Lower Bounds for the Sum of Small-Size Algebraic Branching Programs

C. S. Bhargav(✉), Prateek Dwivedi, and Nitin Saxena

Indian Institute of Technology, Kanpur, Kanpur, India
{bhargav,pdwivedi,nitin}@cse.iitk.ac.in

Abstract. We observe that proving strong enough 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 poly(n) ABPs, each of size (:= number of vertices) $(nd)^{o(1)}$, cannot compute the family of Iterated Matrix Multiplication polynomials $\mathrm{IMM}_{n,d}$ 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) and read-k oblivious ABPs. Both smABP and ROABP are very well-studied 'simple' models; our work puts them at the forefront of understanding Valiant's conjecture.

Keywords: Algebraic Circuits · Algebraic Branching Programs · Polynomials · Lower Bounds

1 Introduction

In a pioneering work, Leslie Valiant proposed [38] 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 (×), 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 ×) 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 [7,24,35].

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

X. Chen and B. Li (Eds.): TAMC 2024, LNCS 14637, pp. 355–366, 2024.
https://doi.org/10.1007/978-981-97-2340-9_30

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 poly(n) from now on). On the other hand, if a polynomial has degree poly(n) and the coefficient of any given monomial can be computed in $\#\mathsf{P}/\text{poly}$, 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 in the boolean world, very little is known in general about Valiant's hypothesis. A result of Strassen [36] and Baur-Strassen [5] 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 poly(n) form the class VF. We refer the interested reader to the excellent book of Bürgisser [6] 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 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 [8] 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 [5] 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 $\text{IMM}_{n,d}$ 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 $n \times n$

[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 [35, Definition 1.3].

matrices). The polynomial is defined as the $(1,1)$-th entry of the matrix product $X_1 \cdot X_2 \cdots X_d$:

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

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

Note that the polynomial $\mathrm{IMM}_{n,d}$ has an ABP of size $O(nd)$. The above theorem shows that this is almost optimal: we cannot reduce the size significantly, even by using a sum of polynomially many ABPs.

Remark 1. When $d > n^{o(1)}$, ABPs of size $(nd)^{o(1)}$ cannot 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 $(nd)^{o(1)}$.

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 $(nd)^{o(1)}$ 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 (exponential) sum of width-1 ABPs (monomials).

The lower bound of Theorem 1 also holds if we replace IMM with an appropriate polynomial from the family of Nisan-Wigderson design-based polynomials.

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. The most

natural ABP for the polynomial $\mathrm{IMM}_{n,d}$ is also set-multilinear: each layer (other than the first and the last) has n nodes and the edge connecting the p-th node in layer i to the q-th node in layer $i+1$ is labeled by $x_{i,pq}$.

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 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* $\mathrm{poly}(n)$*, then* $\mathsf{VBP} \neq \mathsf{VNP}$.

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. Interestingly, analogous reductions to the set-multilinear case were known for formulas [29, Theorem 3.1] and circuits [26, Lemma 2.11]. A series of works [2,17,19,37,39] 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.

Recently, [22] proved near-optimal lower bounds against set-multilinear formulas for a polynomial in VBP. Surprisingly, if the polynomial were computable by an smABP, we would obtain general formula lower bounds. This further illustrates the need to study smABPs.

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 Sect. 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 Sect. 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 [23] showed how to prove superpolynomial

[2] This definition differs slightly from that of Forbes [13] 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.

lower bounds for constant-depth set-multilinear formulas when the degree is small (set-multilinear lower bounds against arbitrary depth were known before [26,28,31], 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.

A Spectrum of Hardness Escalation

We also give a smooth generalization of Theorem 2 using more general versions of both set-multilinear polynomials and smABPs. The variable set is partitioned as before: $X = X_1 \sqcup \ldots \sqcup X_d$ with $|X_i| \leq n$ for all i.

A polynomial g is called *set-multi-k-ic* with respect to X if every monomial of g has exactly k variables (with multiplicity) from each of the d sets. That is, for a monomial m (seen as a multiset) in the support of g, $|m \cap X_i| = k$. When $k = 1$, the polynomial g is set-multilinear.

We call an ABP of length kd a *set-multi-k-ic* ABP (denoted sm(k)ABP) if every layer has edges labeled by linear forms from exactly one of the sets X_i, and there are exactly k layers corresponding to each X_i. As a special case, an sm(1)ABP is just a set-multilinear ABP as defined before.

