LOWER BOUNDS FOR THE SUM OF SMALL-SIZE ALGEBRAIC BRANCHING PROGRAMS

C.S. BHARGAV, PRATEEK DWIVEDI, AND NITIN SAXENA

Abstract. We observe that proving lower bounds for the sum of set-multilinear Algebraic Branching Programs (smABPs) in the *low-degree* regime implies Valiant's conjecture (i.e. it implies general ABP lower bounds). Using this connection, we obtain lower bounds for the sum of small-sized general ABPs. In particular, we show that the sum of $\text{poly}(n)$ ABPs, each of size ($:=$ number of vertices) $n^{o(1)}$, cannot compute the family of Iterated Matrix Multiplication polynomials $\text{IMM}_{n,d}$ for any arbitrary function $d = d(n)$.

We also give a dual version of our result for the sum of *low-variate* ROABPs (read-once oblivious ABPs). Both smABP and ROABP are very well-studied 'simple' models; our work puts them at the forefront of understanding Valiant's conjecture.

1. Introduction

In a pioneering work, Leslie Valiant proposed [32] an *algebraic* framework to study efficient ways of computing multivariate polynomials. The computational model was that of *algebraic circuits* – layered directed acyclic graphs with vertices in intermediate layers alternately labeled by addition ($+$) or multiplication ($\times$), and leaves at the bottom layer labeled with variables $x_1, \ldots, x_n$ or constants of the underlying field $\mathbb{F}$. The circuit inductively computes a multivariate polynomial $f \in \mathbb{F}[x_1, \ldots, x_n]$. Each vertex (gate) performs its corresponding operation ($+$ or $\times$) on the inputs it receives until finally, a designated output vertex computes the polynomial. A measure of efficiency is the *size* of the circuit, that is, the number of vertices in the graph. The *depth* of the circuit is the length of the longest path from the input leaves to the output vertex and measures the amount of *parallelism* in the circuit. For a general survey of algebraic complexity, see [6, 29, 20].

Valiant hypothesized that there are *explicit* polynomials that do not have small algebraic circuits computing them, which we now call the $\mathsf{VP} \neq \mathsf{VNP}$ hypothesis. As algebraic circuits are *non-uniform* models of computation, computing a polynomial more precisely refers to computing a *family* $\{f_n\}_{n \geq 0}$ of polynomials, one for each n. The class VP consists of families of polynomials whose degree and circuit size are both polynomially bounded in the number of variables n (denoted $\text{poly}(n)$ from now on). On the other hand, if a polynomial has degree $\text{poly}(n)$ and the coefficient of any given monomial

can be computed in $\#\mathsf{P}/\operatorname{poly}$, then the polynomial is in VNP [1]. It is not difficult to see that $\mathsf{VP} \subseteq \mathsf{VNP}$.

Much like Cook's original P vs. NP hypothesis [9] in the boolean world, very little is known in general about Valiant's hypothesis. A result of Strassen [30] and Baur-Strassen [4] gives a lower bound of $\Omega(n \log n)$ against general circuits. A slightly better lower bound of $\Omega(n^2)$ is known if the directed acyclic graph underlying the circuit is a *tree* – also known as an *Algebraic Formula*. All polynomials that have formulas of size $\operatorname{poly}(n)$ form the class VF. We refer the interested reader to the excellent book of Bürgisser [5] for more details on Valiant's hypothesis and connections to the Boolean world.

Intermediate in power, and in between circuits and formulas lie Algebraic Branching Programs (ABPs). An ABP is a layered directed acyclic graph with edges labeled by *affine linear forms*. There is a *source* vertex (s) of in-degree 0 in the first layer and a *sink* vertex (t) of out-degree 0 in the last layer, and edges connect vertices in adjacent layers. The maximum number of vertices in any layer is the *width* of the ABP and the number of layers is its *length*. Each path from s to t computes a polynomial that is the product of the edge labels along the path. The polynomial computed by the ABP is the sum of the polynomials computed by all the $s \rightsquigarrow t$ paths.

An ABP of length ℓ with n_i vertices in the i-th layer can be written as a product of $\ell - 1$ matrices $\prod_{i=1}^{\ell-1} M_i$ in a natural way: the matrix M_i is of dimension $n_i \times n_{i+1}$ and contains the edge labels between layers i and $i+1$ as entries. The size of the ABP is the total number of vertices in the graph (or equivalently, the sum of the number of rows of the matrices in matrix representation). Similar to circuits and formulas, the class of polynomials that have ABPs of size $\operatorname{poly}(n)$ is denoted VBP.

It is known that $\mathsf{VF} \subseteq \mathsf{VBP} \subseteq \mathsf{VP}$, and conjectured that all the inclusions are strict. Valiant's hypothesis is considered more generally as the problem of separating any of the classes $\mathsf{VF}, \mathsf{VBP}$ or VP from VNP. Unfortunately (although probably not surprisingly), general lower bounds in any of these models is hard to come by. In a recent work, Chatterjee, Kumar, She and Volk [7] proved a lower bound of $\Omega(n^2)$ for ABPs. Evidently, the state of affairs is quite similar to that of circuits. In fact, the polynomial $\sum_{i=1}^{n} x_i^n$ used in the lower bound is the same one that Baur and Strassen [4] used for their circuit lower bound.

