Electronic Colloquium on Computational Complexity, Report No. 124 (2018)

Separating Monotone VP and VNP

Amir Yehudayoff*

Abstract

This work is about the monotone versions of the algebraic complexity classes VP and VNP. The main result is that monotone VNP is strictly stronger than monotone VP.

1 Introduction

The central open question in algebraic complexity theory is the VP versus VNP problem. It is the algebraic analog of the P versus NP problem. In a nutshell, it asks whether every "explicit" polynomial can be efficiently computed. Here we prove that in the monotone setting VP $\neq$ VNP. Namely, there are "explicit monotone" polynomials that can not be efficiently computed by a monotone circuit.

Algebraic Complexity

Algebraic complexity theory is the study of the computational complexity of polynomials using algebraic operations. In it, VP is the algebraic analog of P, and VNP is the algebraic analog of NP. A boolean sum $\sum$ is the algebraic analog of the existential quantifier $\exists$ from the boolean setting. The boolean P versus NP question (roughly speaking) is about the question "can an existential quantifier significantly reduce running time?" The analogous algebraic question is "can a boolean sum significantly reduce circuit size?"

More formally, VP is the class of polynomials[1] p that have polynomial size algebraic circuits (we assume that the underlying field is $\mathbb{R}$). VNP is the class of polynomials q that can be written as $q(x) = \sum_{b \in \{0,1\}^{\mathrm{poly}(n)}} p(x, b)$ where p is in VP and n is the number of variables in q. For more background, motivation and formal definitions, see [2, 1, 15].

*Department of Mathematics, Techion-IIT. amir.yehudayoff@gmail.com. Research supported by ISF grant 1162/15.

[1] Actually, of families of polynomials $\{p_n\}_{n=1}^{\infty}$.

ISSN 1433-8092

In his seminal work [16], Valiant proved that the determinant polynomial is complete[2] for VP, and that permanent is complete for VNP. In particular, the VP versus VNP question reduces to a "cleaner" problem; deciding whether permanent can be efficiently represented as the projection of determinant.

Several approaches for solving this important problem have been suggested. Geometric complexity theory [10] suggests to use the symmetries of determinant and permanent via representation theory to separate VP from VNP. It was also suggested to use Newton polytopes (see [7] are references within), or to come up with elusive functions [11].

The Monotone Setting

The naive way to compute a polynomial is to write it as a sum of monomials. This type of computation is typically easy to understand but highly not efficient.

The monotone model suggests a more sophisticated way to compute a polynomial, without non trivial cancelations (for more background and motivation, see [12, 13, 14, 6]). In this model, the computation is over the non negative real numbers using addition and multiplication.

The monotone model is restricted in several ways. Obviously it only allows to compute monotone polynomials (with non negative coefficients). So, determinant for example is out of the question, but permanent, matrix multiplication and iterated convolution are viable objectives [6].

Even for monotone polynomials, it is not always the best way to perform computation. Quoting Shamir and Snir [14]: "Although monotone computations have several advantages, such as absolute numerical stability [8, 9], they are usually not practical for functions which can be computed much more efficiently using subtraction (or negation in the Boolean case)."

Nevertheless, there are non trivial monotone computations. For example, the permanent of an $n \times n$ matrix can be computed by a monotone circuit of size exponential in n, even though it has $n!$ monomials.

On the other hand, the monotone model helps to improve our understanding of computational problems. In it, we can prove (often sharp) lower bounds: Jerrum and Snir [6] proved for example that permanent requires monotone circuits of exponential size; Shamir and Snir [13] proved that multiplying d matrices of size $n \times n$ requires monotone formulas of size $n^{\Omega(\log d)}$; and Valiant [17] proved that one negation gate can be exponentially powerful. In addition, the monotone model helps to understand the limitations of reductions between computational devices. Hyafil [5] described a non trivial simulation of circuits by formulas, and the celebrated result of Valiant, Skyum, Berkowitz and Rackoff [18] shows how to

[2] For quasi-polynomial sized circuits.

simulate general small circuits by circuits that are both small and shallow. These simulations respect the monotone setting; namely, they simulate a monotone device by a monotone device. Now, the lower bound from [13] shows that Hyafil's simulation is optimal, as long as it respects monotonicity; and the authors of [4] showed that the simulation of Valiant et al. can not be made more efficient even if one is allowed to use algebraic branching programs, as long as it respects monotonicity.

