VNP=VP in the multilinear world

Meena Mahajan[a], Nitin Saurabh[a], Sébastien Tavenas[b,1]

[a] *The Institute of Mathematical Sciences, Chennai 600113, India.*
{meena,nitin}@imsc.res.in
[b] *Microsoft Research India, Bangalore 560001, India.* t-sebat@microsoft.com

Abstract

In this note, we show that over fields of any characteristic, exponential sums of Boolean instantiations of polynomials computed by multilinear circuits can be computed by multilinear circuits with polynomial blow-up in size. In particular, multilinear-VNP equals multilinear-VP. Our result showing closure under exponential sums also holds for other restricted multilinear classes – polynomials computed by multilinear (bounded-width) algebraic branching programs and formulas. Furthermore, it holds even if the circuit class is not fully multilinear but computes a polynomial that is multilinear in the summation variables.

Key words: algebraic complexity, VP, VNP, circuits, branching programs, multilinearity

1. Introduction

Valiant [1, 2] introduced algebraic complexity theory to study the complexity of polynomial families. One of the most fundamental problems in algebraic complexity theory is to decide whether the classes VP and VNP are distinct. These classes were first defined by Valiant in [1, 2] as the algebraic analogues of the Boolean complexity classes P and NP (however, a closer look at the definitions reveals that they are really an analogue of LOGCFL and #P; LOGCFL is a subclass of P). For basics and detailed treatment of the subject we refer the reader to [3, 4]. We borrow notations from [4, 5]. VP is the class of sequences of polynomials with polynomially bounded degree that are

[1]This work was done when the author was at ENS Lyon, France and supported by ANR project CompA (project number: ANR-13-BS02-0001-01).

computable by polynomial sized arithmetic circuits. Moreover, a sequence of polynomials (f_n) belongs to VNP if and only if there exist polynomials p and q, and a sequence $(g_n) \in \mathsf{VP}$ such that for all n,

$$f_n(x_1, \ldots, x_{q(n)}) = \sum_{\bar{y} \in \{0,1\}^{p(\bar{x})}} g_n(x_1, \ldots, x_{q(n)}, y_1, \ldots, y_{p(n)}).$$

So, in other words, one can think of VNP as *exponential sums* of polynomial sized circuits; $\mathsf{VNP} = \sum \cdot \mathsf{VP}$. Hence the VP versus VNP question can also be thought of as understanding the power of exponential sums. In the foundational paper [2], Valiant observed that exponential sums of polynomial sized formulas $(\sum \cdot \mathsf{VF})$ or $(\sum \cdot \mathsf{VP_e})$ exactly capture exponential sums of polynomial sized circuits $(\sum \cdot \mathsf{VP})$. That is, $\mathsf{VNP}_e = \mathsf{VNP}$ (see also [6]). He used this observation crucially to show that the *permanent* polynomial is VNP-hard. VBP is the class of polynomial families computed by algebraic branching programs of polynomial size; equivalently, families that can be represented as the *determinant* of a polynomially large matrix, where the entries are either field elements or variables. It is known that $\mathsf{VF} \subseteq \mathsf{VBP} \subseteq \mathsf{VP}$; hence from Valiant's observation, it follows that $\sum \cdot \mathsf{VF} = \sum \cdot \mathsf{VBP} = \sum \cdot \mathsf{VP} = \mathsf{VNP}$.

Valiant's observation raises a natural question to study: How powerful are *exponential sums* of *restricted* circuit classes?

A natural restriction on arithmetic circuits is *multilinearity*. A polynomial is called *multilinear* if each variable in the polynomial has degree at most 1. An arithmetic circuit is called *multilinear* if every gate in it computes a multilinear polynomial. Furthermore, if for every product gate, the subcircuits rooted at the left and right child are variable-disjoint, then the circuit is called *syntactic multilinear*.

