}_{\mathbf{p}}^{(\tau_2)}$ are BIWAs for V_{τ_1} and V_{τ_2} respectively. Observe that $\Omega_{\mathbf{p}}^{(k-1)}$ restricted to the appropriate subset of variables is a refinement of the weight assignments $\mathrm{wt}_{\mathbf{p}}^{(\tau_1)}$ and $\mathrm{wt}_{\mathbf{p}}^{(\tau_2)}$ (as $|\tau_1|_\times$ or $|\tau_2|_\times$ could have been smaller than $k-1$). Nevertheless, if $\mathrm{wt}_{\mathbf{p}}^{(\tau_1)}$ and $\mathrm{wt}_{\mathbf{p}}^{(\tau_2)}$ are BIWAs for V_{τ_1} and V_{τ_2} respectively, then the following weight assignments

$$\begin{aligned}\mathrm{wt}_1 &: \bigcup_{i \in S_1} \mathbf{y}_i \to \mathbb{N}^k & \qquad \mathrm{wt}_2 &: \bigcup_{i \in S_2} \mathbf{y}_i \to \mathbb{N}^k\\ \mathrm{wt}_1 &: y_{ij} \mapsto \Omega_{\mathbf{p}}^{(k-1)}(y_{ij}) & \qquad \mathrm{wt}_2 &: y_{ij} \mapsto \Omega_{\mathbf{p}}^{(k-1)}(y_{ij})\end{aligned}$$

are also BIWAs for V_{τ_1} and V_{τ_2} respectively. It follows from Lemma 5.7, Lemma 5.6 and Lemma 5.5, that besides perhaps $\binom{w}{2} n^2$ primes $p \in P$, the weight assignment defined by

$$\begin{aligned}\mathrm{wt} &: \bigcup_{i \in S_1 \cup S_2} \mathbf{y}_i \to \mathbb{N}^{k+1}\\ \mathrm{wt}(y_{ij}) &= \begin{cases} (\mathrm{wt}_1(y_{ij}), 2^{(i-1)n+(j-1)} \bmod p) & \text{if } i \in S_1,\\ (\mathrm{wt}_2(y_{ij}), 2^{(i-1)n+(j-1)} \bmod p) & \text{if } i \in S_2,\end{cases}\\ &= (\Omega_{\mathbf{p}}^{(k-1)}(y_{ij}), 2^{in+j} \bmod p)\end{aligned}$$

is a BIWA for V_τ. For different τs in T_k there may a different set of $\binom{w}{2}n^2$ primes that we should exclude. But since the set P of primes is at least $\binom{w}{2}n^2d+1$, there is a prime $p \in P$ for which $\mathrm{wt}(y_{ij}) = (\Omega_{\mathbf{p}}^{(k-1)}, 2^{(i-1)n+(j-1)} \bmod p)$ is a BIWA for every V_τ where $\tau \in T_k$. By extending $\mathbf{p}$ by p in the last coordinate, this shows that there is a $\mathbf{p}' \in P^k$ such that for each $\tau \in T_k$, the weight assignment $\mathrm{wt}_{\mathbf{p}'}^{(\tau)}$ is a BIWA for V_τ.

To complete the inductive step, we also need to prove the same for $\tau \in T$ that are + gates with $|\tau|_\times = k$. Hence, there must be a $\times$ gate $\tau' \in T_k$ that is a descendant of τ such that the path from τ to τ' consists only of + gates. Once again, this forces $\mathrm{wt}_{\mathbf{p}}^{(\tau)} = \mathrm{wt}_{\mathbf{p}}^{(\tau')}$ and V_τ is a subspace of $V_{\tau'}$. Hence, by Claim 5.9 and Lemma 5.6, it follows that $\mathrm{wt}_{\mathbf{p}}^{(\tau)} = \mathrm{wt}_{\mathbf{p}}^{(\tau')}$ is a BIWA for V_τ as well. And that completes the proof of the inductive step.

Hence, if f is a polynomial computed by a preimage-width w UPT set-multilinear circuit of depth r, $\Omega_{\mathbf{p}}^{(r)}$ is a BIWA for V_{root}. Furthermore, by the prime number theorem, we know that the $\left(\binom{w}{2}n^2d+1\right)$-th prime cannot be bigger than $\left(\binom{w}{2}n^2d+1\right)^2$. Hence, the constructed BIWA is in fact a map

$$\Omega_{\mathbf{p}}^{(r)} : \mathbf{y} \to [M]^{r+1}$$

where $M \le \left(\binom{w}{2}n^2d+1\right)^2$. Therefore, by Lemma 5.3 and the polynomial identity lemma, if we pick a set $A \subseteq \mathbb{F}$ with $|A| > d \cdot \left(\binom{w}{2}n^2d+1\right)^2$, then

$$\mathcal{H} = \left\{ (b_{11}, \ldots, b_{dn}) \;:\; \mathbf{p} \in [M]^r \;,\; a_k \in A \;,\; b_{ij} = \prod_{k=1}^{r+1} a_k^{2^{(i-1)n+(j-1)} \bmod p_i} \right\}$$

is a hitting set for UPT set-multilinear circuits of preimage-width w and depth r, such that $|\mathcal{H}| = \mathrm{poly}((ndw)^r)$.

□

5.3 Poly-sized hitting sets for constant width UPT circuits

Theorem 1.3 (Hitting sets for known-shape low-width UPT circuits)**.** *Let $\mathcal{C}_{n,d,T,w}$ be the class of n-variate degree d non-commutative polynomials that are computable by UPT circuits of preimage-width at most w and underlying parse-tree shape as T. Over any field of zero or large characteristic, there is an explicit hitting set $\mathcal{H}_{n,d,T,w}$ of size $w^{O(\log d)}\,\mathrm{poly}(nd)$ for $\mathcal{C}_{n,d,T,w}$.*

The proof is an easy extension of the ideas from [GKS17], the details of which are in Section 9.2.

6 FewPT circuits

In this section we describe the black-box identity test for FewPT(k) circuits. The following lemma from [LLS19] shows that this class is equivalent to polynomials computed by sum of k UPT circuits (of possibly different shapes).

6.1 Preliminaries

Lemma 6.1. ([LLS19, Lemma 16]) *Let $f(\mathbf{x})$ be a polynomial computed by FewPT(k) circuit of preimage-width w. Then f can be equivalently computed by a sum of k UPT circuits of preimage-width w each.*

Like in [LLS19], we'll refer to this class by FewPT(k). We shall further qualify this notation to use FewPT(k)(w) to denote the class of circuits that is a sum of k UPT circuits of preimage-width w.

From this lemma, we can just work with FewPT(k) SML circuits. The proof largely follows the ideas of Gurjar, Korwar, Saxena and Thierauf [GKST17][4].

Notation

Let $\mathbf{y} = \mathbf{y}_1 \sqcup \cdots \sqcup \mathbf{y}_d$ be a partition of the variables and let $S = \{s_1, \ldots, s_p\}$ be a subset of $[d]$. Define the set of variables $\mathbf{y}_S = \mathbf{y}_{s_1} \cup \cdots \cup \mathbf{y}_{s_p}$ and the set of monomials $\mathbf{y}^S = \mathbf{y}_{s_1} \times \cdots \times \mathbf{y}_{s_p}$. Also, define $\mathbf{y}_{-S} = \mathbf{y} \setminus \mathbf{y}_S$ and $\mathbf{y}^{-S} = \mathbf{y}^{[d]\setminus S}$.