Moving to the focus of this work, VP and VNP have natural monotone versions. MVP is the class of polynomials that can be computed by polynomial size monotone circuits. MVNP is the class of polynomials q that can be written as $q(x) = \sum_{b \in \{0,1\}^{\mathrm{poly}(n)}} p(x,b)$ where p is in MVP and n is the number of variables in q.

It seems appropriate to discuss the boolean analog of the MVP versus MVNP question. Consider a boolean function $q(x)$ that can be written as

$$q(x) = \exists b \in \{0,1\}^{\mathrm{poly}(n)} \; p(x,b)$$

where p is a function in P and n is the number of variables. This is the structure of a general NP language. Now, if we assume that p is a monotone boolean function then

$$q(x) = p(x, 1, 1, \ldots, 1).$$

Namely, q is actually in P. In this boolean version of the question, monotone P and NP are the same.

Lower Bounds and Equivalence

All lower bounds for monotone algebraic complexity we are aware of use the combinatorial structure of the monomials in the polynomial of interest. Here is a quote from [6]:

> Stated informally, once a monomial has been created, it must find its way into the final result; this "conservation of monomials" ensures that no "invalid" monomials are formed and severely limits the rate at which monomials may be accumulated in the computation.

The lower bound for permanent, for example, holds for every polynomial that has the same list of monomials as permanent. This naturally leads to the following definition. (Write $\alpha \in g$ if the coefficient of the monomial α in the polynomial g is non zero.)

Definition. *Two polynomials p, q are* equivalent *if the monomials that appear in both are the same. That is, $\alpha \in p$ iff $\alpha \in q$ for every monomial α.*

With this definition in mind, we make the following observation.

Observation. *If a monotone circuit-size lower bound for q holds also for all polynomials that are equivalent to q then it also holds for every monotone VNP circuit computing q.*

Reason. If $q(x) = \sum_b p(x, b)$ in the monotone setting, then $q(x)$ and $p(x, 1, 1, \dots, 1)$ are equivalent. □

In other words, all known monotone VP lower bounds hold for monotone VNP as well. Specifically, the proof from [6] implies

Observation. *The permanent requires monotone VNP circuits of size $2^{\Omega(n)}$.*

Stated differently, although there are many known lower bounds in the monotone setting, none of them separates MVP and MVNP. In particular, the two facts that (i) permanent is VNP complete and (ii) permanent requires exponentially large monotone circuits are somehow not relevant to the MVP versus MVNP question.

The discussion above shows that the MVP versus MVNP question can not be answered by looking at the list of monomials that appear in the polynomial of interest. We must consider the specific values of its coefficients. The separation question is more "analytic" than the lower bound question. To prove the separation, we must find a polynomial with a "simple" structure of monomials but a "complicated" structure of coefficients.

2 The Separation

Given an integer n, the separating polynomial is

$$P = P_n = 2^{-n} \sum_{b \in \{0,1\}^n} \prod_{i=1}^{n} \sum_{j=1}^{n} b_j x_{ij}$$

over the variables

$$X = X_n = \{x_{i,j} : i, j \in [n]\}.$$

The following theorem is our main result.

Theorem. *The polynomial P is in MVNP but not in MVP.*

The fact that P is in MVNP holds by definitions. Our lower bound proof shows that P requires monotone circuits of size at least $2^{\Omega(n/\log n)}$. The argument consists of two standard parts. The first part is about identifying a useful canonical form for monotone circuits, and in the second part we exploit the weakness revealed by the canonical form. The second part is often more technical and challenging.