In this work, we will mainly be interested in *set-multilinear* polynomials, of which the Iterated Matrix Multiplication polynomial is an excellent example. The polynomial $\mathrm{IMM}_{n,d}$ is defined on $N = dn^2$ variables. The variable set X is partitioned into d sets $(X_1, \ldots, X_d)$ of n^2 variables each (viewed as $n \times n$ matrices). The polynomial is defined as the $(1,1)$-th entry of the matrix product $X_1 \cdot X_2 \cdots X_d$:

[1] This is simply a sufficient condition for a polynomial to be in VNP, but is enough for our purpose. A precise definition can be found in [29, Definition 1.3]

$$\mathrm{IMM}_{n,d} = \left(\begin{bmatrix} x_{1,1} & \dots & x_{1,n} \\ \vdots & \ddots & \vdots \\ x_{1,n^2-n+1} & \dots & x_{1,n^2} \end{bmatrix} \cdots\cdots \begin{bmatrix} x_{d,1} & \dots & x_{d,n} \\ \vdots & \ddots & \vdots \\ x_{d,n^2-n+1} & \dots & x_{d,n^2} \end{bmatrix} \right)_{(1,1)}.$$

As all monomials are of the same degree d, the polynomial is *homogeneous*. It is also *multilinear* since every variable has individual degree at most 1. Additionally, every monomial has exactly one variable from each of the d sets of the partition. Thus it is *set-multilinear*. Henceforth, by a set-multilinear polynomial $P_{n,d}$ over the variable set $X = X_1 \sqcup \ldots \sqcup X_d$ (with $|X_i| \leq n$ for all $i \in [d]$), we mean a homogeneous multilinear polynomial with the following property: every monomial m (seen as a set) in $P_{n,d}$ satisfies $|m \cap X_i| = 1$ for all $i \in [d]$.

1.1. **Our Results.** Our first result is a lower bound against the sum of general *small-size* algebraic branching programs.

Theorem 1.1 ($\sum$smABP lower bound)**.** *The polynomial* $\mathrm{IMM}_{n,d}$ *cannot be computed by the sum of* poly(n) ABP*s, each of size* $n^{o(1)}$.

For an arbitrary function $d = d(n)$, the family of polynomials $\{\mathrm{IMM}_{n,d}\}_n$ has an ABP of size $O(nd)$. The above theorem shows that reducing the ABP size *slightly*, suddenly requires a superpolynomial sum of ABPs to compute the polynomial.

Remark. When $d > n^{o(1)}$, ABPs of size $n^{o(1)}$ cannot even produce monomials of degree d. Hence, the theorem statement is obtained trivially (in general, a lower bound of d is trivial for ABPs). But when $d < n^{o(1)}$, the model is quite powerful. In fact, for $d < n^{o(1)}$, the power sum polynomial $\sum_{i=1}^{n} x_i^d$, that was used in previous ABP lower bounds, can be computed efficiently using a sum of n ABPs, each of size $n^{o(1)}$.

Note that a lower bound of n is not trivial for ABPs (unlike circuits and formulas). Moreover, each edge label can be a general affine linear form, allowing a single path to generate exponentially many monomials. Notwithstanding that, ABPs of size $n^{o(1)}$ are still an incomplete model of computation. Nevertheless, the sum of such ABPs is a complete model – every polynomial of degree less than $n^{o(1)}$ can be written as a sum of width-1 ABPs (monomials).

Note. The lower bound of Theorem 1.1 also holds if we replace IMM with an appropriate polynomial from the family of Nisan-Wigderson design-based polynomials (see Section 4).

Our next result is a reformulation of Valiant's conjecture in terms of a different model: the sum of *set-multilinear* ABPs (smABPs) on the set of variables $X = X_1 \sqcup \ldots \sqcup X_d$. An smABP in the *natural order* is a $(d+1)$ layered ABP with edges between layers i and $i+1$ labeled by *linear forms*

in X_i. More generally, for a permutation $\pi \in S_d$ of the variable sets, we say that an smABP is in the order π if the edges between i-th and $(i+1)$-th layer are labeled by *linear forms* in $X_{\pi(i)}$ [2].

We denote by $\sum$smABP the sum of set-multilinear ABPs, each in a possibly different order. The *width* of a $\sum$smABP is the sum of the widths of the constituent smABPs.

We show that in the *low-degree* regime, superpolynomial lower bounds against $\sum$smABP imply superpolynomial ABP lower bounds.

Theorem 1.2 (Hardness bootstrapping)**.** *Let n, d be integers such that $d = O(\log n / \log\log n)$. Let $P_{n,d}$ be a set-multilinear polynomial in* VNP *of degree d. If $P_{n,d}$ cannot be computed by a $\sum$*smABP *of width* poly(n)*, then* VBP $\neq$ VNP*.*

The above theorem shows that the sum of set-multilinear ABPs which looks quite restrictive, is surprisingly powerful. This is a recurring theme in algebraic complexity. A series of works [33, 2, 16, 31, 13] on reducing the *depth* of algebraic circuits culminated in the rather surprising fact that good enough lower bounds for depth-3 circuits imply general circuit lower bounds. The above theorem is in a similar vein. The model of $\sum$smABP is particularly appealing to study since smABPs are one of the most well understood objects in algebraic complexity.

Non-commuting matrices make it powerful. Note that if the matrices in the smABP were commutative, we can treat $\sum$smABP as a *single* smABP, against which we know how to prove lower bounds (see Section 1.2). So in order to lift the lower bound to VNP, it is essential that we understand the sum of smABPs with non-commuting matrices (see Section 1.3 for a detailed discussion).

Arbitrarily low degree suffices. The low-degree regime has recently gained a lot of attention. In a breakthrough work, Limaye Srinivasan and Tavenas [19] showed how to prove superpolynomial lower bounds for constant-depth set-multilinear formulas when the degree is small (set-multilinear lower bounds against arbitrary depth were known before [22, 23, 25], but degenerated to trivial bounds when the degree was small). They were able to then escalate the low-degree, set-multilinear lower bounds to *general* constant-depth circuit lower bounds. The theorem above shows that the low-degree regime can be helpful in proving lower bounds for ABPs as well.

1.2. **The sum of ROABPs perspective: the arbitrarily low variate case.** One can also view Theorem 1.2 through the lens of another well-studied model in the literature, first defined by Forbes and Shpilka [10].