Definition 6.2 (Coefficient operator)**.** *Given a set-multilinear polynomial $f = \sum_{m \in \mathbf{y}^{[d]}} \alpha_m m$ of degree d, for $S \subseteq [d]$ and a monomial $m \in \mathbf{y}^S$, define $\mathrm{coeff}_m : \mathbb{F}[\mathbf{y}] \to \mathbb{F}[\mathbf{y}_{-S}]$ to be as follows.*

$$\mathrm{coeff}_m(f) = \sum_{m' \in \mathbf{y}^{-S}} \alpha_{(m \cdot m')} m'$$

where $\alpha_{(m\cdot m')}$ is the coefficient of mm' in f. $\diamondsuit$

Lemma 6.3. *Let $\mathbf{y} = \mathbf{y}_1 \sqcup \ldots \sqcup \mathbf{y}_d$ be a partition and $f(\mathbf{y})$ be a set-multilinear polynomial (with respect to the above partition) computed by a* UPT-SML *circuit of preimage-width w and underlying parse-tree shape T. Suppose $g(\mathbf{y})$ is another set-multilinear polynomial (under the same partition) that* cannot *be computed by a* UPT-SML *circuit of preimage-width w with the same shape T.*

Then, there exists $S \subseteq [d]$ and $R \in \mathbb{F}[\mathbf{y}_S]^{1 \times w'}$, and $P, Q \in \mathbb{F}[y_{-S}]^{w' \times 1}$ with $w' \leq w^2$ such that:

- *For each $i \in [w']$, there is a monomial $m_i \in \mathbf{y}^S$ such that the i-th element of P and Q is $\mathrm{coeff}_{m_i}(f)$ and $\mathrm{coeff}_{m_i}(g)$ respectively,*
- *there is a vector $\Gamma \in \mathbb{F}^{1 \times w'}$ of support size at most $w+1$ such that $\Gamma P = 0$ and $\Gamma Q \neq 0$,*
- *the coefficient space of R is full-rank. That is, if we interpret R as a matrix over $\mathbb{F}$ by listing each of its w' entries as a column vector of coefficients, then this matrix has full column-rank.*
- *the vector of polynomials R is simultaneously computable by a* UPT-SML *circuit of preimage-width at most w'.*

This lemma is a fairly natural and straightforward generalization of [GKST17, Lemma 4.5] and a proof of this is provided in Section 10.

[4][GKST17] constructed hitting sets for sums of ROABPs and we use similar techniques for sums of UPT circuits. Roughly speaking, if we have a class $\mathscr{C}$ that has a *characterizing set of dependencies* for which we know how to construct BIWAs, then we can also construct hitting sets for $\Sigma^k\mathscr{C}$.

Lemma 6.4. *Suppose $f(\mathbf{y})$ is a non-zero polynomial computed by a* FewPT(k) SML(w) *circuit. Suppose* $\mathrm{wt}:\mathbf{y}\to M^r$ *is a weight assignment that satisfies the following properties:*

- wt *is a BIWA for spaces of polynomials simultaneously computed by* UPT-SML *circuits of preimage-width at most $w(w+1)$,*
- *For any g in* FewPT$(k-1)$ SML$(w(w+1))$, *the polynomial $g(\mathbf{y}+\mathbf{t}^{\mathrm{wt}})\in\mathbb{F}(\mathbf{t})[\mathbf{y}]$ has a monomial with non-zero coefficient that depends on at most ℓ distinct variables in* $\mathbf{y}$.

Then, the polynomial $f(\mathbf{y}+\mathbf{t}^{\mathrm{wt}})$ has a monomial, depending on at most $\log(w(w+1))+\ell$ *distinct variables in* $\mathbf{y}$, *with a non-zero coefficient.*

This is essentially a restatement of [GKST17, Lemma 4.6, Lemma 4.8] and follows from their proof. Unravelling the recursion, we get the following corollary.

Corollary 6.5. *Let $f(\mathbf{y})$ be a non-zero polynomial that can computed by a* FewPT(k) SML(w) *circuit. Suppose* $\mathrm{wt}:\mathbf{y}\to M^r$ *is a BIWA for the class of polynomials simultaneously computed by* UPT-SML *circuits of preimage-width at most $w^{2^{O(k)}}$. Then, the polynomial $f(\mathbf{y}+\mathbf{t}^{\mathrm{wt}})\in\mathbb{F}(\mathbf{t})[\mathbf{y}]$ has a monomial with a non-zero coefficient that depends on at most $2^{O(k)}\log w$ variables in* $\mathbf{y}$.

Once we are guaranteed to retain a monomial of small-support, we can construct a hitting set by enumerating over all possible supports and applying the polynomial identity lemma (or apply standard generators such as the Shpilka-Volkovich generator [SV15]). This completes the proof of Theorem 1.2, which we restate below for convenience.

Theorem 1.2 (Hitting sets for circuits with few parse tree shapes)**.** *There is an explicit hitting set $\mathscr{H}_{d,n,s,k}$ of size at most $(s^{2^k}nd)^{O(\log d)}$ for the class of n-variate degree d homogeneous non-commutative polynomials in $\mathbb{F}\langle x_1,\ldots,x_n\rangle$ that are computed by non-commutative circuits of size at most s consisting of parse trees of at most k shapes.*

7 Separating ABPs from UPT circuits

This section contains the proofs of the separation between ABPs and UPT circuits. Recall the definition of the polynomial P_d (of degree $D=2^{d+1}-1$).

$$P_d(x_1,\ldots,x_m)=\sum_{\substack{\gamma\in[m]^D\\ \gamma\text{ is legal}}} x_{\gamma(v_1)}x_{\gamma(v_2)}\cdots x_{\gamma(v_D)}.$$

Upper bound

Lemma 4.2 (Upper bound)**.** *For every $m,d>0$, the polynomial $P_d(y_1,\ldots,y_m)$ can be computed by a non-commutative UPT circuit of size $O(m^2d)$ and preimage-width $O(m^2)$.*

Proof. Let $\mathcal{G}(d,\alpha)$ be the set of all legal colourings γ with v_{2^d} (root of T_d) satisfying $\gamma(v_{2^d}) = \alpha$. Now we define $P_{d,\alpha}(x_1,\ldots,x_m)$ as

$$P_{d,\alpha}(x_1,\ldots,x_m) = \sum_{\gamma\in\mathcal{G}(d,\alpha)} x_{\gamma(v_1)}x_{\gamma(v_2)}\cdots x_{\gamma(v_D)}.$$

Clearly, $P_d(x_1,\ldots,x_m) = \sum_{\alpha\in[m]} P_{d,\alpha}(x_1,\ldots,x_m)$. Therefore we can now recursively write

$$P_{d,\alpha}(x_1,\ldots,x_m) = \sum_{\beta\in[m]} P_{d-1,\beta}(x_1,\ldots,x_m)\cdot x_\alpha \cdot P_{d-1,(\alpha-_m\beta)}(x_1,\ldots,x_m), \tag{7.1}$$

where $\alpha -_m \beta = (\alpha-\beta) \bmod m$.

Now using (7.1) it is easy to see that if we have UPT circuits with $O(m^2k)$ gates *simultaneously* computing the polynomials $P_{k,\alpha}(x_1,\ldots,x_m)$ for any $k \le d-1$ and all $\alpha \in [m]$, then a UPT circuit with $O(m^2d)$ gates computing $P_d(x_1,\ldots,x_m)$ can be obtained and this follows directly by induction. Hence, repeated application of (7.1) yields a UPT circuit computing P_d of size $O(m^2d)$. It is also easy to see that the preimage-width of such a UPT circuit is at most $O(m^2)$, as there are no more than $O(m^2)$ computing polynomials of the same degree. $\square$

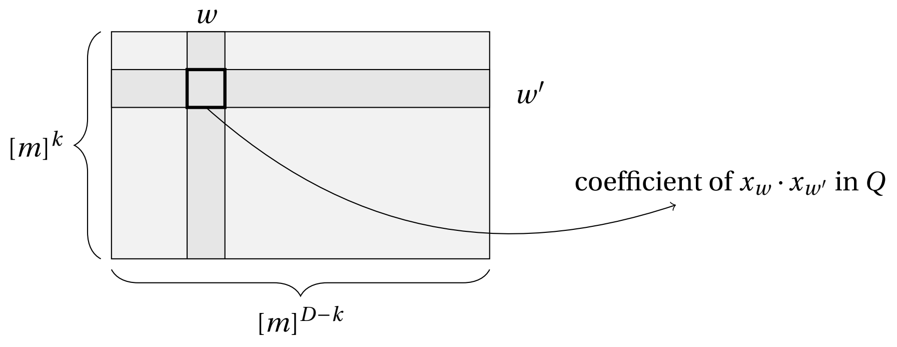
Lower bound

As mentioned earlier, much of the lower bound argument is exactly along the lines of the proof of [HY16]. The modifications required from their proof are quite minor but we present the proof here for completeness.

Theorem 4.3 (Lower bound)**.** *For every permutation $\sigma \in S_D$, any non-commutative ABP computing the polynomial $\Delta_\sigma(P_d)$ has width $m^{\Omega(d)}$.*

Proof. Let us fix some $\sigma \in S_D$ and let $Q(x_1,\ldots,x_m) = \Delta_\sigma(P_d)$. In order to show that Q requires ABPs of large width, it suffices to show that there exists some $0 \le k \le D$ for which the partial derivative matrix, given by

$M_k(Q) =$



has rank at least $m^{\Omega(d)}$. We shall prove this by exhibiting an $r \times r$ identity matrix as a submatrix in $M_k(Q)$ with $r = m^{\Omega(d)}$. The k that we will work with would be the number whose binary expansion is $10101\cdots$.