The two lemmas below summarize the two parts of the proof. The first lemma (proved in Section 3) is similar to standard structural results for arithmetic circuits [13, 3, 15]. The second (proved in Section 4) summarizes the "exploiting the weakness" part.

Proof Overview

Notation. *We only consider polynomials over the set of variables X_n. We focus on monomials of the form $\alpha = \prod_{i \in I} x_{if(i)}$, where $I \subseteq [n]$ and $f : [n] \to [n]$. For such a monomial, let $I(\alpha) = I$, let $J(\alpha) = f(I)$ and let $j(\alpha) = |J(\alpha)|$. For a polynomial g, denote by $I(g)$ the union of $I(\alpha)$ over all $\alpha \in g$. Write $g \leq h$ if $g(\alpha) \leq h(\alpha)$ for all monomials α. Denote by $g(\alpha)$ the coefficient of the monomial α in the polynomial g, and let $\|g\|_1 = \sum_\alpha |g(\alpha)|$.*

The first lemma is based on the specific structure of P.

Definition. *We call g* ordered *if $I(\alpha) = I(g)$ for all $\alpha \in g$.*

The polynomial P is ordered:

$$\begin{aligned}
P &= 2^{-n} \sum_{b \in \{0,1\}^n} \prod_{i=1}^{n} \sum_{j=1}^{n} b_j x_{ij} \\
&= 2^{-n} \sum_{b} \sum_{f:[n]\to[n]} \prod_{i} b_{f(i)} x_{if(i)} \\
&= 2^{-n} \sum_{f} \prod_{i} x_{if(i)} 2^{n-|f([n])|} \\
&= \sum_{f} \alpha_f 2^{-j(\alpha_f)},
\end{aligned}$$

where $\alpha_f = \prod_i x_{i,f(i)}$. So, for every $\alpha \in P$, we have $I(P) = I(\alpha) = [n]$.

Lemma 1 (Structure)**.** *Let $n > 2$ and $q \in \mathbb{R}[X]$ be an ordered polynomial that can be computed by a monotone circuit of size s. Then, we can write q as*

$$q = \sum_{t=1}^{s} a_t b_t$$

where for each $t \in [s]$, the polynomials a_t, b_t are ordered so that $0 \leq a_t b_t \leq q$ with $n/3 \leq |I(a_t)| \leq 2n/3$ and $I(b_t) = [n] \setminus I(a_t)$.

The second lemma focuses on a single pair a_t, b_t, and analyzes the norm of a carefully chosen part of $a_t b_t$.

Lemma 2 (Weakness)**.** *There is a universal constant $c > 0$ so that the following holds. Let $n \geq 30$ and*

$$\delta = \frac{\lfloor \frac{n}{20+\log n} \rfloor}{n} > 0.$$

Let $a, b \in \mathbb{R}[X]$ be so that $0 \leq ab \leq p$ with $n/3 \leq |I(a)| \leq 2n/3$ and $I(b) = [n] \setminus I(a)$. Let π be the projection to the set of all monomials so that their j is exactly δn. Then,

$$\|\pi(ab)\|_1 \leq 2^{-cn/\log n} \|\pi(P)\|_1.$$

The theorem easily follows from the two lemmas: Assume that P can be computed by a monotone circuit of size s. By the structure lemma, write $P = \sum_{t=1}^{s} a_t b_t$. By the weakness lemma,

$$\|\pi(P)\|_1 \leq \sum_{t=1}^{s} \|\pi(a_t b_t)\|_1 \leq s 2^{-cn/\log n} \|\pi(P)\|_1.$$

Intuition for Weakness Lemma

Monomials in a correspond to maps from $I(a)$ to $[n]$, and in b to maps from $I(b)$ to $[n]$. Since $0 \leq ab \leq P$, for all monomials $\alpha \in a$ and $\beta \in b$,

$$0 \leq a(\alpha) b(\beta) \leq P(\alpha\beta) = 2^{-j(\alpha\beta)}. \tag{1}$$

Now, think of a two player game in which player a gets α as input, and player b gets β as input. Their goal is to output numbers $a(\alpha)$ and $b(\beta)$ so that (1) holds, without communication. Their mutual gain is $a(\alpha)b(\beta)$ if they succeed, so that they wish to maximize the numbers they choose.