The exponential summation under the restriction of syntactic multilinearity was studied by Jansen et al. [7, 8, 5]. They showed that syntactic multilinear classes are closed under exponential sums. In particular, exponential summation does not add any power to syntactic multilinear formulas. Contrast this with the case of general formulas, where it become as powerful as VNP. Exponential summations of polynomials were also studied by Juma et al. [9]. Their motivation was to obtain query algorithms for #SAT that are better than brute-force. They proved that over fields of characteristic different from 2, multilinear polynomials are closed under exponential sums (Observation 1.3, [9]).

In this note we study the exponential summation under the restriction of *multilinearity* (not necessarily syntactic). Using techniques different from those used in [5, 9], we extend their results by showing that over any field, exponential summation does not add power to multilinear circuit classes. In particular, since in the multilinear world we know that VF is strictly weaker than VBP ([10, 11]), our result implies that in the multilinear world we do not have an analogue of the collapse $\sum \cdot \mathsf{VF} = \sum \cdot \mathsf{VBP} = \sum \cdot \mathsf{VP}$ that holds in the general world. A corollary of our result is that $\mathsf{VNP} = \mathsf{VP}$ in the multilinear setting, whereas we do not believe that a similar thing holds in non-multilinear setting. Thus our result highlights essential differences between the general and multilinear worlds, and indicates that separations/collapses in the restricted multilinear world may have no bearing on the true state of affairs in the general world.

We obtain our result (Theorem 1) by considering summations of general polynomials, but the summation is over variables that have degree at most 1 in the polynomial. We show that such a summation over multilinear variables is as good as evaluating the polynomial at one or a small number of points.

2. Preliminaries

An arithmetic circuit is a directed acyclic graph with leaves labeled by variables or field elements, and internal nodes (called gates) labeled by one of the field operations $+$ and $\times$. The vertex with out-degree 0 is called the output gate. Note that each gate computes a polynomial in a natural way. The polynomial computed by the circuit is the polynomial computed at the output gate of the circuit. If the out-degree of each internal node is 1, that is, the undirected graph underlying the circuit is a tree, then the circuit is called a *formula*. The *size* of a circuit or formula is the number of gates in it including the input gates. We say that a circuit is *layered* if there is a partition $V_1, \ldots, V_\ell$ of non-input gates in the circuits into layers such that all incoming edges of V_i are either from input gates or from $V_{i-1}, 1 < i \leq \ell$. The *width* of a layered circuit is the maximum number of gates per layer.

An algebraic branching program (ABP) is a directed acyclic graph with two distinguished vertices s and t. Vertex s has in-degree 0, and vertex t has out-degree 0. Each edge is labeled with either a field element or a variable. The *weight* of a path from s to t is the product of the labels of the edges appearing in the path. The polynomial computed by an algebraic branching program G is the sum of the weights of all paths from s to t in G. The *size*

of an algebraic branching program is the number of vertices in it. An ABP is called *layered* if the vertices can be partitioned into layers such that edges are present only in between consecutive layers. The *width* of a layered ABP is the maximum number of nodes in any layer.

A variable x is said to have degree at most d in the polynomial f if each non-zero monomial in f has degree at most d in x and at least one monomial containing x has degree exactly d. We say that a polynomial $f(x_1, \ldots, x_n, y_1, \ldots, y_n)$ is multilinear in the $Y = \{y_1, \ldots, y_n\}$ variables if all variables in Y have degree at most 1 in f. Thus a polynomial over the set of variables X is called *multilinear* if the polynomial is multilinear in X. By a *multilinear* circuit/formula we mean a circuit/formula computing multilinear polynomials at each gate in the circuit/formula.

3. Exponential sums of multilinear polynomials

We now state and prove our main theorem.

Theorem 1. *Let $f(x_1, \ldots, x_N, y_1, \ldots, y_m)$ be a polynomial that is multilinear in the $Y = \{y_1, \ldots, y_m\}$ variables. Let $h(X)$ be the exponential sum polynomial*

$$h(X) = \sum_{e \in \{0,1\}^m} f(X, e_1, \ldots, e_m).$$

If f has an efficient computation, so does h. The following table gives upper bounds on the complexity measures of h in terms of the corresponding measures of f.

