Algebraic Pseudorandomness in VNC^0

Robert Andrews
Cheriton School of Computer Science, University of Waterloo, Canada

Abstract

We study the arithmetic complexity of hitting set generators, which are pseudorandom objects used for derandomization of the polynomial identity testing problem. We give new explicit constructions of hitting set generators whose outputs are computable in VNC^0, i.e., can be computed by arithmetic formulas of constant size. Unconditionally, we construct a VNC^0-computable generator that hits arithmetic circuits of constant depth and polynomial size. We also give conditional constructions, under strong but plausible hardness assumptions, of VNC^0-computable generators that hit arithmetic formulas and arithmetic branching programs of polynomial size, respectively. As a corollary of our constructions, we derive lower bounds for subsystems of the Geometric Ideal Proof System of Grochow and Pitassi.

Constructions of such generators are implicit in prior work of Kayal on lower bounds for the degree of annihilating polynomials. Our main contribution is a construction whose correctness relies on circuit complexity lower bounds rather than degree lower bounds.

2012 ACM Subject Classification Theory of computation → Algebraic complexity theory; Theory of computation → Pseudorandomness and derandomization; Theory of computation → Proof complexity

Keywords and phrases Polynomial identity testing, Algebraic circuits, Ideal Proof System

Digital Object Identifier 10.4230/LIPIcs.CCC.2025.15

Related Version *Full Version*: `https://arxiv.org/abs/2505.10675` [3]

Funding *Robert Andrews*: Part of this work was done at the Institute for Advanced Study and was supported by NSF grant CCF-1900460 and the Erik Ellentuck Endowed Fellowship Fund.

Acknowledgements We thank Ramprasad Saptharishi for a simpler formulation and proof of [3, Lemma 3.5], Rafael Oliveira for helpful conversations on Nullstellensatz degree bounds, and Abhibhav Garg for useful feedback that improved the presentation of this work.

1 Introduction

1.1 Polynomial Identity Testing

Algebraic complexity is a vibrant subarea of complexity theory that studies computation of polynomial and rational functions using basic arithmetic operations. Like boolean complexity theory, algebraic complexity enjoys a rich theory of pseudorandomness, with the polynomial identity testing (PIT) problem playing a central role. The input to the PIT problem is an arithmetic circuit, and the goal is to decide whether the circuit computes the identically zero polynomial. This problem can be efficiently solved with randomness using the Schwartz–Zippel lemma [43, 47], which says that if a degree-d polynomial f is nonzero, then with probability at least $1/2$ a randomly-chosen point from a grid of side-length $2d$ will lead to a nonzero evaluation of f. A great deal of work has gone into designing efficient deterministic algorithms for polynomial identity testing, leading to beautiful constructions and connections to other areas of computer science and mathematics.

Algorithms for PIT are often designed by constructing *hitting set generators*, which play a role analogous to pseudorandom generators in boolean complexity. For a complexity class $\mathscr{C}$, a hitting set generator $\mathcal{G}$ for $\mathscr{C}$ is a (family of) polynomial map(s) $\mathcal{G} : \mathbb{F}^\ell \to \mathbb{F}^n$ such

that for every polynomial $f \in \mathscr{C}$, we have $f = 0$ if and only if $f \circ \mathcal{G} = 0$.[1] Conceptually, one can think of a generator as mapping ℓ truly random field elements to n pseudorandom field elements. If the generator $\mathcal{G}$ can be implemented by an efficient algorithm, then we are led to an improved deterministic PIT algorithm for $\mathscr{C}$-circuits: given a circuit C, test the composition $C \circ \mathcal{G}$ by brute force. The running time of the naïve brute-force algorithm for PIT has an exponential dependence on the number of variables, and the generator $\mathcal{G}$ improves this exponent from n to the smaller ℓ, a parameter referred to as the *seed length* of the generator. The usual aim is to construct generators with seed length that is as small as possible, since this is the most important parameter for improving the running time of the deterministic algorithm.