The point is that a does not know β and b does not know α. So, it is reasonable to conjecture that almost always $a(\alpha)b(\beta)$ is actually much small than $2^{-j(\alpha\beta)}$; it should be something like $2^{-j(\alpha)-j(\beta)}$.

We are not able to prove such a strong statement, and there are some more choices to make and technical problems to overcome. The choice to focus on monomials so that their j is small (equals δn) has two reasons. One is that if $J(\alpha)$ and $J(\beta)$ are typical sets of size at most δn for small δ, then $j(\alpha\beta)$ is close to $j(\alpha) + j(\beta)$. Namely, the smaller δ is, the more likely the intuition above can be made formal. Another is that for $\delta < \frac{1}{1+\log n}$ we get simple yet useful estimates on $\|\pi(p)\|_1$.

As a final remark, we note that although the lemma is about $\pi(ab)$ and $\pi(P)$, the proof uses monomials outside the image of π. This seems to be a necessity, and not just an artifact of the proof; the polynomial $q = \eta \cdot \prod_i \sum_j x_{i,j}$ can be written as a single product $q = ab$ with $n/3 \leq |I(a)| \leq 2n/3$ and $I(b) = [n] \setminus I(a)$, and it satisfies $\pi(q) = \pi(P)$ for the appropriate $\eta > 0$.

3 Proof of Structure Lemma

Consider a monotone circuit of size s for q. Assume that there are no gates that are not connected to the output gate, and no gate that computes the zero polynomial. For each gate v in it, let $I(v) = I(a_v)$ where a_v is the polynomial computed at v. Monotonicity implies that each a_v is ordered, since if some a_v is not ordered then the output gate is not ordered as well. In particular, if $v = v_1 + v_2$ then

$$I(v) = I(v_1) = I(v_2),$$

and if $v = v_1 \times v_2$ then

$$I(v) = I(v_1) \cup I(v_2) \text{ and } I(v_1) \cap I(v_2) = \emptyset.$$

Going from output to inputs, let v be a first gate so that $I(v) \leq 2n/3$. Thus, $n/3 \leq |I(v)| \leq 2n/3$. There is a polynomial $b_v \geq 0$ so that

$$q = a_v b_v + r_v,$$

where r_v has a monotone circuit of size at most $s - 1$. Since $a_v b_v \leq q$ and q is ordered, the polynomial b_v is ordered and $I(b_v) = [n] \setminus I(a_v)$. So, if $r_v = 0$ we are done, and if $r_v \neq 0$ we can apply induction.

4 Proof of Weakness Lemma

Start with

$$\|\pi(p)\|_1 = \sum_{S \subset [n] : |S| = \delta n} \sum_{f : f([n]) = S} 2^{-|f([n])|} = 2^{-\delta n} \binom{n}{\delta n} F_{n,\delta n},$$

where $F_{n,k}$ is the number of onto maps from $[n]$ to $[k]$.

Claim 3. $\frac{1}{2}(\delta n)^n \leq F_{n,\delta n} \leq (\delta n)^n$.