		Char $\neq 2$	*Char* $= 2$	
			infinite fields	*finite fields*
Circuit (size,width)	f	s, w	s, w	s, w
	h	$s+1, w$	$3s(m+1), w+1$	$s(m+1)^2, w(m+1)$
ABP (size,width)	f	s, w	s, w	s, w
	h	$s+1, w$	$3s(m+1), w+2$	$s(m+1), w(m+1)$
Formula size	f	s	s	s
	h	$s+1$	*O(s) [5]*	*O(s) [5]*

Furthermore, if the circuit/ABP/formula for f is multilinear, then so is the circuit/ABP/formula for h.

Proof: Let $f(x_1, \ldots, x_N, y_1, \ldots, y_m)$ be some polynomial that is multilinear in the $Y = \{y_1, \ldots, y_m\}$ variables. Then there are polynomials $f_S(X)$ in the

variables $X = \{x_1, \ldots, x_N\}$, one for each $S \subseteq [m]$, such that we can express f in terms of them:

$$f(X, Y) = \sum_{S \subseteq [m]} f_S(X) \prod_{i \in S} y_i.$$

Now consider the exponential Boolean sum

$$\begin{aligned} h(X) &= \sum_{e \in \{0,1\}^m} f(X, e) = \sum_{e \in \{0,1\}^m} \sum_{S \subseteq [m]} f_S(X) \prod_{i \in S} e_i \\ &= \sum_{e \in \{0,1\}^m} \sum_{S \subseteq [m] : i \in S \Rightarrow e_i = 1} f_S(X) \\ &= \sum_{S \subseteq [m]} f_S(X) \sum_{e \in \{0,1\}^m : i \in S \Rightarrow e_i = 1} 1 \\ &= \sum_{S \subseteq [m]} f_S(X) 2^{m - |S|}. \end{aligned}$$

Char $\neq 2$. If the field has characteristic other than 2, then

$$h(X) = 2^m \sum_{S \subseteq [m]} f_S(X) 2^{-|S|} = 2^m f(x_1, \ldots, x_N, 1/2, \ldots, 1/2).$$

Since $f(x_1, \ldots, x_N, 1/2, \ldots, 1/2)$ is a projection of $f(X, Y)$, we see that if f is computed by a multilinear circuit C, then C' obtained by setting all the y_i variables to $1/2$ is also multilinear (the polynomials at each node are projections of the respective polynomials in C). Multiplying the output of C' with 2^m gives a circuit for h. This observation was also made in [9]. Note that the same thing can be done for ABPs or formulas, again with just one extra node, and no increase in width.

Char $= 2$. If the field $\mathbb{F}$ has characteristic 2, then a little more work is needed to compute $h(X)$. We see that for $|S| < m$, the contribution from f_S to h vanishes due to characteristic 2, and we are left with

$$h(X) = \sum_{S \subseteq [m]} f_S(X) 2^{m - |S|} = f_{[m]}(x_1, \ldots, x_N).$$

So we need to compute $f_{[m]}(X)$. The polynomial

$$g(X, y) \triangleq f(x_1, \ldots, x_n, y, y, \ldots, y)$$

may be viewed as a univariate polynomial $g'(y)$ in $G[y]$, where G is the ring $\mathbb{F}[X]$. Then $f_{[m]}(X)$ is just the coefficient of y^m in g'. (Note that $g'(y)$ has degree at most m, since f is multilinear in Y.)

If a circuit C of size s computes f, then setting each $y_i \in Y$ to y gives circuit C', also of size s, computing g'. Now there are two cases to consider.

Infinite fields. This is the easier case, and we give a construction that even preserves width, using the standard interpolation trick. Let $g'(y) = \sum_{i=0}^{m} c_i y^i$ where $c_i \in \mathbb{F}[X]$. Pick $m+1$ distinct values α_j from $\mathbb{F}$ and consider the system of equations $\sum_{i=0}^{m} c_i \alpha_j^i = g'(\alpha_j); \quad 0 \leq j \leq m$. More succinctly, $V[c_0, \ldots, c_m]^t = [g'(\alpha_o), g'(\alpha_1), \ldots, g'(\alpha_m)]^t$, where V is a Vandermonde matrix and is hence invertible. Hence c_m is a linear combination of the polynomials $g'(\alpha_o), g'(\alpha_1), \ldots, g'(\alpha_m)$ computed by distinct copies of C'. By increasing the depth/length, this linear combination can be computed in a way that increases the width of a circuit only by 1 and of an ABP only by 2.