The relevance for this comes from the fact that the *edge boundary* of any subset $V_0 \subseteq T_d$ is with $|V_0| = k$ for such a k is reasonably large.

Definition 7.2 (Isoperimetric profile of graphs). *Given a graph $G = (V(G), E(G))$ and a subset of vertices $A \subseteq V(G)$, edge isoperimetric profile of G is given by the following function* $\mathrm{eip}(k)$ *defined by*

$$\mathrm{eip}_G(k) = \min\left\{\left|E(A, \overline{A})\right| \;:\; A \subseteq V(G), |A| = k\right\},$$

where $E(A, \overline{A})$ is the set of edges with one end-point in A and the other outside. $\Diamond$

Lemma 7.3. *[HY16] If $k \leq D$ is the number whose binary expansion is* $1010\cdots$, *then* $\mathrm{eip}_{T_d}(k) \geq \frac{d}{4}$.

The relevance for this would become apparent shortly, but let us proceed for now. If there is indeed an ABP for a shuffling of f, then the rows of $M_k(Q)$ is just a partial colouring of a subset $V_0 \subset T_d$ of size exactly k. Similarly, the columns of $M_k(Q)$ are partial colourings of $V_1 := T_d \setminus V_0$. Therefore $M_k(Q)_{(x_w, x_{w'})}$ is 1 only if the colouring of V_0 given by x_w and that of V_1 given by $x_{w'}$ together form a legal colouring of T_d. Hence the task of finding an $r \times r$ submatrix of $M_k(Q)$ reduces to finding colourings $C_1, C_2, \ldots, C_r$ of V_0 and colourings $C'_1, C'_2, \ldots, C'_r$ of V_1 such that the colouring $C_i \circ C'_j$ is legal if and only if $i = j$, for all $i, j \in [r]$.

We will need the notion of *pure nodes* (as defined by [HY16]).

Definition 7.4. *(*Pure *nodes). For $i \in \{0,1\}$, a non-leaf node v in V_i is called said to be* pure *if there is a path* $\Pi = (v, v_1, v_2, \ldots, v_k)$ *in T_d where v_k is a leaf that is a descendant of v, and* $\Pi \cap V_i = \{v\}$. $\Diamond$

There may be multiple witnesses v_k for the fact that v is a pure node. For each pure node, we shall assign one leaf arbitrarily as its *pure leaf*. It is easy to see that the pure leaves are distinct for each pure node.

Let the pure nodes in V_0 be P_0 and those in V_1 be P_1 and say $P := P_0 \cup P_1$. Let $\ell(P)$, $\ell(P_0)$ and $\ell(P_1)$ be the pure leaves of P, P_0 and P_1 respectively.

Lemma 7.5. ([HY16, Claim 11]) $|P| \geq \frac{|E(V_0, V_1)|}{4}$.

Without loss of generality, we may assume that P_0 is bigger than P_1 and the above lemma, in conjunction with Lemma 7.3, gives that $|P_0| \geq d/32$. We are now ready to define our colourings $C_1, \ldots, C_r$ and $C'_1, \ldots, C'_r$ for $r = m^{|P_0|} \geq m^{d/32}$.

Let L be the set of all leaves in T_d. For each $\mathbf{c}_i \in [m]^{|P_0|}$, define $\tilde{C}_i : T_d \to \mathbb{Z}_m$ obtained by assigning colour 1 to all leaves in $L \setminus \ell(P_0)$, assigning $\mathbf{c}_i$ to the leaves in $\ell(P_0)$ and extending it uniquely to the other vertices of T_d in order to make it legal. The partial colourings C_i and C'_i be the restriction of $\tilde{C}_i$ to V_0 and V_1 respectively.

Clearly, $C_i \circ C'_i = \tilde{C}_i$ and hence is a valid colouring. Now consider C_i and C'_j for $i \neq j$. There must exist some leaf $v \in \ell(P_0)$ that gets different colours in C_i and C_j and let u be the node in P_1 that v was a pure leaf of. We shall assume that u is *minimal* in the sense that any pure node $u' \in P_1$ that is a descendant has all its leaves identically coloured in C_i and C_j. But then, the colour of u in $\tilde{C}_i$ and in $\tilde{C}_j$ cannot be the same as exactly one leaf if u has a different colour in $\tilde{C}_i$ and $\tilde{C}_j$ respectively. This would then imply that C_i forces u to be given a colour different than what C'_j assigns and hence $C_i \circ C'_j$ is not legal.

Therefore, this shows that the matrix $M_k(Q)$ has an $r \times r$ identity submatrix with $r \geq m^{d/32}$. Therefore, any ABP computing Q must have width at least $m^{\Omega(d)}$. □

8 Exponential lower bound under any shuffling

Here we give an explicit polynomial that has polynomial sized arithmetic circuits but requires exponential sized UPT circuits under any shuffling. A version of the hard polynomial appears in [LMP19]. They show that the polynomial requires exponential sized UPT circuits and that it is efficiently computable by what are known as *skew circuits* (see [LMP19] for a formal definition). Here we extend the lower bound and show that it applies to any *shuffling* of the polynomial.

8.1 The polynomial

The hard polynomial we discuss is called the *moving palindrome* which is a variant of the *palindrome* polynomial. The palindrome polynomial of degree d on n variables, as known, is defined as follows.

$$\mathrm{Pal}_d(x_1,\ldots,x_n) := \sum_{w \in \{x_1,\ldots,x_n\}^{d/2}} w \cdot w^R$$

where w^R denotes the reverse of the word w.
Using this definition, we define the $(n+1)$-variate moving palindrome of degree D as follows.

$$\mathrm{Pal}_D^{\mathrm{mov}}(x_1,\ldots,x_n,z) := \sum_{0 \leq \ell \leq D/2} z^\ell \cdot \mathrm{Pal}_{\frac{D}{2}}(x_1,\ldots,x_n) \cdot z^{\frac{D}{2}-\ell}$$

8.2 The lower bound

Similar to the matrix M_k defined in Section 7 for a commutative polynomial, define a *partial derivative* matrix $M_{(i,p)}$ for a non-commutative polynomial g. Here the (w,w') entry of $M_{(i,p)}$ will be the coefficient of $w \times_p w'$ in g, where $\deg(w) = i$. We will show that $M_{(i,p)}$ for $\mathrm{Pal}_D^{\mathrm{mov}}$ has rank $n^{\Omega(D)}$ for a *range of types* (i,p), such that any UPT circuit computing any shuffling of $\mathrm{Pal}_D^{\mathrm{mov}}$ must admit at least one of those types. Then using the characterization from [LMP19], we will conclude the following theorem.

Theorem 8.1. *For any $\sigma \in S_D$, a UPT circuit computing $\Delta_\sigma(\mathrm{Pal}_D^{\mathrm{mov}})$ has $n^{\Omega(D)}$ gates.*

Proof. Let $2d$ be the degree of the palindrome, so that $\mathrm{Pal}_D^{\mathrm{mov}}$ has degree $D = 4d$. Also, let $P_\ell(\mathbf{x},z) = z^\ell\,\mathrm{Pal}_{2d}(\mathbf{x})z^{2d-\ell}$. Therefore $\mathrm{Pal}_D^{\mathrm{mov}} = \sum_{\ell=0}^{2d} P_\ell(\mathbf{x},z) = f(\mathbf{x},z)$ (say). For P_ℓ, and for $\ell < j_1, j_2 \leq 4d-\ell$, we will say that j_1 and j_2 are *dependent* with respect to P_ℓ if all monomials in P_ℓ contain the same variable in positions j_1 and j_2. It is easy to see that the criterion $j_1 + j_2 = 2(d+\ell)+1$ captures this relation. Define a *dependency* graph $G_\ell = (V, E_\ell)$ with $V = \{1, 2, \ldots, 4d\}$ such that $(j_1, j_2) \in E_\ell$ if and only if j_1 and j_2 are dependent with respect to P_ℓ. Let $G = (V, E)$ with $E = \cup_\ell E_\ell$.

If $[4d] = V_0 \sqcup V_1$ is a partition, let us define a matrix $\tilde{M}_{V_0,V_1}(f)$ to be the one where rows and columns are indexed by a partial assignment to the positions V_0 and V_1 respectively.