Proof. The right inequality is clear. The left inequality:

$$\begin{aligned} (\delta n)^n - F_{n,\delta n} &\leq \delta n (\delta n - 1)^n && \text{(union bound)} \\ &\leq \delta n e^{-\frac{n}{\delta n}} \cdot (\delta n)^n && (1 - \xi \leq e^{-\xi}) \\ &\leq \tfrac{1}{2}(\delta n)^n. && (\tfrac{1}{\delta} \geq 1 + \log n) \end{aligned}$$

□

The upper bound on $\|\pi(ab)\|_1$ is partitioned to two cases as follows. Let

$$\gamma = \tfrac{1}{20}.$$

Let π' be the projection to the set of monomials with j in $[1-\gamma, 1]\delta n$. We can assume without loss of generality that

$$\|\pi'(a)\|_\infty = \|\pi'(b)\|_\infty. \tag{2}$$

Indeed, setting

$$y = \sqrt{\tfrac{\|\pi'(a)\|_\infty}{\|\pi'(b)\|_\infty}} > 0$$

(if $\pi'(b) = 0$ or $\pi'(a) = 0$ then we are done; see case one below), we get

$$\|\tfrac{\pi'(a)}{y}\|_\infty = \|y \cdot \pi'(b)\|_\infty = \sqrt{\|\pi'(a)\|_\infty \|\pi'(b)\|_\infty}.$$

Case one (easier): $\|\pi'(a)\|_\infty < 2^{-(1-\gamma)\delta n}$

For all $\alpha \in a$ so that $j(\alpha) \in [1-\gamma, 1]\delta n$,

$$0 \le a(\alpha) < 2^{-(1-\gamma)\delta n}.$$

A similar property holds for b, by (2).

We wish to upper bound

$$\|\pi(ab)\|_1 = \sum_{S:|S|=\delta n} \sum_{\alpha,\beta:J(\alpha\beta)=S} a(\alpha)b(\beta).$$

Fix S for now. The sum over α so that $j(\alpha) < (1-\gamma)\delta n$ and all β is at most

$$\begin{aligned}
&\binom{\delta n}{< (1-\gamma)\delta n}((1-\gamma)\delta n)^{|I(a)|}(\delta n)^{|I(b)|}2^{-\delta n} && \text{(by (1))}\\
&\le 2\binom{\delta n}{< (1-\gamma)\delta n}(1-\gamma)^{n/3} \cdot 2^{-\delta n} F_{n,\delta n} && \text{(Claim 3 \& } |I(a)| \ge n/3)\\
&\le 2^{-\Omega(n)} 2^{-\delta n} F_{n,\delta n}.
\end{aligned}$$

Similarly, we can upper bound the sum over β so that $j(\beta)$ is small. So, we are left with the sum over α, β so that their j is at least $(1-\gamma)\delta n$. This sum is at most

$$(\delta n)^n 2^{-2(1-\gamma)\delta n} \le 2^{-\Omega(\delta n)} 2^{-\delta n} F_{n,\delta n}.$$

Finally, we sum over S:

$$\|\pi(ab)\|_1 \le 3 \cdot 2^{-\Omega(\delta n)} \binom{n}{\delta n} 2^{-\delta n} F_{n,\delta n} \le 2^{-\Omega(n/\log n)} \|\pi(P)\|_1.$$

So the proof in case one is complete.

Case two (harder): $\|\pi'(a)\|_\infty \geq 2^{-(1-\gamma)\delta n}$

There is a monomial $\alpha_0 \in a$ so that

$$j(\alpha_0) \in [1-\gamma, 1]\delta n$$

and

$$a(\alpha_0) \geq 2^{-(1-\gamma)\delta n}.$$

There is a similar $\beta_0 \in b$.

Partition the sum over S to two parts according to

$$\mathcal{S} = \big\{S \subset [n] : |S| = \delta n,\ |S \setminus (J(\alpha_0) \cup J(\beta_0))| < (1 - 2\delta - \gamma)\delta n\big\}.$$

The family $\mathcal{S}$ is small:

Claim 4. $\frac{|\mathcal{S}|}{\binom{n}{\delta n}} \leq 2^{-\Omega(\delta n)}$.