[2]This definition differs slightly from that of Forbes [11] as it does not allow *affine* linear forms as edge labels. We use this definition as the ABPs we encounter are of this more restricted form and proving lower bounds for them is sufficient

An algebraic branching program over the variables $(x_1, \ldots, x_n)$ is said to be read-once oblivious (or an ROABP) in the *natural order* if the edges between layers i and $i+1$ are labeled by univariate polynomials in x_i of degree d. If instead the labels were univariate polynomials in $x_{\pi(i)}$ for some permutation $\pi \in S_d$ of the variables, then we say that the ROABP is in the order π.

The computation that an ROABP (or equivalently, an smABP) performs is essentially non-commutative since the variables along a path get multiplied in the same order π as that of the ROABP (smABP). Nisan [21] introduced the powerful technique of using spaces of partial derivatives to study lower bound questions in non-commutative models. This technique can be used to calculate the exact width of the ROABP computing a polynomial.

Following our definition for smABPs, we denote by $\sum$RO the sum of ROABPs, each possibly in a different order. The width of a $\sum$RO is the sum of the widths of the constituent ROABPs. A version of Theorem 1.2 can also be stated for this model. In contrast to the case of smABPs, we will be interested in the dual *low-variate* regime.

Corollary 1.3 (Low variate $\sum$RO)**.** *Let n, d be integers such that $n = O(\log d / \log\log d)$. Let $f \in \mathsf{VNP}$ be a polynomial on n variables of individual degree d. If f cannot be computed by a $\sum$RO of width* $\mathrm{poly}(d)$*, then* $\mathsf{VBP} \neq \mathsf{VNP}$.

Proof. Consider the *invertible* map $\phi : x_i^j \mapsto x_{ij}$ for the indices $i \in [n]$ and $j \in [d]$. This transforms an ROABP on n variables $(x_1, \ldots, x_n)$ of individual degree d and order π, to an smABP in the same order that is set-multilinear with respect to $X = X_1 \sqcup \ldots \sqcup X_n$ with $|X_i| \leq d$.

We apply the map ϕ to the $\sum$RO computing f. This gives us a $\sum$smABP of the same width that computes a set-multilinear polynomial $Q_{d,n}$ over $O(nd)$ variables with $n = O(\log d / \log\log d)$. Since f does not have a $\sum$RO of width $\mathrm{poly}(d)$, $Q_{d,n}$ does not have $\sum$smABP of width $\mathrm{poly}(d)$. Now Theorem 1.2 gives us our desired separation. □

The low-variate regime has also recently been shown to be extremely important. The Polynomial Identity Testing (PIT) problem asks to efficiently test whether a polynomial (given as an algebraic circuit, for example) is identically *zero*. In the black-box setting, we are only allowed to evaluate the polynomial (circuit) at various points. Hence, PIT algorithms are equivalent to the construction of *hitting sets* – a collection of points that witness the (non)zeroness of the polynomial computed by the circuit (see [27, 28] for a survey of PIT and techniques used).

Recently, several surprising results [1, 18, 12] essentially conclude that hitting sets for circuits computing extremely low-variate polynomials can be "bootstrapped" to obtain hitting sets for general circuits. See the survey of Kumar and Saptharishi [17] for an exposition of the ideas involved.

1.3. **Proof techniques and previous work.**

Simulating ABPs using sum of smABPs. Unlike the boolean world, *both* the degree d of the polynomial, and the number of variables n are important parameters in algebraic complexity. Often times, it is reasonable and useful to impose restrictions on one of them. Even in the definitions VP and VNP, we require that the degree d be restricted by a polynomial in n (see [14] for more discussion on the motivation behind this choice). Further restrictions on the degree help in proving better structural results which would otherwise be prohibitively costly to perform.

In order to prove Theorem 1.2, we perform a sequence of structural transformations to the algebraic branching program to obtain a $\sum$smABP. We first *homogenize* the ABP (Lemma 2.2), i.e., we alter the ABP so that every vertex in the ABP computes a homogeneous polynomial. In addition, we will ensure that the ABP has d layers and all the edge labels are *linear forms*. The homogenization of ABPs to this form was folklore. Subsequently, we set-multilinearize the branching program (Lemma 2.1). This step is only efficient in the low-degree regime since what we obtain is a sum of $d^{O(d)}$ set-multilinear ABPs.

With the reduction in place, superpolynomial lower bounds for $\sum$smABP imply the same for ABPs, albeit in the low-degree regime.

Lower bounds for the sum of ABP*s.* Using Nisan's characterization [21] mentioned before, we can prove exponential lower bounds against smABPs (ROABPs), but their sums have resisted attempts at strong lower bounds since the characterization does not extend to the sum. The best known lower bound is due to Arvind and Raja [3] who proved that, in any sum of k ROABPs computing the Permanent polynomial $\mathrm{Per}_n = \sum_{\pi \in S_n} \prod_{i=1}^{n} x_{i,\pi(i)}$, at least one of them must have size $2^{\Omega(n/k)}$. Notice that if we want to prove a superpolynomial lower bound of $n^{\Omega(\log n)}$ (say), then the number of ROABPs in the sum can only be about $O(n/\log^2 n)$.

Our proof of the $\sum$ABP lower bound (Theorem 1.1) uses the implicit reduction of Theorem 1.2 to $\sum$smABP. To prove lower bounds for the latter model, we use the *partial derivative method*, introduced in the highly influential work of Nisan and Wigderson [22]. We show that the partial derivative measure $\mu(\cdot)$ is large for our hard polynomial but small for the model. In fact, a majority of the lower bounds in algebraic complexity (including the Arvind–Raja bound described above) use modifications and extensions of this measure. For a comprehensive survey of lower bounds and the use of partial derivative measure in algebraic complexity, see [8, 26].