Theorem 3 (Hardness bootstrapping spectrum). *Let n, d, k be integers such that* $\min(d^{kd}, (kd)^d) = \text{poly}(n)$*, and let $P_{n,d,k}$ be a* set-multi-k-ic *polynomial in* VNP *of degree kd. If $P_{n,d,k}$ cannot be computed by a* $\sum$sm(k)ABP *of width* poly(n)*, then* VBP $\neq$ VNP*.*

Remark 2. We note that Theorem 2 is an immediate consequence of Theorem 3 when $k = 1$. An added advantage of this generalization is the flexibility with the degree of the hard polynomial. For example, if $k = d = O(\log n / \log\log n)$, the degree of the polynomial we are allowed is $O(\log^2 n/(\log\log n)^2)$. In contrast, Theorem 2 could only work when the degree is $O(\log n/\log\log n)$.

The *set-multi-k-ic* ABP is inspired from the well-studied *multi-k-ic* depth-restricted circuits and formulas, initiated by Kayal and Saha [18]. We encourage readers to refer [32, Chapter 14] and references therein for a comprehensive discussion.

1.2 The Sum of ROABPs Perspective: The Arbitrarily Low Variate Case

One can also view Theorem 2 through the lens of another well-studied model in the literature, first defined by Forbes and Shpilka [12]. An algebraic branching program over the variables $(x_1, \ldots, x_n)$ is said to be *oblivious* if, for every layer, all the edge labels are univariate polynomials in a single variable. It is further called a *read-once* oblivious ABP (or a ROABP) if every variable appears in at most one layer.

A ROABP in the *natural order* is $n+1$ layered ABP where 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 a 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 [25] 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 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 (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 $\mathsf{VBP} \neq \mathsf{VNP}$.*

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 [33,34] for a survey of PIT and techniques used).

Recently, several surprising results [1,16,21] 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 [20] for an exposition of the ideas involved.

We now state a corollary of Theorem 3 analogous to Corollary 1. An *oblivious* ABP is said to be *read-k* if each variable x_i appears in at most k layers. We denote the sum of *read-k* oblivious ABPs as $\sum$R(k)O. Once again, the width of a $\sum$R(k)O is the sum of the widths of the constituent branching programs.

Corollary 2. *Let n, d, k be integers such that $\min(n^{kn}, (kn)^n) = \mathrm{poly}(d)$. Let $f \in \mathsf{VNP}$ be a polynomial on n variables of individual degree d. If f cannot be computed by a $\sum$R(k)O of width $\mathrm{poly}(d)$, then $\mathsf{VBP} \neq \mathsf{VNP}$.*

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 [15] 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 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), 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 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. The proof of Theorem 3 is similar.

Lower Bounds for the Sum of ABPs. Our proof of the $\sum$ABP lower bound (Theorem 1) uses the implicit reduction of Theorem 2 to $\sum$smABP. Using Nisan's characterization [25] mentioned before, we can prove exponential lower bounds against single smABPs (ROABPs), but the characterization does not extend to their sums. There has been progress in handling the sums in recent years, which we now briefly describe.

Arvind and Raja [4] proved a superpolynomial lower bound for the Permanent polynomial against the sum of sub-linear many ROABPs (the bound is exponential if the number of ROABPs is bounded by a constant). Ramya and Rao [27] showed that a sum of sub-exponential size ROABPs computing the multilinear polynomial defined by Raz and Yehudayoff [30] needs exponentially many summands. Ghosal and Rao [14] showed an exponential lower bound for the sum of ROABPs computing the multilinear polynomial defined by Dvir, Malod, Perifel and Yehudayoff [11], provided each of the constituent ABPs is polynomial in size.

Unfortunately, these results do not imply general ABP lower bounds using our hardness escalation theorems, as they only work in regimes where the degree and number of variables are comparable. Viewed differently, they cannot handle a sum of $d!$ smABPs (or $n!$ ROABPs) which is necessary to prove lower bounds in our low-degree (low-variate) regime. In a very recent work Chatterjee, Kush, Saraf and Shpilka [9] improve the bounds in the above works and also prove superpolynomial lower bounds against the sum of smABPs when the degree is $d = \omega(\log n)$. Improving this to work for $d = O(\log n / \log\log n)$ would have dramatic consequences.