Claim 8.2. *Let $[4d] = V_0 \sqcup V_1$ be a partition of the positions, and suppose that for some $\ell \in \{0, \ldots, 2d\}$ we have t edges in E_ℓ crossing the cut (V_0, V_1) in G_ℓ. Then,* $\operatorname{rank}\left(\tilde{M}_{V_0,V_1}(f)\right) \geq n^t$.

Proof. In the polynomial P_ℓ, let $Z_\ell \subseteq [4d]$ be the positions that are fixed to z. Consider the submatrix of $\tilde{M}_{V_0,V_1}$ where $V_0 \cap Z_\ell$ and $V_1 \cap Z_\ell$ are assigned to z. Observe that this submatrix is precisely $\tilde{M}_{V_0',V_1'}(P_\ell)$ where $V_0' = V_0 \cap \overline{Z_\ell}$ and $V_1' = V_1 \cap \overline{Z_\ell}$.

If we have t edges crossing the cut (V_0', V_1') (none of the cut edges can be adjacent on Z_ℓ), then we have a size t matching in (V_0', V_1'). This means that fixing the variables in their V_0' end-points uniquely fixes their V_1' end-points. Hence, it is clear that we have an $n^t \times n^t$ identity submatrix and hence that the rank of $\tilde{M}_{V_0,V_1}(f)$ is at least n^t. □

The next claim shows that for any V_0 in a fairly wide range of sizes, there will always be some ℓ with G_ℓ exhibiting a large cut.

Claim 8.3. *For any set $V_0 \subseteq [4d]$ of size k with $\frac{d}{6} \leq k \leq \frac{d}{3}$, there is some $\ell \in \{0, \ldots, 2d\}$ such that $\Omega(d)$ edges in E_ℓ cross the cut (V_0, V_1).*

Proof. Let V_0 be a set of k positions with $k \leq \frac{d}{3}$. Let us partition the set of positions $V = \{1, 2, \ldots, 4d\}$ into blocks of size k,$(2d - 2k)$, and k, and k,$(2d - 2k)$, and k, labelled S_1, M_1 and T_1, and T_2, M_2 and S_2, respectively. We illustrate such a partition below.

S_1	M_1	T_1	T_2	M_2	S_2
1 $\quad k$	$(k+1)$ $\quad (2d-k)$	$(2d-k+1)$ $\quad 2d$	$(2d+1)$ $\quad (2d+k)$	$(2d+k+1)$ $\quad (4d-k)$	$(4d-k+1)$ $\quad D$

Now the possible choices for V_0 can be split into the following (possibly overlapping) cases:

1. $V_0 \cap T_1 \geq \frac{k}{8}$:
   Note that the degree of any vertex in T_1 is at least $(2d - k)$, and that every even (or odd) vertex in M_1 is connected to every odd (or even) vertex in S_2. Now $V_1 \cap M_1$ is at least $2d - k - (k - \frac{k}{8}) \geq 2(d - k)$. Total number of edges crossing (V_0, V_1) is therefore $\geq |(V_0 \cap T_1, V_1 \cap M_1)| \geq 2\left(\frac{1}{4} \times (d-k) \times \frac{k}{8}\right) = \Omega(dk)$. Therefore there exists an E_i that achieves the average $\Omega(k) = \Omega(d)$ edges crossing the cut (V_0, V_1).

2. $V_0 \cap S_1 \geq \frac{k}{4}$:
   Consider the neighbourhood of $V_0 \cup S_1$ due to E_0. All these positions are in T_1. If more than $\frac{k}{8}$ of them are in V_0 then case 1 applies. Else we get that $\geq \frac{k}{8}$ edges from E_0 cross (V_0, V_1).

3. $V_0 \cap M_1 \geq \frac{k}{4}$:
   Again, every even (or odd) position in M_1 is connected to every odd (or even) position in T_1, the degree of every position in M_1 is at least k, and $|V_1 \cap T_1| \geq \frac{k}{8}$. Therefore a total of $\Omega(k^2)$ edges cross (V_0, V_1), thereby again giving us that some E_i achieves $\Omega(d)$ edges crossing (V_0, V_1).

Since the other cases (with T_2, S_2, M_2) are symmetric to those discussed above, we can conclude the statement of the claim. □

In order to complete the proof, we just need to show that any UPT circuit computing a homogeneous degree d polynomial, there will be a gate of position-type (i, p) with $\frac{d}{6} \leq i \leq \frac{d}{3}$.

Lemma 8.4. *For all $0 < \alpha < \frac{1}{2}$, any UPT circuit (with fan-in 2 $\times$ gates) computing a polynomial of degree D contains a gate computing a degree i polynomial for some $\alpha D \leq i \leq 2\alpha D$.*

Sketch of Proof. Let C be a UPT circuit computing a degree D polynomial with multiplication gates of fan-in 2. Starting from the root of C, choose an arbitrary child at every addition gate and the child computing a higher degree polynomial at every multiplication gate. As the degree never drops to a fraction less than half in any step, we eventually reach an appropriate gate. □

Now Lemma 8.4 tells us that for any UPT circuit computing $\Delta_\sigma(\mathrm{Pal}_D^{\mathrm{mov}})$, will have a gate of position-type (i, p) with $\frac{D}{24} \leq i \leq \frac{D}{12}$. We can then apply Claim 8.3 and then Claim 8.2 to obtain an $n^{\Omega(D)}$ lower bound on the number of gates in C. □

9 Hitting sets for UPT circuits

9.1 Commutative analogue of UPT circuits

Consider substitution map $\Phi : \{\mathbf{x}\} \to \mathbb{F}[y_{1,1}, \ldots, y_{d,n}]^{(d+1)\times(d+1)}$ given by

$$\Phi(x_i) = \begin{bmatrix} 0 & y_{1,i} & 0 & \ldots & 0 & 0 \\ 0 & 0 & y_{2,i} & \ldots & 0 & 0 \\ 0 & 0 & 0 & \ldots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \ldots & 0 & y_{d,i} \\ 0 & 0 & 0 & \ldots & 0 & 0 \end{bmatrix}, \quad \text{for all } i \in [n].$$

To understand the effect of Φ on a homogeneous non-commutative polynomial $f(\mathbf{x})$ of degree d, define $\Psi : \mathbb{F}\langle\mathbf{x}\rangle_{\deg=d} \to \mathbb{F}[y_{1,1}, \ldots, y_{d,n}]$ as the unique $\mathbb{F}$-linear map given by
$\Psi : x_{w_1} \cdots x_{w_d} \mapsto y_{1,w_1} \cdots y_{d,w_d}$.

Lemma 9.1 ([FS13])**.** *Let $f = \sum_w a_w x_w \in \mathbb{F}\langle \mathbf{x} \rangle$ be a homogeneous degree d non-commutative polynomial. Then, f under the substitution map Φ (defined above) is given by*

$$f \circ \Phi = f(\Phi(x_1), \ldots, \Phi(x_n)) = \begin{bmatrix} 0 & \cdots & 0 & \Psi(f) \\ 0 & \cdots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 \end{bmatrix}_{(d+1)\times(d+1)} \qquad \square$$

Similar to the above definition of Ψ, we define a *shifted* version of it called Ψ_a (for a parameter $a \in \mathbb{N}$) as $\Psi_a : x_{w_1} \cdots x_{w_d} \mapsto y_{a+1,w_1} \cdots y_{a+d,w_d}$.

Observation 9.2. *If $f \in \mathbb{F}\langle \mathbf{x} \rangle_{\deg=d_1}$ and $g \in \mathbb{F}\langle \mathbf{x} \rangle_{\deg=d_2}$, then for any $a \in \mathbb{N}$, we have $\Psi_a(f \cdot g) = \Psi_a(f) \cdot \Psi_{a+d_1}(g)$.*

In the case of [FS13], when f was computable by non-commutative ABPs, they showed that $\Psi(f)$ is computable by an ROABP. In our setting of non-commutative UPT circuits, the following is the commutative analogue.