Proof. Let T be a random set in $[n]$ so that each i is in T with probability δ independently of other i's. The size of $Q = [n] \setminus (J(\alpha_0) \cup J(\beta_0))$ is at least $(1-2\delta)n$. The expectation of

$$Y = \tfrac{|T \cap Q|}{|Q|}$$

is $\mathbb{E}\, Y = \delta$. By the Chernoff-Hoeffding inequality,

$$\Pr\left[Y < \mathbb{E}\, Y - \gamma\delta\right] \leq e^{-D((1-\gamma)\delta || \delta)|Q|} \leq e^{-\frac{(\delta - (1-\gamma)\delta)^2}{2\delta} n(1-2\delta)} \leq e^{-\Omega(\delta n)}.$$

We now need to move from T to S. The uniform distribution on $\binom{[n]}{\delta n}$ is that of T conditioned on $|T| = \delta n$. Since the mode of $|T|$ is δn, we have $\Pr[|T| = \delta n] \geq \frac{1}{n}$. If

$$|T \setminus (J(\alpha_0) \cup J(\beta_0))| < (1 - 2\delta - \gamma)\delta n$$

then

$$Y < \frac{(1-2\delta-\gamma)\delta n}{|Q|} \leq \frac{1 - 2\delta - \gamma}{1 - 2\delta}\delta \leq \mathbb{E}\, Y - \gamma\delta.$$

So, finally

$$\frac{|\mathcal{S}|}{\binom{n}{\delta n}} \leq \Pr\left[Y < \mathbb{E}\, Y - \gamma\delta \;\middle|\; |T| = \delta n\right] \leq n \Pr\left[Y < \mathbb{E}\, Y - \gamma\delta\right] = 2^{-\Omega(\delta n)}.$$

$\square$

So, the part of the sum over $\mathcal{S}$ is at most

$$\sum_{S\in\mathcal{S}}\sum_{\alpha,\beta:J(\alpha\beta)=S} 2^{-\delta n} \le |\mathcal{S}|2^{-\delta n}(\delta n)^{|I(a)|+|I(b)|} \le 2^{-\Omega(\delta n)}\|\pi(p)\|_1. \tag{3}$$

It remains to bound the part of the sum over $\overline{\mathcal{S}}$. Fix $S \notin \mathcal{S}$, and consider

$$\sum_{\alpha,\beta:J(\alpha\beta)=S} a(\alpha)b(\beta).$$

As in case one, the sum over α so that $j(\alpha) < (1-\gamma)\delta n$ is at most $2^{-\Omega(n)}2^{-\delta n}F_{n,\delta n}$. We can similarly bound the sum over β so that $j(\beta)$ is small. So, we are left with the sum over α, β so that both of their j value is at least $(1-\gamma)\delta n$. In fact, we are left with the sum over α, β so that their J is contained in S and is at least of size $(1-\gamma)\delta n$.

By (1), for all $\beta \in b$,

$$2^{-(1-\gamma)\delta n} b(\beta) \le a(\alpha_0)b(\beta) \le 2^{-j(\alpha_0\beta)}$$

or

$$b(\beta) \le 2^{-j(\alpha_0\beta)+(1-\gamma)\delta n}.$$

For each β we need to sum over, bound

$$\begin{aligned} j(\alpha_0\beta) &\ge j(\alpha_0) + |J(\beta) \setminus J(\alpha_0)| \\ &\ge (1-\gamma)\delta n + |S \setminus J(\alpha_0)| - |S \setminus J(\beta)| \\ &\ge (1-\gamma)\delta n + (1-2\delta-\gamma)\delta n - \gamma\delta n. && (S \notin \mathcal{S} \ \&\ J(\beta) \text{ is large}) \end{aligned}$$