Fewer results are known about *read-k* oblivious ABPs. They were studied in [3] as a natural generalization of ROABPs and a lower bound of $\exp(n/k^{O(k)})$ for a single *read-k* oblivious ABP was shown. It remains open to improve this

result to prove non-trivial lower bounds when k is large, as well as to prove lower bounds for sums of *read-k* oblivious ABPs. When k is small, the results of Ramya and Rao [27] extend to the sum of multilinear k-pass ABPs, a restriction of *read-k* oblivious ABPs in which the variables are read k times in sequence, each time in a possibly different order.

We demonstrate a way to handle our low-degree regime in certain cases. To prove lower bounds for the sum of smABPs, we use the *partial derivative method*, introduced in the highly influential work of Nisan and Wigderson [26]. 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 results 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 [10,32].

We work with the polynomial $\mathrm{IMM}_{n,d}$, which gives us more flexibility in 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, as we saw before. 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 usual 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 smABPs are sub-polynomial in size. It also enables us to handle extremely large sums of smABPs (including those that occur from considering sums of multiple ABPs).

This approach works in the low-degree regime, since our reductions are 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 Hardness Bootstrapping Spectrum

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 ([23], Proposition 9) for small-depth circuits. Due to space constraints, we omit detailed proofs which can be found in the full version.[3]

Lemma 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.*

[3] Full version - https://www.cse.iitk.ac.in/users/nitin/papers/sumRO.pdf.

We immediately obtain Theorem 2 as an easy consequence. We omit the proof. In order to prove Lemma 1, we first homogenize the ABP (similar to the approach of Raz [29] and LST [23]). 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 (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 [25]. As our central argument, we show that this homogeneous ABP can be *efficiently* set-multilinearized.

Proposition 1. *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 $P_{n,d}$.*

With this transformation in hand, we can complete the reduction and obtain Lemma 1. The proof of Theorem 3 follows the template of Theorem 2. We begin with ABP homogenization, followed by a structural transformation to the sum of *set-multi-k-ic* ABP. The superpolynomial lower bound assumption on $\sum$sm(k)ABP gives the desired separation result. The following lemma is analogous to Lemma 1.

Lemma 3 (ABP to $\sum$sm(k)ABP). *Let P be a set-multi-k-ic polynomial 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 can be computed by an* ABP *of size s, then there is a $\sum$*sm(k)ABP *of width $s \cdot \binom{d+kd}{d}$ computing the same polynomial.*

It is straightforward to prove Theorem 3 using the above lemma. The proof is similar to Theorem 2 and we omit it.

3 Lower Bound for the Sum of ABPs

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

Lemma 4. *Any $\sum$*smABP *computing the polynomial $\mathrm{IMM}_{n,d}$ with $d = O(\log n / \log\log n)$, must have width at least $n^{\Omega(1)}$.*

Suppose we had to prove the lower bound of Theorem 1 for a single ABP computing IMM. We could then use Lemma 4 above in conjunction with Lemma 1 to conclude the result. But when we are dealing with a sum of ABPs, we need to be more careful in how we set-multilinearize since the ABPs no longer need to compute set-multilinear or even homogenous polynomials.

4 Discussion and Open Problems

In order to separate VBP from VNP, we 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 cannot be used for such a separation. Our Theorem 1 also holds for a polynomial from the Nisan-Wigderson family of design-based polynomials that is in VNP and is a better candidate.

A first step toward proving ABP lower bounds would be to prove any non-trivial lower bounds against the sum of smABPs in the low degree regime, i.e. prove some lower bound for the sum of $d!$ smABPs. 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. Agrawal, M., Ghosh, S., Saxena, N.: Bootstrapping variables in algebraic circuits. Proc. Natl. Acad. Sci. U.S.A. **116**(17), 8107–8118 (2019). https://doi.org/10.1073/pnas.1901272116
2. Agrawal, M., Vinay, V.: Arithmetic circuits: a chasm at depth four. In: 2008 49th Annual IEEE Symposium on Foundations of Computer Science, pp. 67–75 (2008). https://doi.org/10.1109/FOCS.2008.32
3. Anderson, M., Forbes, M.A., Saptharishi, R., Shpilka, A., Volk, B.L.: Identity testing and lower bounds for read-k oblivious algebraic branching programs. ACM Trans. Comput. Theory **10**(1), 3:1–3:30 (2018). https://doi.org/10.1145/3170709
4. Arvind, V., Raja, S.: Some lower bound results for set-multilinear arithmetic computations. Chic. J. Theor. Comput. Sci. pp. Art. 6, 26 (2016). https://doi.org/10.4086/cjtcs.2016.006
5. Baur, W., Strassen, V.: The complexity of partial derivatives. Theor. Comput. Sci. **22**(3), 317–330 (1983). https://doi.org/10.1016/0304-3975(83)90110-X
6. Bürgisser, P.: Completeness and Reduction in Algebraic Complexity Theory, Algorithms and Computation in Mathematics, vol. 7. Springer, Berlin (2000). https://doi.org/10.1007/978-3-662-04179-6
7. Bürgisser, P., Clausen, M., Shokrollahi, M.A.: Algebraic complexity theory, Grundlehren der mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences], vol. 315. Springer, Berlin (1997). https://doi.org/10.1007/978-3-662-03338-8, with the collaboration of Thomas Lickteig