Observation 9.3. *Let C be a UPT circuit computing a polynomial $f \in \mathbb{F}\langle \mathbf{x} \rangle$ of size s and depth r. Consider the commutative circuit C' where each leaf variable of type $(1, p)$ that is labelled by x_i is replaced by $y_{p+1,i}$. Then the circuit C' computes $\Psi(f)$ and is UPT and set-multilinear with respect to $\mathbf{y} = \mathbf{y}_1 \sqcup \cdots \sqcup \mathbf{y}_d$ where $\mathbf{y}_i = \{y_{i,j} \ : \ j \in [n]\}$.*

BIWAs for subspaces and products

Lemma 5.6 (BIWA for subspaces)**.** *Say V is a vector space of polynomials and suppose* wt *is a BIWA for V. Then, if V' is a subspace of V, then* wt *is a BIWA for V' as well.*

Proof. If B is a monomial basis of V that is isolated by wt, then the columns indexed by B span the column space of V' as well. Starting with the columns of V' indexed by B, pick a *minimum weight basis* B' according to wt, so that any column of V' that is outside B' is spanned by lower weight monomials in B'. By definition wt is a BIWA of V' isolating B', as all columns in B' get distinct weights and every column outside B' is spanned by lower weight columns in B'. $\square$

Lemma 5.7 (BIWA for variable disjoint products)**.** *Say $V_1 \subseteq \mathbb{F}[\mathbf{y}]$ and $V_2 \subseteq \mathbb{F}[\mathbf{z}]$ are two vector spaces of polynomials over disjoint sets of variables, and of dimension at most s. Suppose*

$$\mathrm{wt}_1 : \mathbf{y} \to \mathbb{N}^k$$
$$\mathrm{wt}_2 : \mathbf{z} \to \mathbb{N}^k$$

are BIWAs for V_1 and V_2 isolating bases B_1 and B_2 respectively. If $w : \mathbf{y} \cup \mathbf{z} \to \mathbb{N}$ is a weight assignment that separates $B_1 \cdot B_2 = \{m_1 m_2 \,:\, m_1 \in B_1\,,\, m_2 \in B_2\}$. *Then the weight assignment defined by*

$$\begin{aligned}
&\mathrm{wt} : \mathbf{y} \cup \mathbf{z} \to \mathbb{N}^{k+1}\\
&\mathrm{wt} : y_i \longmapsto (\mathrm{wt}_1(y_i), w(y_i)) \quad \text{for all } y_i \in \mathbf{y},\\
&\mathrm{wt} : z_i \longmapsto (\mathrm{wt}_2(z_i), w(z_i)) \quad \text{for all } z_i \in \mathbf{z},
\end{aligned}$$

is a BIWA for $V = V_1 \cdot V_2 = \mathrm{span}\left\{f \cdot g \,:\, f \in V_1\,,\, g \in V_2\right\}$.

Proof. Observe that by the definition of wt, $\mathrm{wt}(m_1 \cdot m_2) = (\mathrm{wt}_1(m_1) + \mathrm{wt}_2(m_2), w(m_1 \cdot m_2))$ for any $m \in \mathrm{Mons}(\mathbf{y})$ and $m' \in \mathrm{Mons}(\mathbf{z})$.

If V_1 and V_2 are expressed as matrices (with the generators listed as rows), then the matrix corresponding to V is just $V_1 \otimes V_2$, the tensor product. Let $B_1 = \{m_1, \dots, m_r\}$ and $B_2 = \left\{m'_1, \dots, m'_s\right\}$. We shall prove that the weight assignment wt is a BIWA that isolates the natural spanning set $B = B_1 \cdot B_2 = \left\{m_i m'_j \,:\, i \in [r]\,,\, j \in [s]\right\}$. Firstly, note that all the elements of B have distinct weights due to the presence of the last coordinate from wt, which separates the rs monomials in $B_1 \cdot B_2$.

Now suppose $\tilde{m} = m \cdot m' \notin B$ for $m \in \mathrm{Mons}(\mathbf{y})$ and $m' \in \mathrm{Mons}(\mathbf{z})$ and say without loss of generality $m \notin B_1$. The column indexed by $\tilde{m}$ in $V_1 \cdot V_2$ is just the tensor product of the columns indexed by m in V_1 and the column indexed by m' in V_2. But since wt_1 is basis isolating for V_1, the column of V_1 indexed by m can be expressed as a linear combination of lower weight terms.

$$\begin{aligned}
V_{1,m} &= \sum_{\mathrm{wt}_1(m_i) \prec \mathrm{wt}_1(m)} a_i \cdot V_{1,m_i}\\
\implies V_{\tilde{m}} = V_{1,m} \otimes V_{2,m'} &= \sum_{\mathrm{wt}_1(m_i) \prec \mathrm{wt}_1(m)} a_i \cdot \left(V_{1,m_i} \otimes V_{2,m'}\right)\\
&= \sum_{\mathrm{wt}_1(m_i) \prec \mathrm{wt}_1(m)} a_i \cdot V_{m_i m'}
\end{aligned}$$

But notice that $\mathrm{wt}_1(m_i) \prec \mathrm{wt}_1(m)$ also implies that $\mathrm{wt}(m_i m') \prec \mathrm{wt}(m m')$. Therefore, (repeating this argument on m' if $m' \notin B_2$) we can write any column with index outside B as a linear combination of columns of smaller weight in B. Hence, wt is indeed a BIWA for V that isolates B. □

9.2 Constant width UPT circuits

In this subsection we prove the existence of a poly(n,d) hitting set for UPT circuits of constant preimage-width computing n-variate degree-d polynomials, when the *shape* of the circuit is known. The proof is an easy extension of the ideas of [GKS17] to the UPT-SML circuits regime. We will construct a univariate substitution map that preserves its nonzeroness and has degree poly(n,d), which will imply a hitting set naturally.

Say $\mathbf{y} = \mathbf{y}_1 \sqcup \cdots \sqcup \mathbf{y}_d$ and let $f(\mathbf{y})$ be an nd-variate degree d polynomial computable by a UPT-SML circuit (with respect to the above partition) of constant preimage-width. From Observation 3.8, we may assume that the circuit has depth $\log d$. We will need the following lemma for bivariate polynomials over *large* fields.

Lemma 9.4. ([GKS17, Lemma 3.2]) *Let $f(y_1, y_2) = \sum_{i=1}^{w} u_i(y_1)v_i(y_2)$ be a nonzero bivariate polynomial of degree d over $\mathbb{F}$. If $\mathrm{char}(\mathbb{F}) = 0$ or $\mathrm{char}(\mathbb{F}) > d$, then $f(t^w, t^{w-1} + t^w) \neq 0$.*

Suppose $f(\mathbf{y})$ is computable by a circuit C that has shape T. Define the set of variables $\mathbf{t} = \{t_\tau : \tau \in T\}$. We will begin by substituting $t_{\tau_i}^j$ for every y_{ij} where the leaf in C computing polynomials over $\mathbf{y}_i$ corresponds to τ_i in T. As long as we can, we will pick a multiplication gate τ that has its left and right children (say τ_L and τ_R) computing univariate polynomials in t_{τ_L} and t_{τ_R} respectively; and then substitute $t_{\tau_L} \leftarrow t_\tau^w$ and $t_{\tau_R} \leftarrow t_\tau^w + t_\tau^{w-1}$. Let us call this substitution Φ_τ.

Lemma 9.5. *Consider the above iterative process of substituting some of the $\mathbf{y}_i$'s by suitable polynomials in $\mathbf{t}$. Let $\tilde{\Phi}(f) = \tilde{f}(\mathbf{t}, \mathbf{y}) \neq 0$ be the polynomial just before applying the substitution Φ_τ. Then $\tilde{f}' = \Phi_\tau(\tilde{f}) := \tilde{f}(t_{\tau_L} \leftarrow t_\tau^w, t_{\tau_R} \leftarrow t_\tau^w + t_\tau^{w-1}) \neq 0$.*

Proof. From (3.4), we have

$$\begin{aligned} f &= \sum_{u \sim \tau} [u] \cdot [\mathrm{root} : u] \\ &= \sum_{u \sim \tau} [u_L] \cdot [u_R] \cdot [\mathrm{root} : u], \\ \Longrightarrow \tilde{\Phi}(f) &= \sum_{u \sim \tau} a_u(t_{\tau_L}) \cdot b_u(t_{\tau_R}) \cdot h_u(\mathbf{y}, \mathbf{t} \setminus t_{\tau_L}, t_{\tau_R}) \neq 0. \end{aligned}$$

We may treat $\tilde{\Phi}(f)$ as a bivariate polynomial in t_{τ_L}, t_{τ_R} over the field $\mathbb{F}(\mathbf{t} \setminus \{t_{\tau_L}, t_{\tau_R}\})$ and apply Lemma 9.4 to conclude that $\Phi_\tau(\tilde{\Phi}(f))$ will be nonzero if and only if $\tilde{\Phi}(f)$ was nonzero. □

Now for every leaf node in T, create a sequence which we will call its *signature*, by walking down from the root to the leaf. Every time we pick the left child, we append L to the signature and every time we pick the right child, we append R. For $\tau \in T$, call the sequence $\mathrm{sig}_\tau = (a_1\ a_2\ \cdots\ a_r)$. Let t be a fresh variable and τ_i be the node corresponding to $\mathbf{y}_i$. Define

$$\begin{aligned} &\Phi_L : t \mapsto t^w \quad , \quad \Phi_R : t \mapsto t^w + t^{w-1} \\ &\Psi : \mathbf{y} \to \mathbb{F}[t] \\ &\Psi : y_{ij} \mapsto \Phi_{a_1} \circ \Phi_{a_1} \circ \cdots \circ \Phi_{a_r}(t^j) \end{aligned}$$

where $(a_1 \cdots a_r) = \mathrm{sig}_{\tau_i}$. Observe that the procedure described above essentially executes the substitution Ψ on $\mathbf{y}$. We can then infer from Lemma 9.5 that for any $f(\mathbf{y})$ computable by UPT-SML circuits, $f(\mathbf{y}) \neq 0 \iff f(\Psi(\mathbf{y})) \neq 0$. This gives us the following theorem.