Numerous constructions of hitting set generators are known, both conditional and unconditional. In this work, we will be interested in designing hitting set generators for strong circuit classes, at or beyond the frontier of our ability to prove super-polynomial lower bounds for explicit polynomials.[2] Starting with Kabanets and Impagliazzo [29], who adapted the Nisan–Wigderson [37] generator to the algebraic setting, there has been a successful line of work constructing hitting set generators for strong circuit classes – including general, unrestricted circuits – under hardness assumptions [14, 12, 2, 44, 24]. In fact, some of the lower bounds assumed by these works have since been proven unconditionally, leading to new deterministic algorithms for PIT! Specifically, the super-polynomial lower bounds of Limaye, Srinivasan, and Tavenas [36] against constant-depth arithmetic circuits, combined with the hardness-to-randomness theorem of Chou, Kumar, and Solomon [12], led to the first deterministic sub-exponential time algorithm to test polynomial identities written as constant-depth circuits.

Recently, Chatterjee and Tengse [11] raised a very interesting question about the complexity of hitting set generators. They asked if it is possible to construct a hitting set generator that is computable in some small complexity class $\mathscr{C}$, yet appears pseudorandom to a larger complexity class $\mathscr{D} \supsetneq \mathscr{C}$. This sort of parameter regime is common throughout cryptography, so they termed such a generator a *cryptographic* hitting set generator. The question of constructing cryptographic hitting set generators is an extremely interesting one, and is the focus of our work. Aside from this question's inherent interest, one might expect the techniques underlying the construction of a cryptographic hitting set generator to have other applications within algebraic complexity theory. For example, the analogous question of constructing low-complexity pseudorandom generators was answered in a beautiful work of Applebaum, Ishai, and Kushilevitz [5], and the techniques therein have found numerous applications within cryptography and complexity, such as to the recent study of the range avoidance problem [39].

Most known hardness-randomness connections in algebraic complexity cannot hope to produce a hitting set generator that operates in this cryptographic regime of parameters. Almost all generators that come from algebraic hardness-randomness are *reconstructive*, meaning that their correctness proofs follow a common template. In a reconstructive proof of correctness, the generator $\mathcal{G}$ is proven correct by an argument of the following form: suppose some explicit polynomial f, such as the $n \times n$ permanent, is hard to compute. Given a

[1] Formally, we consider *families* of polynomials $(f_1, f_2, \ldots)$ and families of polynomial maps $(\mathcal{G}_1, \mathcal{G}_2, \ldots)$, rather than a single polynomial f and a single map $\mathcal{G}$. We will gloss over this distinction throughout the introduction for the sake of readability, focusing instead on single elements f_n and $\mathcal{G}_n$ of the respective families.

[2] For results on polynomial identity testing for weaker circuit classes, we refer the reader to the recent survey of Dutta and Ghosh [13], as well as the surveys of Saxena [41, 42].

nonzero circuit C of size s that satisfies $C \circ \mathcal{G} = 0$, there is a reconstruction procedure that modifies C into a circuit C' of size $s + t$ that computes the supposedly-hard polynomial f. If $s + t \ll \text{size}(f)$, where $\text{size}(f)$ is the complexity of f, then we arrive at a contradiction. The upshot of this argument is that if the circuit C is sufficiently small – in particular, when the circuit C is of size $s \ll \text{size}(f)$ – the hardness of f implies that the composition $C \circ \mathcal{G}$ must be nonzero, i.e., that $\mathcal{G}$ hits small circuits. The bottleneck in this argument is that the generator $\mathcal{G}$ is usually defined using the hard polynomial f (for example, $\mathcal{G}$ may be the Nisan–Wigderson generator applied to f), so the complexity of $\mathcal{G}$ is necessarily bounded from below by $\text{size}(f)$. Because the reconstruction argument only proves that $\mathcal{G}$ is pseudorandom against circuits of size $s \ll \text{size}(f) \leqslant \text{size}(\mathcal{G})$, this proof template is doomed to fail in constructing a cryptographic hitting set generator. To design cryptographic hitting set generators for strong circuit classes, we need to avoid this reconstructive approach.