8. Chatterjee, P., Kumar, M., She, A., Volk, B.L.: Quadratic lower bounds for algebraic branching programs and formulas. Comput. Complexity **31**(2), Paper No. 8, 54 (2022). https://doi.org/10.1007/s00037-022-00223-8
9. Chatterjee, P., Kush, D., Saraf, S., Shpilka, A.: Lower bounds for set-multilinear branching programs (2024). https://arxiv.org/abs/2312.15874, preprint
10. Chen, X., Kayal, N., Wigderson, A.: Partial derivatives in arithmetic complexity and beyond. Found. Trends Theor. Comput. Sci. **6**(1-2), 1–138 (2010). https://doi.org/10.1561/0400000043
11. Dvir, Z., Malod, G., Perifel, S., Yehudayoff, A.: Separating multilinear branching programs and formulas. In: Proceedings of the 2012 ACM Symposium on Theory of Computing, STOC 2012, pp. 615–623. ACM, New York (2012). https://doi.org/10.1145/2213977.2214034
12. Forbes, M.A., Shpilka, A.: Quasipolynomial-time identity testing of non-commutative and read-once oblivious algebraic branching programs. In: 2013 IEEE 54th Annual Symposium on Foundations of Computer Science, FOCS 2013, Los Alamitos, CA, pp. 243–252. IEEE Computer Society (2013). https://doi.org/10.1109/FOCS.2013.34
13. Forbes, M.A.: Polynomial Identity Testing of Read-Once Oblivious Algebraic Branching Programs. ProQuest LLC, Ann Arbor, MI, thesis (Ph.D.)–Massachusetts Institute of Technology (2014)
14. Ghosal, P., Rao, B.V.R.: Limitations of sums of bounded read formulas and ABPs. In: Santhanam, R., Musatov, D. (eds.) CSR 2021. LNCS, vol. 12730, pp. 147–169. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-79416-3_9
15. Grochow, J.: Degree restriction for polynomials in VP. Theoretical Computer Science Stack Exchange (2013). https://cstheory.stackexchange.com/q/19268
16. Guo, Z., Kumar, M., Saptharishi, R., Solomon, N.: Derandomization from algebraic hardness. SIAM J. Comput. **51**(2), 315–335 (2022). https://doi.org/10.1137/20M1347395
17. Gupta, A., Kamath, P., Kayal, N., Saptharishi, R.: Arithmetic circuits: a chasm at depth 3. SIAM J. Comput. **45**(3), 1064–1079 (2016). https://doi.org/10.1137/140957123
18. Kayal, N., Saha, C.: Multi-k-ic depth three circuit lower bound. Theory Comput. Syst. **61**(4), 1237–1251 (2017). https://doi.org/10.1007/S00224-016-9742-9
19. Koiran, P.: Arithmetic circuits: the chasm at depth four gets wider. Theor. Comput. Sci. **448**, 56–65 (2012). https://doi.org/10.1016/j.tcs.2012.03.041
20. Kumar, M., Saptharishi, R.: Hardness-randomness tradeoffs for algebraic computation. Bull. Eur. Assoc. Theor. Comput. Sci. EATCS **3**(129), 56–87 (2019). http://bulletin.eatcs.org/index.php/beatcs/article/view/591/599
21. Kumar, M., Saptharishi, R., Tengse, A.: Near-optimal bootstrapping of hitting sets for algebraic circuits. In: Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, Philadelphia, PA, pp. 639–646. SIAM (2019). https://doi.org/10.1137/1.9781611975482.40
22. Kush, D., Saraf, S.: Near-optimal set-multilinear formula lower bounds. In: 38th Computational Complexity Conference, LIPIcs. Leibniz Int. Proc. Inform., vol. 264, pp. Art. No. 15, 33. Schloss Dagstuhl. Leibniz-Zent. Inform., Wadern (2023). https://doi.org/10.4230/lipics.ccc.2023.15
23. Limaye, N., Srinivasan, S., Tavenas, S.: Superpolynomial lower bounds against low-depth algebraic circuits. In: 2021 IEEE 62nd Annual Symposium on Foundations of Computer Science, FOCS 2021, Los Alamitos, CA, pp. 804–814. IEEE Computer Society (2021). https://doi.org/10.1109/FOCS52979.2021.00083