Theorem 9.6. *Let $f(\mathbf{y})$ be a polynomial computed by an* UPT-SML *circuit of width w and depth r. Consider the following substitution $\Psi : \mathbf{y} \to \mathbb{F}[t]$ given by*

$$\Psi : y_{ij} \mapsto \Phi_{a_1} \circ \Phi_{a_2} \circ \cdots \circ \Phi_{a_r}(t^j),$$

where the signature *of the part $\mathbf{y}_i$ is $a_1 a_2 \cdots a_r$. Then $f(\mathbf{y})$ is non-zero if and only if $f(\Psi(\mathbf{y}))$ is non-zero.*

Now since the depth of the circuit is at most $O(\log d)$, if the width is constant, then the final degree of $f(\Psi(\mathbf{y}))$ is at most $O(nw^{O(\log d)})$, which is $\mathrm{poly}(n,d)$ if $w = O(1)$. This finishes the proof of Theorem 1.3.

10 Hitting sets for FewPT circuits

We will need the following fact about coefficient operators (defined in Definition 6.2).

Observation 10.1 (Coefficients of UPT circuits are also UPT circuits)**.** *Suppose $f(\mathbf{y})$ is a homogeneous degree d polynomial that is computable by a UPT set-multilinear circuit with respect to $\mathbf{y} = \mathbf{y}_1 \sqcup \cdots \sqcup \mathbf{y}_d$ of preimage-width w. If $S \subseteq [d]$ and m is any monomial in $\mathbf{y}^S$, then the polynomial $\mathrm{coeff}_m(f)$ can also be computed by a UPT set-multilinear circuit of preimage-width w.*

Sketch of Proof. Since the UPT-SML circuit C can be made canonical without loss of generality, we only need to set the corresponding leaves in $\mathbf{y}_S$ as 0 or 1 depending on whether the variable appears in m. □

The following is an analogue of [GKST17, Lemma 4.5].

Lemma 6.3. *Let $\mathbf{y} = \mathbf{y}_1 \sqcup \ldots \sqcup \mathbf{y}_d$ be a partition and $f(\mathbf{y})$ be a set-multilinear polynomial (with respect to the above partition) computed by a* UPT-SML *circuit of preimage-width w and underlying parse-tree shape T. Suppose $g(\mathbf{y})$ is another set-multilinear polynomial (under the same partition) that* cannot *be computed by a* UPT-SML *circuit of preimage-width w with the same shape T.*

Then, there exists $S \subseteq [d]$ and $R \in \mathbb{F}[\mathbf{y}_S]^{1\times w'}$, and $P, Q \in \mathbb{F}[y_{-S}]^{w'\times 1}$ with $w' \le w^2$ such that:

- *For each $i \in [w']$, there is a monomial $m_i \in \mathbf{y}^S$ such that the i-th element of P and Q is $\mathrm{coeff}_{m_i}(f)$ and $\mathrm{coeff}_{m_i}(g)$ respectively,*
- *there is a vector $\Gamma \in \mathbb{F}^{1\times w'}$ of support size at most $w+1$ such that $\Gamma P = 0$ and $\Gamma Q \neq 0$,*
- *the coefficient space of R is full-rank. That is, if we interpret R as a matrix over $\mathbb{F}$ by listing each of its w' entries as a column vector of coefficients, then this matrix has full column-rank.*
- *the vector of polynomials R is simultaneously computable by a* UPT-SML *circuit of preimage-width at most w'.*

Proof. For an $S \subseteq [d]$, let $\mathbf{y}^S = \{m_1, \ldots, m_r\}$ and $\mathbf{y}^{-S} = \{n_1, \ldots, n_t\}$ in some order. Define $M_{f,S} \in \mathbb{F}^{r\times t}$ such that $M_{f,S}(i,j)$ is the coefficient of n_j in $\mathrm{coeff}_{m_i}(f)$. Note that the i^{th} row of $M_{f,S}$ is the polynomial $\mathrm{coeff}_{m_i}(f)$ written in the coefficient vector form.

For a type τ in a tree T, S_τ will denote the set of leaves of the node τ in T. Consequently, we will also use just $M_{f,\tau}$ to mean M_{f,S_τ}. We will denote by $B_{f,\tau}$ a set of monomials from $\mathbf{y}^{S_\tau}$ such that the rows indexed by them in $M_{f,S}$ will form a basis of the rows of $M_{f,S}$. Note that if τ has children τ_1, τ_2, then we can ensure that our choice of $B_{f,\tau}$ satisfies $B_{f,\tau} \subseteq B_{f,\tau_1} \times B_{f,\tau_2}$ as the latter is clearly a spanning set. Using such a basis $B_{f,\tau}$, we can then write down a set of dependencies as below corresponding to f and τ.

$$\forall m \in \mathbf{y}^{S_\tau} : \operatorname{coeff}_m(f) = \sum_{m' \in B_{f,\tau}} \gamma_{m,m'} \operatorname{coeff}_{m'}(f). \tag{10.2}$$

Using this, we can rewrite f in the following way for any $\tau \in T$.

$$\begin{aligned} f &= \sum_{m_k \in \mathbf{y}^{S_\tau}} m_k \left(\sum_{m_i' \in B_{f,\tau}} \gamma_{i,k} \operatorname{coeff}_{m_i'}(f) \right) = \sum_{m_i' \in B_{f,\tau}} \left(\sum_{m_k \in \mathbf{y}^{S_\tau}} \gamma_{i,k} m_k \right) \operatorname{coeff}_{m_i'}(f) \\ f &= \sum_{m_i' \in B_{f,\tau}} u_i(\mathbf{y}^{S_\tau}) \operatorname{coeff}_{m_i'}(f) \quad \text{for some } u_i \in \mathbb{F}[\mathbf{y}_{S_\tau}]. \end{aligned} \tag{10.3}$$

Suppose $\tau \in T$ has two children τ_1 and τ_2 that share the same dependencies for g as well. That is,

$$\begin{aligned} f &= \sum_{m_i' \in B_{f,\tau_1}} u_i(\mathbf{y}^{S_{\tau_1}}) \operatorname{coeff}_{m_i'}(f), & f &= \sum_{n_j' \in B_{f,\tau_2}} v_j(\mathbf{y}^{S_{\tau_2}}) \operatorname{coeff}_{n_j'}(f), \\ g &= \sum_{m_i' \in B_{f,\tau_1}} u_i(\mathbf{y}^{S_{\tau_1}}) \operatorname{coeff}_{m_i'}(g), & g &= \sum_{n_j' \in B_{f,\tau_2}} v_j(\mathbf{y}^{S_{\tau_2}}) \operatorname{coeff}_{n_j'}(g). \end{aligned}$$

Combining them (and renaming the variables by dropping the $'$s), we get

$$\begin{aligned} f &= \sum_{(m_i,n_j) \in B_{f,\tau_1} \times B_{f,\tau_2}} u_i(\mathbf{y}^{S_{\tau_1}}) v_j(\mathbf{y}^{S_{\tau_2}}) \cdot \operatorname{coeff}_{m_i \cdot n_j}(f), \\ g &= \sum_{(m_i,n_j) \in B_{f,\tau_1} \times B_{f,\tau_2}} u_i(\mathbf{y}^{S_{\tau_1}}) v_j(\mathbf{y}^{S_{\tau_2}}) \cdot \operatorname{coeff}_{m_i \cdot n_j}(g). \end{aligned}$$