Consider first the problem of proving lower bounds for $\sum$smABP with no restriction. A major impediment to using Per_n as the hard polynomial is its degree n, which is polynomially related to the number of variables n^2, making it infeasible to handle sums that are exponentially large in the degree. Instead, we work with $\mathrm{IMM}_{n,d}$, which gives us more flexibility in terms of

independently choosing n and d. Unfortunately, this choice creates a two-fold problem. The fundamental one is that $\mathrm{IMM}_{n,d}$ has a small smABP. So we can never prove a superpolynomial lower bound for even a single $\mathrm{poly}(n,d)$ sized smABP (let alone their sum).

One might try to avoid this by choosing a different hard polynomial that gives similar flexibility, perhaps something from the family of Nisan-Wigderson design-based polynomials. But in fact, the complexity measure μ is also *maximal* for $\mathrm{IMM}_{n,d}$. Hence, the partial derivative method cannot be used to prove lower bounds against any model that efficiently computes $\mathrm{IMM}_{n,d}$. Be that as it may, it might still be possible to use the same technique to prove lower bounds for restrictions of the model. We are able to do this when the widths of the smABPs are small. It also enables us to handle extremely large sums of smABPs (including those that occur from considering sums of multiple ABPs). The recent low-depth circuit lower bound of LST [19] does something very similar in spirit. Although $\mathrm{IMM}_{n,d}$ can be computed efficiently using depth $O(\log d)$ circuits, they were able to use (a slight but ingenious modification of) the partial derivative method to prove superpolynomial lower bounds for constant-depth circuits.

It is worth recalling that the techniques we just described work only in the low-degree regime, since our reductions are only efficient if the degree is very small. To handle higher degrees, we note that $\mathrm{IMM}_{n,d'}$ with d' small can be obtained as a set-multilinear restriction of $\mathrm{IMM}_{n,d}$. Therefore, our lower bounds translate to higher degrees to finally give superpolynomial lower bounds against sums of small-sized general ABPs.

2. Reducing ABPs to the sum of set-multilinear ABPs

We begin by showing that in the low-degree regime, a small sized ABP can be simulated by a $\sum$smABP of small width. This is very much in the spirit of the set-multilinearization result of Limaye, Srinivasan and Tavenas ([19], Proposition 9) for small-depth circuits.

Lemma 2.1 (ABP set-multilinearization)**.** *Let $P_{n,d}$ be a polynomial of degree d, set-multilinear with respect to the partition $X = X_1 \sqcup \ldots \sqcup X_d$ where $|X_i| \leq n$ for all $i \in [d]$. If $P_{n,d}$ can be computed by an* ABP *of size s, then there is a $\sum$*smABP *of width $d^{O(d)}s$ computing the same polynomial.*

We immediately obtain

Proof of Theorem 1.2. Suppose that the polynomial $P_{n,d} \in \mathsf{VNP}$ can be computed by an ABP of size s. By Lemma 2.1, the polynomial can also be computed by a $\sum$smABP of width $d^{O(d)}s$. The width of any $\sum$smABP computing $P_{n,d}$ is, by assumption $n^{\omega(1)}$.

Consequently, our desired separation is obtained by first noting that $d^{O(d)}s \geq n^{\omega(1)}$, whereby the degree bound $d = O(\log n / \log\log n)$ implies $d^{O(d)} = \mathrm{poly}(n)$ and hence $s \geq n^{\omega(1)}$. □

In order to prove Lemma 2.1, we first homogenize the ABP (similar to the approach of Raz [24] and LST [19]). Any vertex v in an ABP can be thought of as computing a polynomial corresponding to the 'sub-ABP' between the source s and the vertex v. An ABP is homogenous if the polynomial computed at every vertex is homogenous.

Lemma 2.2 (ABP homogenization)**.** *Let $f(x_1, \ldots, x_n)$ be a degree d polynomial. Suppose that f can be computed by an* ABP *of size s. Then there is a* homogeneous ABP *of width s and length d that can compute the same polynomial. Furthermore, all the edge labels are* linear forms.

The above lemma is "folklore" with the proof idea already present in [21]. We provide a proof for completeness (see Appendix A), based on the exposition of [15]. It turns out that this homogeneous ABP can be *efficiently* set-multilinearized.

Proposition 2.3. *Consider a set-multilinear polynomial $P_{n,d}$ over the variable set $X = X_1 \sqcup \ldots \sqcup X_d$ (with $|X_i| \leq n$ for all $i \in [d]$) computed by a* homogeneous ABP *of width w and length d. Then, there is a $\sum$*smABP *of width $d!w$ computing $P_{n,d}$.*

Proof. We begin by writing the homogeneous ABP in its matrix form

$$P_{n,d} = \prod_{i=1}^{d} M_i, \tag{2.1}$$

where each M_i is a $w \times w$ matrix with entries that are *linear forms* in the variables X. We further write each M_i as a sum $\sum_{j=1}^{d} M_{ij}$, where for all j, M_{ij} is an $w \times w$ matrix with entries that are linear forms, but now in the X_j variables. Doing this for every M_i yields

$$P_{n,d} = \prod_{i=1}^{d} \sum_{j=1}^{d} M_{ij}. \tag{2.2}$$

Note that since $P_{n,d}$ is a homogeneous set-multilinear polynomial, the non-set-multilinear products in this expression can be ignored. The matrices only contain linear forms, and thus non-set-multilinear products in the above equation only produce non-set-multilinear monomials. We can ignore any product of the form $(\cdots M_{ij} \cdots M_{i'j} \cdots)$ for different i, i'. We can rearrange to obtain

$$P_{n,d} = \sum_{\pi \in S_d} \prod_{i=1}^{d} M_{i\pi(i)}. \tag{2.3}$$

This represents $P_{n,d}$ as the sum of $d!$ set-multilinear ABPs, each of width w. □

We can now complete the reduction.