24. Mahajan, M.: Algebraic complexity classes. In: Perspectives in Computational Complexity, Progr. Comput. Sci. Appl. Logic, vol. 26, pp. 51–75. Birkhäuser/Springer, Cham (2014). https://arxiv.org/abs/1307.3863
25. Nisan, N.: Lower bounds for non-commutative computation. In: Proceedings of the Twenty-Third Annual ACM Symposium on Theory of Computing, STOC 1991, pp. 410–418. Association for Computing Machinery, New York (1991). https://doi.org/10.1145/103418.103462
26. Nisan, N., Wigderson, A.: Lower bounds on arithmetic circuits via partial derivatives. Comput. Complex. **6**(3), 217–234 (1996). https://doi.org/10.1007/BF01294256
27. Ramya, C., Raghavendra Rao, B.V.: Lower bounds for special cases of syntactic multilinear ABPs. Theor. Comput. Sci. **809**, 1–20 (2020). https://doi.org/10.1016/j.tcs.2019.10.047
28. Raz, R.: Multi-linear formulas for permanent and determinant are of super-polynomial size. J. ACM **56**(2), Art. 8, 17 (2009). https://doi.org/10.1145/1502793.1502797
29. Raz, R.: Tensor-rank and lower bounds for arithmetic formulas. J. ACM **60**(6), Art. 40, 15 (2013). https://doi.org/10.1145/2535928
30. Raz, R., Yehudayoff, A.: Balancing syntactically multilinear arithmetic circuits. Comput. Complexity **17**(4), 515–535 (2008). https://doi.org/10.1007/s00037-008-0254-0
31. Raz, R., Yehudayoff, A.: Lower bounds and separations for constant depth multilinear circuits. Comput. Complexity **18**(2), 171–207 (2009). https://doi.org/10.1007/s00037-009-0270-8
32. Saptharishi, R.: A survey of lower bounds in arithmetic circuit complexity. Github Survey (2021). https://github.com/dasarpmar/lowerbounds-survey
33. Saxena, N.: Progress on polynomial identity testing. Bull. Eur. Assoc. Theor. Comput. Sci. EATCS (99), 49–79 (2009). https://www.cse.iitk.ac.in/users/nitin/papers/pit-survey09.pdf
34. Saxena, N.: Progress on polynomial identity testing-II. In: Perspectives in Computational Complexity, Progr. Comput. Sci. Appl. Logic, vol. 26, pp. 131–146. Birkhäuser/Springer, Cham (2014). https://arxiv.org/abs/1401.0976
35. Shpilka, A., Yehudayoff, A.: Arithmetic circuits: a survey of recent results and open questions. Found. Trends Theor. Comput. Sci. **5**(3–4), 207–388 (2009). https://doi.org/10.1561/0400000039
36. Strassen, V.: Die Berechnungskomplexität von elementarsymmetrischen Funktionen und von Interpolationskoeffizienten. Numer. Math. **20**, 238–251 (1972/73). https://doi.org/10.1007/BF01436566
37. Tavenas, S.: Improved bounds for reduction to depth 4 and depth 3. Inform. Comput. **240**, 2–11 (2015). https://doi.org/10.1016/j.ic.2014.09.004
38. Valiant, L.G.: Completeness classes in algebra. In: Conference Record of the Eleventh Annual ACM Symposium on Theory of Computing (Atlanta, Ga., 1979), pp. pp 249–261. ACM, New York (1979). https://doi.org/10.1145/800135.804419
39. Valiant, L.G., Skyum, S., Berkowitz, S., Rackoff, C.: Fast parallel computation of polynomials using few processors. SIAM J. Comput. **12**(4), 641–644 (1983). https://doi.org/10.1137/0212043