Observe that if for all $m \in B_{f,\tau_1} \times B_{f,\tau_2}$ we have

$$\operatorname{coeff}_m(f) = \sum_{m' \in B_{f,\tau}} \gamma_{m,m'} \operatorname{coeff}_{m'}(f) \quad , \quad \operatorname{coeff}_m(g) = \sum_{m' \in B_{f,\tau}} \gamma_{m,m'} \operatorname{coeff}_{m'}(g),$$

then this also forces that by (10.3), for τ:

$$f = \sum_{m_i \in B_{f,\tau}} u_i'(\mathbf{y}^{S_\tau}) \operatorname{coeff}_{m_i}(f) \quad , \quad g = \sum_{m_i \in B_{f,\tau}} u_i'(\mathbf{y}^{S_\tau}) \operatorname{coeff}_{m_i}(g).$$

Since g is *not* computable by a UPT-SML circuit with underlying shape T this cannot happen for all $\tau \in T$. Let us pick the lowest τ (closest to the leaves; and say its children are τ_1, τ_2) such that for some

$m \in B_{f,\tau_1} \times B_{f,\tau_2}$ we have

$$\begin{aligned} \operatorname{coeff}_m(f) &= \sum_{m' \in B_{f,\tau}} \gamma_{m,m'} \operatorname{coeff}_{m'}(f), \\ \operatorname{coeff}_m(g) &\neq \sum_{m' \in B_{f,\tau}} \gamma_{m,m'} \operatorname{coeff}_{m'}(g). \end{aligned} \tag{10.4}$$

The choice of the vector of polynomials is now clear. If $w' = |B_{f,\tau_1}| \cdot |B_{f,\tau_2}| \leq w^2$, then

$$\begin{aligned} R &:= \left(u_i(\mathbf{y}^{S_{\tau_1}}) v_j(\mathbf{y}^{S_{\tau_2}}) \,:\, (m_i, n_j) \in B_{f,\tau_1} \times B_{f,\tau_2} \right) \in \mathbb{F}[\mathbf{y}_{S_\tau}]^{1 \times w'} \\ P &:= \left(\operatorname{coeff}_{m_i \cdot n_j}(f) \,:\, (m_i, n_j) \in B_{f,\tau_1} \times B_{f,\tau_2} \right)^T \in \mathbb{F}[\mathbf{y}_{-S_\tau}]^{w' \times 1} \\ Q &:= \left(\operatorname{coeff}_{m_i \cdot n_j}(g) \,:\, (m_i, n_j) \in B_{f,\tau_1} \times B_{f,\tau_2} \right)^T \in \mathbb{F}[\mathbf{y}_{-S_\tau}]^{w' \times 1}. \end{aligned}$$

It is clear from the definition that the vectors P and Q are made up of coefficients of f and g. Also, (10.4) provides a suitable vector Γ of support at most $w+1$ such that $\Gamma P = 0$ but $\Gamma Q \neq 0$.
It follows that the coefficient space of R is full-rank as the sets of polynomials $\{u_i \,:\, i \in B_{f,\tau_1}\}$ and $\{v_j \,:\, j \in B_{f,\tau_2}\}$ are linearly independent and are on disjoint sets of variables.
We only need to show that every entry of R can also be computed by a UPT-SML circuit of preimage-width at most w^2. To see this, observe that the set $\{\operatorname{coeff}_m(f) \,:\, m \in \mathbf{y}^{-S_{\tau_1}}\}$ is spanned by the set $\{u_i(\mathbf{y}^{S_{\tau_1}}) \,:\, i \in B_{f,\tau_1}\}$, and similarly the set $\{\operatorname{coeff}_n(f) \,:\, n \in \mathbf{y}^{-S_{\tau_2}}\}$ is spanned by $\{v_j(\mathbf{y}^{S_{\tau_2}}) \,:\, j \in B_{f,\tau_2}\}$. Since the dimension of these spaces is at most w, it follows that each $u_i(\mathbf{y}^{S_{\tau_1}})$ can be written as a linear combination of at most w many $\operatorname{coeff}_m(f)$'s, and similarly each $v_j(\mathbf{y}^{S_{\tau_2}})$. Observation 10.1 shows that each of the coefficient polynomials can also be computed by UPT-SML circuits of preimage-width at most w. Thus, by computing each of the u_i's and v_j's separately, and then taking all w^2 products, we have a UPT-SML circuit of preimage-width at most w^2 that simultaneously computes all the entries of R. □

11 Finer analysis of constant width UPT circuits

We now present some results comparing the power of constant width UPT circuits to variants of non-commutative ABPs and formulas. We recall the following statements that we shall use in the rest of this section.

- Nisan [Nis91] showed that for any homogeneous non-commutative polynomial f the *width* of a (homogeneous) ABP computing it, in the layer i, is exactly equal to the rank of the partial derivative matrix of f for degree i.

- Lagarde et al. [LMP19] show that in the smallest UPT circuit of shape T computing a polynomial f, the number of gates of a type $\tau \in T$ is equal to the rank of the *generalised partial derivative matrix* (formally defined in the proof of Theorem 1.5) for the type τ.

Thus, the ranks of the appropriate generalised partial derivative matrices for f characterize the ABP complexity and the UPT complexity of f. We will drop the term 'generalised' for the remainder of this discussion.

11.1 Constant width ABPs

Note that an ABP of width w can directly be converted to a UPT circuit of preimage-width $O(w^2)$. Also, for an $m = 2$ the polynomial family $\{P_d(y_1,\dots,y_m)\}$ from Section 7 yields the following lemma using in Lemma 4.2 and Theorem 4.3.

Lemma 11.1. *There is a family of non-commutative bivariate degree n polynomials $\{Q_n\}$[5] such that for all large enough n, there is a UPT circuit for Q_n with size $O(\log n)$ and preimage-width $O(1)$, and for any shuffling of Q_n, an ABP computing it has width $n^{\Omega(1)}$.* □

Putting these observations together we get that constant width ABPs are strictly weaker than constant width UPT circuits, even under shufflings.

11.2 General non-commutative ABPs and formulas

Constant width UPT circuits are weaker. For a large enough constant c, consider the following family of a non-commutative variant of the elementary symmetric polynomial family.

$$\mathrm{NESym}_n(x_1,\dots,x_n) = \sum_{1\le i_1<\cdots<i_c\le n} x_{i_1}x_{i_2}\cdots x_{i_c}$$

Lemma 11.2. *For all large enough n, the polynomial* NESym_n *is computable by a* formula *of size $O(n^c)$. Also, any shuffling of* NESym_n *requires UPT circuits of width $n^{\Omega(1)}$.*

Proof Sketch. Note that for any bipartition of the positions, the corresponding *partial derivative matrix* of $\mathrm{NESym}_n(\mathbf{x})$ is a *set disjointness* matrix. Therefore it has full row rank, which is at least $n^{\Omega(1)}$ for any bipartition with the smaller part having size between $c/3$ and $2c/3$. Once we have this, it is clear that any shuffling of NESym_n continues to have this property. Thus, we have that no shuffling of NESym_n has a UPT circuit of constant preimage-width.

Also, NESym_n has exactly $\binom{n}{c}$ monomials and therefore has poly(n) sized formulas. □

ABPs are weaker (without shufflings). Next, let us consider family of the bivariate palindrome polynomials of degree $2n$.

$$\mathrm{Pal}_n(x_1,x_2) = \sum_{(i_1,\dots,i_n)\in[2]^n} \left(x_{i_1}x_{i_2}\cdots x_{i_n}\right)\cdot\left(x_{i_n}\cdots x_{i_2}x_{i_1}\right)$$

[5]The polynomial Q_n is P_d for $d = \lfloor \log n \rfloor$.

The following lemma is now easy to verify using the discussions in rest of the paper about the palindrome family.

Lemma 11.3. *For all large enough n, $P_n(x_1, x_2)$ has a UPT circuit of size* $\mathrm{poly}(n)$ *and constant preimage-width. Also, any ABP computing $P_n(x_1, x_2)$ requires size $2^{\Omega(d)} = n^{\omega(1)}$.* □

Formulas simulate shufflings of constant width UPT circuits. The following easy lemma tells us that any polynomial that has a constant width UPT circuit has a shuffling that is efficiently computable by a formula.

Lemma 11.4. *If f_n is an n-variate degree d polynomial that has a UPT circuit of size s and preimage-width w, then there is a shuffling $\Delta_\sigma(f)$ of f_n that has formulas of size* $\mathrm{poly}(s, w^{O(\log d)})$.