One instance where the overhead from reconstruction can be avoided is found in the work of Andrews and Forbes [4]. They gave a subexponential-time algorithm to test polynomial identities that are written as constant-depth circuits. The same algorithmic result was already obtained by the previously-mentioned work of Limaye, Srinivasan, and Tavenas [36]. However, these two algorithms differ in the complexity of the hitting set generator underlying the algorithm: the generator of [36] is reconstructive, whereas the generator of [4] provably has smaller complexity than the circuits it hits. Unlike most works in algebraic hardness-randomness, Andrews and Forbes [4] do not design a generator that is based on evaluations of hard functions, but rather design the generator so that its annihilator ideal consists only of hard polynomials. For a generator $\mathcal{G}$, its *annihilator ideal*, denoted by $\text{Ann}(\mathcal{G})$, is the set of all polynomials f such that $f \circ \mathcal{G} = 0$. In principle, one could show that a generator hits a circuit class $\mathscr{C}$ by showing that all nonzero polynomials in the annihilator ideal are so complex that they lie outside the circuit class $\mathscr{C}$. Proving a statement like this is necessary for the proof of correctness of a generator, but these statements are usually not the core thrust of the argument and only arise as a byproduct of the correctness proof. As we will see, adopting this viewpoint will be useful for constructing further examples of generators that hit circuits more complex than the generator itself.

1.2 Cryptographic Generators from Degree Bounds for Annihilating Polynomials

Although the question of constructing cryptographic hitting set generators in algebraic complexity is a recent one, prior work on degree bounds for annihilating polynomials leads, at least implicitly, to constructions of cryptographic generators. Kayal [31, Theorem 12] showed that over a field of characteristic zero, any annihilator of the $n + 1$ polynomials

$$\begin{aligned} g_1(\overline{x}) &= x_1^d - 1 \\ &\vdots \\ g_n(\overline{x}) &= x_n^d - 1 \\ g_{n+1}(\overline{x}) &= x_1 + x_2 + \cdots + x_n - n \end{aligned}$$

must have degree at least d^n. Equivalently, the corresponding generator $\mathcal{G} : \mathbb{F}^n \to \mathbb{F}^{n+1}$ given by $\mathcal{G}(\overline{x}) = (g_1(\overline{x}), \ldots, g_{n+1}(\overline{x}))$ hits all polynomials of degree less than d^n. Because many restricted forms of arithmetic circuits, such as arithmetic formulas and branching programs, necessarily compute low-degree polynomials, this generator hits powerful classes of circuits. For example, size-s arithmetic branching programs cannot compute polynomials of degree

greater than s, so this generator hits *all* branching programs of size less than d^n. This is a bit unusual. The best-known lower bounds against arithmetic branching programs are only quadratic [10], so intuition from the hardness versus randomness paradigm suggests we should only expect to have constructions of generators that hit branching programs of size up to $O(n^2)$.

The preceding example of Kayal [31] fits into the cryptographic regime, as the outputs are extremely simple to compute. Taking $d = 2$, we obtain a generator where the first n outputs can be computed by formulas of constant size, the last output can be computed by a formula of size n and depth two, and yet the generator hits all branching programs of size less than 2^n. Although this generator only stretches its input by one field element in length, it is possible to improve the stretch of the generator by invoking $n^{1-\varepsilon}$ copies in parallel, each on a fresh set of n^ε variables. This results in a generator $\mathcal{G}' : \mathbb{F}^n \to \mathbb{F}^{n+n^{1-\varepsilon}}$ that stretches its input by $n^{1-\varepsilon}$ field elements. Kayal's construction can also be modified to obtain a generator with similar parameters where *all* outputs are computable by constant-size formulas. To do this, we encode the final output g_{n+1} with additional variables $y_2, \ldots, y_{n-1}$ via

$$\begin{aligned} h_{n+1}(\overline{x}, \overline{y}) &= x_1 + x_2 - y_2 \\ h_{n+2}(\overline{x}, \overline{y}) &= y_2 + x_3 - y_3 \\ &\vdots \\ h_{2n-1}(\overline{x}, \overline{y}) &= y_{n-1} + x_n - n. \end{aligned}$$

Kayal's lower bound on the degree of the annihilator extends to this variation on his example, so we already have constructions of generators where each output is computable by a formula of constant size, yet the generator itself hits polynomials of much higher complexity.