Thus,

$$\begin{aligned} b(\beta) &\le 2^{-(1-\gamma)\delta n-(1-2\delta-\gamma)\delta n+\gamma\delta n+(1-\gamma)\delta n} \\ &\le 2^{-\delta n(1-2(\gamma+\delta))} \\ &\le 2^{-\frac{2}{3}\delta n}. && (\gamma+\delta \le \tfrac{1}{10}) \end{aligned}$$

A similar bound holds for the each α we need to sum over. So, the sum over such α, β is at most

$$2^{-2\cdot\frac{2}{3}\delta n}(\delta n)^n \le 2^{-\Omega(n/\log n)}2^{-\delta n}F_{n,\delta n}.$$

Finally, we sum over all $S \notin \mathcal{S}$:

$$\sum_{S\notin\mathcal{S}}\sum_{\alpha,\beta:J(\alpha\beta)=S} a(\alpha)b(\beta) \le 2^{-\Omega(n/\log n)}\|\pi(P)\|_1.$$

Together with (3) the proof of the weakness lemma is complete.

Acknowledgement

I thank Pavel Hrubeš and Avi Wigderson for their contribution to this work.

References

[1] P. Bürgisser, M. Clausen, and M. A. Shokrollahi. *Algebraic complexity theory*, volume 315. Springer Science and Business Media, 1997.

[2] J. von zur Gathen. Algebraic complexity theory. *Annual Review of Computer Science* 3, pages 317–347, 1988.

[3] P. Hrubeš and A. Yehudayoff. Monotone separations for constant degree polynomials. *Information Processing Letters* 110 (1), pages 1–3, 2009.

[4] P. Hrubeš and A. Yehudayoff. On isoperimetric profiles and computational complexity. *LIPIcs-Leibniz International Proceedings in Informatics*, 55, 2016.

[5] L. Hyafil. On the parallel evaluation of multivariate polynomials. *SICOMP* 8(2):120–123, 1979.

[6] M. Jerrum and M. Snir. Some exact complexity results for straight-line computations over semirings. *J. ACM* 29 (3), pages 874–897, 1982.

[7] P. Koiran, N. Portier, S. Tavenas, and S. Thomassé. A τ-Conjecture for Newton Polygons. *Foundations of Computational Mathematics* 15 (1), pages 185–197, 2015.

[8] W. Miller. Computational complexity and numerical stability. *SlCOMP* 4, pages 97–107, 1975.

[9] W. Miller. Computer search for numerical stability. *J. ACM* 22, pages 512–521, 1975.

[10] K. D. Mulmuley, and M. Sohoni. Geometric complexity theory I: An approach to the P vs. NP and related problems. *SIAM Journal on Computing* 31 (2), pages 496–526, 2001.

[11] R. Raz. Elusive functions and lower bounds for arithmetic circuits. In *STOC*, pages 711–720, 2008.

[12] C.P. Schnorr. A lower bound on the number of additions in monotone computations. *Theoretical Computer Science* 2 (3), pages 305–315, 1976.

[13] E. Shamir and M. Snir. Lower bounds on the number of multiplications and the number of additions in monotone computations. *Technical Report* RC-6757, IBM, 1977.

[14] E. Shamir and M. Snir. On the depth complexity of formulas. *Journal Theory of Computing Systems* 13 (1), pages 301–322, 1979.

[15] A. Shpilka and A. Yehudayoff. Arithmetic circuits: A survey of recent results and open questions. *Found. Trends Theor. Comput. Sci.* 5, pages 207–388, 2010.

[16] L. G. Valiant. The complexity of computing the permanent. *Theoretical Computer Science* 8 (2), pages 189–201, 1979.

[17] L. G. Valiant. Negation can be exponentially powerful. *Theoretical Computer Science* 12, pages 303–314, 1980.

[18] L. G. Valiant, S. Skyum, S. Berkowitz, and C. Rackoff. Fast parallel computation of polynomials using few processors. *SIAM J. on Computing*, 12 (4), pages 641–644, 1983.

ECCC ISSN 1433-8092
https://eccc.weizmann.ac.il