Proof Sketch. We know from Theorem 1.4 that there is a shuffling of f, say f', that has a UPT circuit of preimage-width $O(w^2)$ and depth $O(\log d)$. Let the shape of the new circuit be T. Now for any type $\tau \in T$ and any gate $g \sim \tau$, the number of paths from g to the root is at most $w^{\text{depth}} = w^{O(\log d)}$. Thus a careful replication of gates in the depth-reduced UPT circuit will give us a formula of the required size. □

12 Open problems

An interesting open problem (at least to us) is whether we can give non-trivial hitting sets for the class of *non-commutative skew circuits*. Lagarde, Limaye and Srinivasan [LLS19] provide a white-box PIT in some restricted settings when the skew circuits are somewhat closer to UPT (with some restriction on what sort of parse trees they can have) but removing this restriction would be a great step forward.

Another issue is that the current construction of hitting sets for FewPT circuits (which build on [GKST17]) incurs quasipolynomial losses at two different places. The first is in the construction of the *basis isolating weight assignment (BIWA)*, and we only know to construct that using quasipolynomially large weights. The other is in a brute-force enumeration of all monomials of support $O(\log s)$. As a result, even if at a later day we have a construction of a BIWA with polynomially large weights, this proof would still only yield a quasipolynomially large hitting set for FewPT circuits. It would be interesting to see if this brute-force enumeration could be circumvented.

Acknowledgements

We thank the organizers of the NMI Workshop on Arithmetic Complexity 2017 where we learned of the circuit classes that we study in this paper. We thank Nutan Limaye and Srikanth Srinivasan for numerous discussions that eventually led to these results. We thank Rohit Gurjar for pointing out a subtlety in a previous draft of this paper, and also thank Amir Shpilka for inviting RS to Tel Aviv University (where this discussion took place).

References

[AB03] Manindra Agrawal and Somenath Biswas. Primality and identity testing via Chinese remaindering. *Journal of the ACM*, 50(4):429–443, 2003. Preliminary version in the *40th Annual IEEE Symposium on Foundations of Computer Science (FOCS 1999)*.

[AGKS15] Manindra Agrawal, Rohit Gurjar, Arpita Korwar, and Nitin Saxena. Hitting-Sets for ROABP and Sum of Set-Multilinear Circuits. *SIAM Journal of Computing*, 44(3):669–697, 2015. Pre-print available at `arXiv:1406.7535`.

[Agr05] Manindra Agrawal. Proving Lower Bounds Via Pseudo-random Generators. In *Proceedings of the 25th International Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2005)*, pages 92–105, 2005.

[AJMV98] 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. `eccc:TR95-043`.

[AR16] Vikraman Arvind and S. Raja. Some Lower Bound Results for Set-Multilinear Arithmetic Computations. *Chicago Journal of Theoretical Computer Science*, 2016.

[ASS13] Manindra Agrawal, Chandan Saha, and Nitin Saxena. Quasi-polynomial hitting-set for set-depth-Δ formulas. In *Proceedings of the 45th Annual ACM Symposium on Theory of Computing (STOC 2013)*, pages 321–330, 2013. `eccc:TR12-113`.

[BS83] Walter Baur and Volker Strassen. The Complexity of Partial Derivatives. *Theoretical Computer Science*, 22:317–330, 1983.

[DL78] Richard A. DeMillo and Richard J. Lipton. A Probabilistic Remark on Algebraic Program Testing. *Information Processing Letters*, 7(4):193–195, 1978.

[FS13] Michael A. Forbes and Amir Shpilka. Quasipolynomial-Time Identity Testing of Non-commutative and Read-Once Oblivious Algebraic Branching Programs. In *Proceedings of the 54th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2013)*, pages 243–252, 2013. Full version at `arXiv:1209.2408`.

[GKS17] Rohit Gurjar, Arpita Korwar, and Nitin Saxena. Identity Testing for Constant-Width, and Commutative, Read-Once Oblivious ABPs. *Theory of Computing*, 13(1):1–21, 2017. Preliminary version in the *31st Annual Computational Complexity Conference (CCC 2016)*. `arXiv:1601.08031`.

[GKST17] 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. Preliminary version in the *30th Annual Computational Complexity Conference (CCC 2015)*. arXiv:1411.7341.

[HS80] Joos Heintz and Claus-Peter Schnorr. Testing Polynomials which Are Easy to Compute (Extended Abstract). In *Proceedings of the 12th Annual ACM Symposium on Theory of Computing (STOC 1980)*, pages 262–272, 1980.

[HWY10] Pavel Hrubes, Avi Wigderson, and Amir Yehudayoff. Non-commutative circuits and the sum-of-squares problem. In *Proceedings of the 42nd Annual ACM Symposium on Theory of Computing (STOC 2010)*, pages 667–676, 2010.

[HY16] Pavel Hrubeš and Amir Yehudayoff. On Isoperimetric Profiles and Computational Complexity. In *Proceedings of the 43rd International Colloquium on Automata, Languages and Programming (ICALP 2016)*, pages 89:1–89:12, 2016. eccc:TR15-164.

[Hya79] Laurent Hyafil. On the Parallel Evaluation of Multivariate Polynomials. *SIAM Journal of Computing*, 8(2):120–123, 1979. Preliminary version in the *10th Annual ACM Symposium on Theory of Computing (STOC 1978)*.

[KI04] Valentine Kabanets and Russell Impagliazzo. Derandomizing Polynomial Identity Tests Means Proving Circuit Lower Bounds. *Computational Complexity*, 13(1-2):1–46, 2004. Preliminary version in the *35th Annual ACM Symposium on Theory of Computing (STOC 2003)*.

[KS06] Adam R. Klivans and Amir Shpilka. Learning Restricted Models of Arithmetic Circuits. *Theory of Computing*, 2(10):185–206, 2006. Preliminary version in the *16th Annual Conference on Computational Learning Theory (COLT 2003)*.

[LLS19] Guillaume Lagarde, Nutan Limaye, and Srikanth Srinivasan. Lower Bounds and PIT for Non-commutative Arithmetic Circuits with Restricted Parse Trees. *Computational Complexity*, 28(3):471–542, 2019. Preliminary version in the *42nd Internationl Symposium on the Mathematical Foundations of Computer Science (MFCS 2017)*. eccc:TR17-077.

[LMP19] Guillaume Lagarde, Guillaume Malod, and Sylvain Perifel. Non-commutative computations: lower bounds and polynomial identity testing. *Chicago Journal of Theoretical Computer Science*, 2019. eccc:TR16-094.

[LMS16] Nutan Limaye, Guillaume Malod, and Srikanth Srinivasan. Lower Bounds for Non-Commutative Skew Circuits. *Theory of Computing*, 12(1):1–38, 2016. eccc:TR15-22.

[Nis91] Noam Nisan. Lower bounds for non-commutative computation. In *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing (STOC 1991)*, pages 410–418, 1991. Available on `citeseer:10.1.1.17.5067`.

[Nis92] Noam Nisan. Pseudorandom generators for space-bounded computation. *Combinatorica*, 12(4):449–461, 1992.

[Ore22] Øystein Ore. Über höhere Kongruenzen. *Norsk Mat. Forenings Skrifter*, 1(7):15, 1922.

[RS05] Ran Raz and Amir Shpilka. Deterministic polynomial identity testing in non-commutative models. *Computational Complexity*, 14(1):1–19, 2005. Preliminary version in the *19th Annual IEEE Conference on Computational Complexity (CCC 2004)*.

[Sch80] Jacob T. Schwartz. Fast Probabilistic Algorithms for Verification of Polynomial Identities. *Journal of the ACM*, 27(4):701–717, 1980.

[SV15] Amir Shpilka and Ilya Volkovich. Read-once polynomial identity testing. *Computational Complexity*, 24(3):477–532, 2015. Preliminary version in the *40th Annual ACM Symposium on Theory of Computing (STOC 2008)*.

[Val79] Leslie G. Valiant. Completeness Classes in Algebra. In *Proceedings of the 11th Annual ACM Symposium on Theory of Computing (STOC 1979)*, pages 249–261, 1979.

[VSBR83] Leslie G. Valiant, Sven Skyum, S. Berkowitz, and Charles Rackoff. Fast Parallel Computation of Polynomials Using Few Processors. *SIAM Journal of Computing*, 12(4):641–644, 1983. Preliminary version in the *6th Internationl Symposium on the Mathematical Foundations of Computer Science (MFCS 1981)*.

[Zip79] Richard Zippel. Probabilistic algorithms for sparse polynomials. In *Symbolic and Algebraic Computation, EUROSAM '79, An International Symposiumon Symbolic and Algebraic Computation*, volume 72 of *Lecture Notes in Computer Science*, pages 216–226. Springer, 1979.