It follows easily that if f is computed by a multilinear circuit, then the circuit obtained above is also multilinear.

Finite fields. Using the standard procedure of homogenization (see [3, 4]), we can obtain circuits $C_0, C_1, \ldots, C_m$ for the homogeneous components of g'. The circuit $D(X) = C_m(X, 1)$, is the desired circuit for h. The size is bounded by $s(m+1)^2$, and width by $w(m+1)$.

It remains to see why D is multilinear when C is multilinear. For this, we need to look at the structure of the homogeneous circuit obtained by the homogenization procedure. For every gate u in C, we introduce $m+1$ gates in the homogeneous circuit. We refer to these $m+1$ copies as the major gates. Each major gate computes a homogeneous part of the polynomial computed at u in C. Therefore, a major gate corresponds to a gate u in C, and a degree $i \in \{0, 1, \ldots, m\}$; we refer to this gate as $[u, i]$. Hence the output gate of the circuit $D(X)$ is the gate $[r, m]$, where r is the output gate of C. The edge connections in the homogeneous circuit D are defined inductively. For the input gates we label the copies appropriately. If $u = v + z$, we make the connections based on the rule $[u, i] = [v, i] + [z, i]$ for all i. Otherwise if $u = v \times z$, the connections are based on the rule $[u, i] = \sum_{k=0}^{i} [v, k] \times [z, i-k]$. In this last case, we refer to the intermediate (multiplication) gates used at each major gate to accumulate the homogeneous parts as the minor gates.

Let $p_C(X, Y)$ and $p_D(X)$ be the polynomials computed at u in C and at $[u, i]$ in D respectively. We know that $p_C(X, Y)$ is multilinear in X and Y. Hence in the expression $p_C(X, y, y, \ldots, y) = \sum_{j=0}^{m} p_{C,j}(X) y^j$, each $p_{C,j}(X)$ is multilinear. By construction, $p_D(X) = p_{C,i}(X)$; hence it is multilinear.

Now consider the minor gates. It seems possible that two minor gates compute non-multilinear terms that cancel out when accumulated in the major gate. We need to show that this does not happen, and that the minor gates also compute multilinear polynomials.

Lemma 1. *Let α, β and γ be three gates in C such that $\alpha = \beta \times \gamma$. If the three gates are multilinear, then the variables appearing in the polynomials computed by β and γ are disjoint.*

Proof: With slight abuse of notation, let α, β, γ also denote the polynomials computed by the respective gates. Suppose there exists a variable v which is in β and in γ. Consider the total order on the variables: $x_1 < \ldots < x_n < y_1 < \ldots < y_m$ and the derived lexicographic order on the monomials. Let m_β and m_γ be the maximal monomials in β and γ which contain v. We show that the monomial $m_\beta m_\gamma$ is in α. If it is not, it is cancelled by some other monomial $n_\beta n_\gamma$. But, $m_\beta m_\gamma$ contains v^2, so n_β and n_γ must both contain v. By assumption, $n_\beta \leq m_\beta$ and $n_\gamma \leq m_\gamma$. So, $n_\beta n_\gamma \leq m_\beta m_\gamma$ and the equality holds only if $m_\beta = n_\beta$ and $m_\gamma = n_\gamma$. Thus the monomial $m_\beta m_\gamma$ does not get cancelled and appears in α. Hence α is not multilinear, a contradiction. □

Lemma 2. *If $[u, i]$ is a major gate in D, then every variable appearing in the polynomial computed at $[u, i]$ also appears in the polynomial computed at the gate u in C.*