Proof of Lemma 2.1. Suppose that the ABP for the polynomial $P_{n,d}$ has size s. Using Lemma 2.2, we can *homogenize* it to obtain a d-layered homogeneous ABP of *width* s. By Proposition 2.3, we obtain a $\sum$smABP of width $d!s = d^{O(d)}s$. □

3. Lower Bound for the sum of ABPs

We now show that the family of Iterated Matrix Multiplication polynomial $\{\mathrm{IMM}_{n,d(n)}\}_n$ cannot be computed even by a polynomially large sum of ABPs, provided that each of the ABPs is small in size. We begin by proving a lower bound for $\sum$smABP in the *low-degree* regime. Note that in this regime, IMM has an smABP of width $O(n^2)$. The lemma shows that even using the sum of multiple smABPs cannot help in reducing the width.

Lemma 3.1. *Any* $\sum$smABP *computing the polynomial* $\mathrm{IMM}_{n,d}$ *with* $d = O(\log n/\log\log n)$*, must have width at least* $n^{\Omega(1)}$.

Proof. Let the maximum width of any smABP in the sum be w. Every path in a particular set-multilinear ABP is of length d and computes a product of linear forms. Using the definition of ABP computation, we sum over all paths to obtain a depth-3 *set-multilinear circuit*[3] of top fanin w^d. Doing the same for all the smABPs, we get a depth-3 set-multilinear circuit of top fan-in at most $d!w^d$.

We now apply the partial derivative method. Split $X = X_1 \sqcup \ldots \sqcup X_d$ into 'even' and 'odd' parts. That is, we consider the partition $X = X^{(0)} \sqcup X^{(1)}$, with

$$X^{(0)} = X_2 \sqcup X_4 \sqcup \ldots \sqcup X_k, \text{ and } X^{(1)} = X_1 \sqcup X_3 \sqcup \ldots \sqcup X_{k'}, \tag{3.1}$$

where $k = 2\lfloor d/2 \rfloor$ and $k' = 2\lceil d/2 \rceil - 1$.

The partial derivative matrix $\mathcal{M}(f)$ for any polynomial f has rows indexed by set-multilinear monomials in $X^{(0)}$ and columns indexed by set-multilinear monomials in $X^{(1)}$. Consider now monomials m_0, m_1 that are set-multilinear in $X^{(0)}, X^{(1)}$ respectively. For any set-multilinear polynomial f, the (m_0, m_1) entry in $\mathcal{M}(f)$ is the coefficient of the monomial $m_0 \cdot m_1$ in f. It is straightforward to see that the partial derivative matrix of $\mathrm{IMM}_{n,d}$ is of full rank, that is, $\mathrm{rank}(\mathcal{M}(\mathrm{IMM}_{n,d})) = n^{d/2}$.

On the other hand, when we consider a set-multilinear $\sum\prod\sum$ circuit, the linear forms at the bottom have a rank of at most 1 with respect to any partition of X. Consequently, taking products of linear forms cannot result in a polynomial of rank greater than 1. Finally, *subadditivity* of matrix rank implies that the rank of the set-multilinear circuit is at most the top-fanin $d!w^d$, giving

$$n^{d/2} \leq d!w^d. \tag{3.2}$$

[3]Every vertex in a set-multilinear circuit computes a set-multilinear polynomial with respect to a subset of the variable sets.

Using the fact that $d! = O(d^d) = \text{poly}(n)$ for our degree regime, it now follows that $w = n^{\Omega(1)}$ and we obtain the $\sum$smABP lower bound. □

The resistance of IMM to width reduction even by a $\sum$smABP helps us in proving our main lower bound.

Proof of Theorem 1.1. Suppose that $\text{IMM}_{n,d}$ (with $d \leq n^{o(1)}$) can be written as the sum of m ABPs of size $s = n^{o(1)}$ each[4]. In the corresponding matrix form, we have

$$\text{IMM}_{n,d} = \sum_{i=1}^{m} \prod_{j=1}^{\ell} M_{ij}, \tag{3.3}$$

where each M_{ij} is an $s \times s$ matrix and $\ell \leq s$.

Consider now the polynomial $\text{IMM}_{n,d'}$ with $d' = O(\log n / \log\log n)$. This polynomial can be obtained as a restriction of $\text{IMM}_{n,d}$ by setting all matrices other than the first d' in the definition of IMM to the identity matrix I_n. Correspondingly, Equation 3.3 now becomes

$$\text{IMM}_{n,d'} = \sum_{i=1}^{m} \prod_{j=1}^{\ell} M'_{ij}, \tag{3.4}$$

where just like in (3.3), each M'_{ij} is an $s \times s$ matrix and $\ell \leq s$. Note that any lower bound on $\text{IMM}_{n,d'}$ also holds for $\text{IMM}_{n,d}$.

We would like to set-multilinearize Equation 3.4. But we cannot directly apply Lemma 2.1 since the ABPs in the sum need not compute a set-multilinear polynomial anymore. In fact, they need not even compute a homogeneous polynomial. Nevertheless, we are only interested in the homogeneous component of degree d' of the polynomials that these ABPs compute, the rest vanishing in the final sum.

Consider a single ABP A of size $s = n^{o(1)}$ from the sum of m ABPs above. Suppose that it computes a (possibly non-homogenous) polynomial of degree d_A. Using Lemma 2.2, we can homogenize A to obtain an ABP of length d_A and width s, with linear forms on the edges. Consider now the (possibly empty) set T of vertices in layer d' of this ABP that have no outgoing edges. For every $v \in T$, the sub-ABP between the start vertex s and the vertex v computes a homogeneous polynomial of degree d', monomials of which might occur in the final polynomial $\text{IMM}_{n,d'}$. Vertices not in T can be safely ignored as they have outgoing edges with linear *forms* on them and hence will only contribute to monomials of degree greater than d' in the polynomial computed by A.

We now identify all the vertices in T with a single vertex t. Furthermore, we replace all the possible multi-edges generated between a vertex u in layer $d' - 1$ and the vertex t, with a single edge that has as its edge label the sum of all the multi-edge labels. This gives us a homogeneous ABP of width

[4]When $d > n^{o(1)}$, the lower bound trivially holds.

s and length d' computing the homogeneous component of degree d' of the polynomial computed by A. Performing this operation for each of the m ABPs, we can write

$$\mathrm{IMM}_{n,d'} = \sum_{i=1}^{m} \prod_{j=1}^{d'} M'_{ij}, \tag{3.5}$$

where the new matrices obtained after homogenization have been renamed to M' for brevity. As before, we split each M'_{ij} as a sum $\sum_{k=1}^{d'} M'_{ijk}$ where for all $k \in [d']$, M'_{ijk} is an $s \times s$ matrix with entries that are *linear forms* in the X_k variables.

$$\mathrm{IMM}_{n,d'} = \sum_{i=1}^{m} \prod_{j=1}^{d'} \sum_{k=1}^{d'} M'_{ijk}, \tag{3.6}$$

In the proof of Proposition 2.3, we were crucially using the fact that the polynomial computed by the ABP was set-multilinear in order to ignore non-set-multilinear products. Although this is not the case any longer, we can still ignore all the non-set-multilinear products since they *only* produce non-set-multilinear monomials and the sum of the ABPs is $\mathrm{IMM}_{n,d'}$, a set-multilinear polynomial. We obtain an expression similar to Equation 2.3:

$$\mathrm{IMM}_{n,d'} = \sum_{i=1}^{m} \sum_{\pi \in S_{d'}} \prod_{j=1}^{d'} M'_{ij\pi(j)}. \tag{3.7}$$

That is, $\mathrm{IMM}_{n,d'}$ can be written as the sum of $md'!$ smABPs, each of width s. We now analyze similarly to the proof of Lemma 3.1. We convert the $\sum$smABP to a depth 3 set-multilinear circuit of top-fanin at most $md'!s^{d'}$. Using the exact same partition of X into $X^{(0)}$ and $X^{(1)}$ as in (3.1), we construct the partial derivative matrix $\mathcal{M}$ for $\mathrm{IMM}_{n,d'}$ and the set-multilinear $\sum\prod\sum$ circuit that we obtained. The rank calculation results in

$$n^{d'/2} \leq md'!s^{d'}, \tag{3.8}$$

which along with $s = n^{o(1)}$ and $d'! = \mathrm{poly}(n)$ gives $m = n^{\omega(1)}$. □

4. Discussion and Open problems

In order to separate VBP from VNP, we will need to prove super-polynomial lower bounds against $\sum$smABP for a polynomial in VNP that we expect to be hard. As noted above, the IMM polynomial is in VBP (in fact, it is a canonical way to define the class VBP) and hence cannot be used for such a separation. We briefly sketch here why Theorem 1.1 also holds for a polynomial from the Nisan-Wigderson family of design-based polynomials that is in VNP and has been used in many other lower bound results. It is conjectured not to be in VBP.

Let $\mathbb{F}_n$ be a field of size n (we assume that n is a power of a prime). We will work in the low-degree regime. For $d = O(\log n / \log\log n)$, consider the

set of variables $X = X_1 \sqcup \ldots \sqcup X_d$ where $X_i = \{x_{ij} \mid j \in [n]\}$ for all $i \in [d]$. Let $\mathcal{F}$ be the set of all *univariate* polynomials $f(y) \in \mathbb{F}_n[y]$ of degree less than $d/2$. The polynomial $\mathrm{NW}_{n,d}$ on the above nd variables is defined as

$$\mathrm{NW}_{n,d}(X) = \sum_{f \in \mathcal{F}} \prod_{i \in [d]} x_{if(i)}.$$

Each monomial encodes a univariate polynomial of degree less than $d/2$. Consider the partition $X = X^{(0)} \sqcup X^{(1)}$ from (3.1). For a monomial $m_0 = x_{2j_2} \cdots x_{kj_k}$ (with all j indices in $[n]$) that is set-multilinear in $X^{(0)}$, there is a unique "extension monomial" m_1 (set-multilinear in $X^{(1)}$) such that $m_0 m_1$ is a monomial of $\mathrm{NW}_{n,d}$. This is because m_0 encodes the evaluations of some univariate polynomial on points $\{2, \ldots, k\}$. As the length of m_0 is at least $d/2$, interpolating these values gives a unique polynomial f which then determines the corresponding m_1 – obtained by evaluating f on the remaining points $\{1, 3, \ldots, k'\}$ in $[d]$.

This implies that the partial derivative matrix $\mathcal{M}(\mathrm{NW}_{n,d})$ of size $n^{d/2} \times n^{d/2}$ has full rank. The same rank analysis as before on sums of ABPs gives us Theorem 1.1, but with $\mathrm{NW}_{n,d}$ as the hard polynomial. Nevertheless, the techniques used seem to not be enough to get us any better lower bounds. In particular, the loss of information in the conversion of an smABP (an essentially non-commutative model) to a set-multilinear circuit seems to be too large [5].

A first step toward proving ABP lower bounds would be to prove lower bounds against the sum of $O(n)$ smABPs in the low degree regime. But the question is open even in the high-degree regime. Another interesting direction is to show a reduction from ABPs to the sum of fewer than $d!$ smABPs, with a possibly super polynomial blow up in the smABP size. This would still lead to ABP lower bounds if we can prove strongly exponential lower bounds against the sum of (fewer) smABPs. This question remains open as well.

References

[1] Manindra Agrawal, Sumanta Ghosh, and Nitin Saxena. Bootstrapping variables in algebraic circuits. *Proc. Natl. Acad. Sci. USA*, 116(17):8107–8118, 2019. doi:10.1073/pnas.1901272116.

[2] Manindra Agrawal and V. Vinay. Arithmetic circuits: A chasm at depth four. In *2008 49th Annual IEEE Symposium on Foundations of Computer Science*, pages 67–75, 2008. doi:10.1109/FOCS.2008.32.