Proof: As before, let $p_D(X)$ and $p_C(X, Y)$ be the polynomials computed at the gates $[u, i]$ and u. Then, $p_D(X)$ is the coefficient of y^i in the polynomial $p_C(x_1, \ldots, x_n, y, \ldots, y)$. Consequently, the variables of $p_D(X)$ are included in the variables of $p_C(x_1, \ldots, x_n, y, \ldots, y)$. Moreover, as $p_C(x_1, \ldots, x_n, y, \ldots, y)$ is a projection of the polynomial $p_C(X, Y)$, the variables of $p_C(x_1, \ldots, x_n, y, \ldots, y)$ are included in the variables in $p_C(X, Y)$. □

A minor gate in D feeding into gate $[u, i]$ corresponds to some $k \in \{0, 1, \ldots, i\}$ and computes $[v, k] \times [z, i - k]$. We have already seen that the polynomials computed at the major gates $[v, k]$ and $[z, i-k]$ are multilinear. Furthermore, by Lemma 2, the variables of $[v, k]$ (respectively $[z, i - k]$) are a subset of the variables of v (respectively z). As v and z share no variables (Lemma 1), the same holds for $[v, k]$ and $[z, i-k]$ as well. Thus their product is also multilinear.

This completes the proof for circuits. The same homogenization trick works for ABPs as well, taking size s and width w to size $s(m+1)$ and width

$w(m+1)$. However for formulas, it could result in a huge blowup in size. But recall that multilinear formulas can be made syntactic multilinear without any increase in size [10]. Hence the result for multilinear formulas follows directly from [5]. □

From Theorem 1 we obtain the closure property of multilinear classes.

Corollary 2. *The following circuit classes are closed under exponential sums: multilinear poly-size bounded-width branching programs (m-VBWBP), multilinear poly-size formulas (m-VF), multilinear poly-size branching programs (m-VBP), and multilinear poly-size circuits (m-VP). In particular, we have m-VP $= m$-VNP.*

Acknowledgements

The authors discussed this work during the Workshops on Algebraic Complexity Theory at Aarhus, Denmark in March 2013 and at Saarbrücken, Germany in March 2014.

References

References

[1] L. Valiant, Completeness classes in algebra, in: Symposium on Theory of Computing STOC, 1979, pp. 249–261.

[2] L. Valiant, Reducibility by algebraic projections, in: Logic and Algorithmic: International Symposium in honour of Ernst Specker, Vol. 30, Monograph. de l'Enseign. Math., 1982, pp. 365–380.

[3] A. Shpilka, A. Yehudayoff, Arithmetic circuits: A survey of recent results and open questions, Foundations and Trends in Theoretical Computer Science 5 (3-4) (2010) 207–388.

[4] M. Mahajan, Algebraic complexity classes, in: Perspectives in Computational Complexity, Vol. 26 of Progress in Computer Science and Applied Logic, Springer International Publishing, 2014, pp. 51–75.

[5] M. Jansen, M. Mahajan, B. Rao, Resource trade-offs in syntactically multilinear arithmetic circuits, Computational Complexity 22 (3) (2013) 517–564.

[6] G. Malod, N. Portier, Characterizing Valiant's algebraic complexity classes, Journal of Complexity 24 (1) (2008) 16–38.

[7] M. Mahajan, B. Raghavendra Rao, Arithmetic circuits, syntactic multilinearity, and the limitations of skew formulae, in: Mathematical Foundations of Computer Science 2008, Vol. 5162 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2008, pp. 455–466.

[8] M. Jansen, R. Rao B.V., Simulation of arithmetical circuits by branching programs with preservation of constant width and syntactic multilinearity, in: Computer Science - Theory and Applications, Vol. 5675 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2009, pp. 179–190.

[9] A. Juma, V. Kabanets, C. Rackoff, A. Shpilka, The black-box query complexity of polynomial summation, Computational Complexity 18 (1) (2009) 59–79.

[10] R. Raz, Separation of multilinear circuit and formula size, Theory of Computing 2 (6) (2006) 121–135.

[11] Z. Dvir, G. Malod, S. Perifel, A. Yehudayoff, Separating multilinear branching programs and formulas, in: Proceedings of the Forty-fourth Annual ACM Symposium on Theory of Computing, STOC '12, ACM, 2012, pp. 615–624.