[3] V. Arvind and S. Raja. Some lower bound results for set-multilinear arithmetic computations. *Chic. J. Theoret. Comput. Sci.*, pages Art. 6, 26, 2016. doi:10.4086/cjtcs.2016.006.

[4] Walter Baur and Volker Strassen. The complexity of partial derivatives. *Theoret. Comput. Sci.*, 22(3):317–330, 1983. doi:10.1016/0304-3975(83)90110-X.

[5] As an informal comment, we note that conversion to much higher depth circuits seems to not help as well, even if we assume optimal set-multilinear lower bounds.

[5] Peter Bürgisser. *Completeness and reduction in algebraic complexity theory*, volume 7 of *Algorithms and Computation in Mathematics*. Springer-Verlag, Berlin, 2000. doi:10.1007/978-3-662-04179-6.

[6] Peter Bürgisser, Michael Clausen, and M. Amin Shokrollahi. *Algebraic complexity theory*, volume 315 of *Grundlehren der mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences]*. Springer-Verlag, Berlin, 1997. With the collaboration of Thomas Lickteig. doi:10.1007/978-3-662-03338-8.

[7] Prerona Chatterjee, Mrinal Kumar, Adrian She, and Ben Lee Volk. Quadratic lower bounds for algebraic branching programs and formulas. *Comput. Complexity*, 31(2):Paper No. 8, 54, 2022. doi:10.1007/s00037-022-00223-8.

[8] Xi Chen, Neeraj Kayal, and Avi Wigderson. Partial derivatives in arithmetic complexity and beyond. *Found. Trends Theor. Comput. Sci.*, 6(1-2):front matter, 1–138, 2010. doi:10.1561/0400000043.

[9] Stephen A. Cook. The complexity of theorem-proving procedures. In *Proceedings of the Third Annual ACM Symposium on Theory of Computing*, STOC '71, page 151–158, New York, NY, USA, 1971. Association for Computing Machinery. doi:10.1145/800157.805047.

[10] Michael A. Forbes and Amir Shpilka. Quasipolynomial-time identity testing of noncommutative and read-once oblivious algebraic branching programs. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science—FOCS 2013*, pages 243–252. IEEE Computer Soc., Los Alamitos, CA, 2013. doi:10.1109/FOCS.2013.34.

[11] Michael Andrew Forbes. *Polynomial Identity Testing of Read-Once Oblivious Algebraic Branching Programs*. ProQuest LLC, Ann Arbor, MI, 2014. Thesis (Ph.D.)–Massachusetts Institute of Technology.

[12] Zeyu Guo, Mrinal Kumar, Ramprasad Saptharishi, and Noam Solomon. Derandomization from algebraic hardness. *SIAM J. Comput.*, 51(2):315–335, 2022. doi:10.1137/20M1347395.

[13] Ankit Gupta, Pritish Kamath, Neeraj Kayal, and Ramprasad Saptharishi. Arithmetic circuits: a chasm at depth 3. *SIAM J. Comput.*, 45(3):1064–1079, 2016. doi:10.1137/140957123.

[14] Joshua Grochow (https://cstheory.stackexchange.com/users/129/joshua grochow). Degree restriction for polynomials in VP. Theoretical Computer Science Stack Exchange. URL:https://cstheory.stackexchange.com/q/19268 (version: 2013-10-03). URL: https://cstheory.stackexchange.com/q/19268, arXiv:https://cstheory.stackexchange.com/q/19268.

[15] Christian Ikenmeyer and J. M. Landsberg. On the complexity of the permanent in various computational models. *Journal of Pure and Applied Algebra*, 221(12):2911–2927, 2017. doi:10.1016/j.jpaa.2017.02.008.

[16] Pascal Koiran. Arithmetic circuits: the chasm at depth four gets wider. *Theoret. Comput. Sci.*, 448:56–65, 2012. doi:10.1016/j.tcs.2012.03.041.

[17] Mrinal Kumar and Ramprasad Saptharishi. Hardness-randomness tradeoffs for algebraic computation. *Bull. Eur. Assoc. Theor. Comput. Sci. EATCS*, 3(129):56–87, 2019.

[18] Mrinal Kumar, Ramprasad Saptharishi, and Anamay Tengse. Near-optimal bootstrapping of hitting sets for algebraic circuits. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 639–646. SIAM, Philadelphia, PA, 2019. doi:10.1137/1.9781611975482.40.

[19] Nutan Limaye, Srikanth Srinivasan, and Sébastien Tavenas. Superpolynomial lower bounds against low-depth algebraic circuits. In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science—FOCS 2021*, pages 804–814. IEEE Computer Soc., Los Alamitos, CA, [2022] ©2022. doi:10.1109/FOCS52979.2021.00083.

[20] Meena Mahajan. Algebraic complexity classes. In *Perspectives in computational complexity*, volume 26 of *Progr. Comput. Sci. Appl. Logic*, pages 51–75. Birkhäuser/Springer, Cham, 2014.

[21] Noam Nisan. Lower bounds for non-commutative computation. In *Proceedings of the twenty-third annual ACM symposium on Theory of Computing*, STOC '91, pages 410–418, New York, NY, USA, 1 1991. Association for Computing Machinery. doi:10.1145/103418.103462.

[22] Noam Nisan and Avi Wigderson. Lower bounds on arithmetic circuits via partial derivatives. *Computational Complexity*, 6(3):217–234, 1996. doi:10.1007/BF01294256.

[23] Ran Raz. Multi-linear formulas for permanent and determinant are of super-polynomial size. *J. ACM*, 56(2):Art. 8, 17, 2009. doi:10.1145/1502793.1502797.

[24] Ran Raz. Tensor-rank and lower bounds for arithmetic formulas. *Journal of the ACM*, 60(6):Art. 40, 15, 2013. doi:10.1145/2535928.

[25] Ran Raz and Amir Yehudayoff. Lower bounds and separations for constant depth multilinear circuits. *Comput. Complexity*, 18(2):171–207, 2009. doi:10.1007/s00037-009-0270-8.

[26] Ramprasad Saptharishi. A survey of lower bounds in arithmetic circuit complexity. Github Survey, 2015. URL: https://github.com/dasarpmar/lowerbounds-survey.

[27] Nitin Saxena. Progress on polynomial identity testing. *Bull. Eur. Assoc. Theor. Comput. Sci. EATCS*, (99):49–79, 2009.

[28] Nitin Saxena. Progress on polynomial identity testing-II. In *Perspectives in computational complexity*, volume 26 of *Progr. Comput. Sci. Appl. Logic*, pages 131–146. Birkhäuser/Springer, Cham, 2014.

[29] Amir Shpilka and Amir Yehudayoff. Arithmetic circuits: a survey of recent results and open questions. *Found. Trends Theor. Comput. Sci.*, 5(3-4):207–388, 2009. doi:10.1561/0400000039.

[30] Volker Strassen. Die Berechnungskomplexität von elementarsymmetrischen Funktionen und von Interpolationskoeffizienten. *Numer. Math.*, 20:238–251, 1972/73. doi:10.1007/BF01436566.

[31] Sébastien Tavenas. Improved bounds for reduction to depth 4 and depth 3. *Inform. and Comput.*, 240:2–11, 2015. doi:10.1016/j.ic.2014.09.004.

[32] L. G. Valiant. Completeness classes in algebra. In *Conference Record of the Eleventh Annual ACM Symposium on Theory of Computing (Atlanta, Ga., 1979)*, pages pp 249–261. ACM, New York, 1979.

[33] L. G. Valiant, S. Skyum, S. Berkowitz, and C. Rackoff. Fast parallel computation of polynomials using few processors. *SIAM J. Comput.*, 12(4):641–644, 1983. doi:10.1137/0212043.

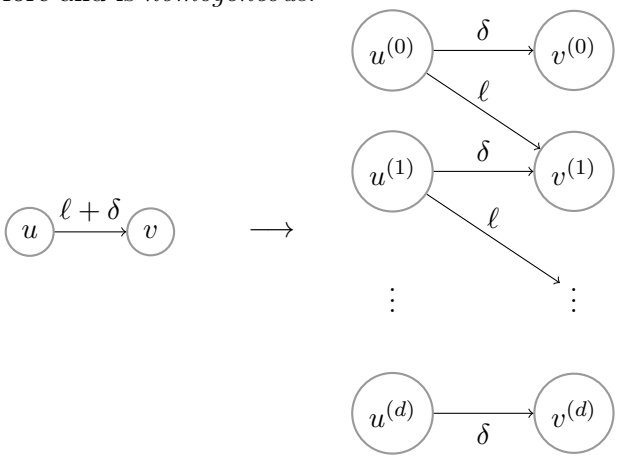
Appendix A. ABP homogenization

To perform the homogenization of ABPs, we follow the exposition in [15].

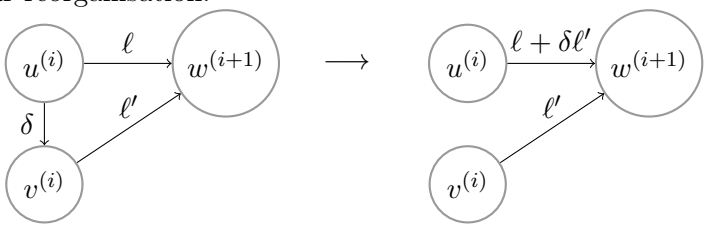
Lemma A.1 (ABP homogenization)**.** *Let $f(x_1, \ldots, x_n)$ be a degree d polynomial. Suppose that f can be computed by an* ABP *of size s. Then there is a* homogeneous ABP *of width s and length d that can compute the same polynomial. Furthermore, all the edge labels are* linear forms.

Proof. We first homogenize the ABP in a manner similar to the case of circuits. For every vertex v (other than the start vertex), we replace it with $d+1$ vertices $v^{(0)}, v^{(1)}, \ldots, v^{(d)}$. Each $v^{(i)}$ corresponds to the homogeneous degree i component of the polynomial computed at v. In the original ABP,

say an edge from vertex u to v is labelled $\ell + \delta$ (ℓ is a linear form and δ is a constant). We replace it with $2d + 1$ edges. We add edges from $u^{(i)}$ to $v^{(i)}$ with label δ for $0 \leq i \leq d$. And we add edges from $u^{(i)}$ to $v^{(i+1)}$ with the label ℓ for $0 \leq i \leq d - 1$. This ABP now computes the same polynomial as before and is *homogeneous*.



To make the length d, we modify it so that all vertices computing degree i polynomials are in the layer i (this makes the width s). If some of these vertices have no incoming edges from layer $i-1$, we can safely remove them. Note that the edges between layers will be linear forms. But we may have edges labeled with constants between two vertices in the i-th layer due to our reorganisation.



So for every vertex u in the i-th layer, and vertex w in the $(i+1)$-th layer, we add an edge with a linear form obtained by the *sub*-ABP between u and w. Then we drop all the in-layer edges. This gives a homogeneous ABP of d layers with all edges being *linear forms*. Indeed the edges we added initially were already linear forms, and the sub-ABPs all compute linear forms as well since every path is of length 2 with one edge label being a constant and the other being a linear form. Note that there are multiple output vertices now. In layer i for example, the sum of the polynomials computed at vertices with no outgoing edges is the degree i homogeneous component of f. □

Indian Institute of Technology Kanpur
Email address: bhargav@cse.iitk.ac.in
URL: https://bhargavcs.github.io

Indian Institute of Technology Kanpur
Email address: pdwivedi@cse.iitk.ac.in
URL: https://www.prateekdwivedi.in

Indian Institute of Technology Kanpur
Email address: nitin@cse.iitk.ac.in
URL: https://www.cse.iitk.ac.in/users/nitin/