1.3 The Ideal Proof System

Not only is the problem of constructing cryptographic hitting set generators interesting in its own right, but it is also closely tied to open problems in algebraic proof complexity. The central goal of propositional proof complexity is to understand the NP versus coNP problem via the complexity of the coNP-complete unsatisfiability problem: given a boolean formula φ, accept φ if and only if φ is unsatisfiable. The typical setting is to first fix a proof system, and then try to find families of boolean formulas $(\varphi_n)_{n\in\mathbb{N}}$ such that any proof π_n of the unsatisfiability of φ_n requires length that is super-polynomial in n. Numerous proof systems based on different areas of mathematics, including logic, algebra, and geometry, have been considered.

Most relevant to our work are proof systems based on algebraic reasoning. Without loss of generality, we may assume that our unsatisfiable boolean formula φ is in 3CNF form. If φ is an n-variate 3CNF formula with m clauses, there is a translation of φ to a system $\mathcal{F}$ of $n + m$ polynomial equations such that $\mathcal{F}$ is satisfiable if and only if φ is satisfiable. This system $\mathcal{F}$ consists of the n boolean axioms $x_i^2 - x_i = 0$, which force the variables x_i to be $\{0, 1\}$-valued, and for each clause of the form $(x_1 \oplus b_1) \vee (x_2 \oplus b_2) \vee (x_3 \oplus b_3)$, the trivariate equation

$$(x_1 - (1 - b_1))\,(x_2 - (1 - b_2))\,(x_3 - (1 - b_3)) = 0.$$

It is easy to see that any boolean assignment to the x variables satisfies this equation if and only if the same assignment satisfies the corresponding clause $(x_1 \oplus b_1) \vee (x_2 \oplus b_2) \vee (x_3 \oplus b_3)$. Thus, if we want to prove that φ is unsatisfiable, we can instead try to prove the unsatisfiability of the resulting system of polynomial equations $\mathcal{F}$.

How does one prove that a system of polynomial equations is unsatisfiable? The answer comes from Hilbert's Nullstellensatz. Over an algebraically closed field $\mathbb{F}$, the system of equations

$$f_1(\overline{x}) = \cdots = f_m(\overline{x}) = 0$$

is unsatisfiable if and only if there are polynomials $g_1, \ldots, g_m \in \mathbb{F}[x_1, \ldots, x_n]$ such that

$$\sum_{i=1}^{m} f_i(\overline{x})\, g_i(\overline{x}) = 1.$$

We can take the polynomials $\{g_1, \ldots, g_m\}$ to be our proof of unsatisfiability. Our goal, then, is to understand the complexity of the simplest refutation $\{g_1, \ldots, g_m\}$ of the system of equations $\mathcal{F}$.

The complexity of the proof depends on the choice of proof system. The Nullstellensatz proof system [6, 8] measures the length of the proof by the total number of monomials in the g_i. A slightly stronger system is the Polynomial Calculus [38, 27], where we are allowed to write the derivation of 1 from the f_i in a line-by-line manner, but we still pay for the number of monomials written on every line. Much stronger is the Ideal Proof System (IPS), introduced by Grochow and Pitassi [23], which allows us to write the g_i succinctly as arithmetic circuits. Given that refutations in IPS are written as arithmetic circuits, one would hope that techniques for proving arithmetic circuit lower bounds will eventually lead to lower bounds for the IPS.

This hope has played out successfully in recent years. Forbes, Shpilka, Tzameret, and Wigderson [18] introduced two techniques to infer IPS lower bounds from arithmetic circuit lower bounds, and applied these techniques for restricted circuit classes to conclude unconditional lower bounds. The first technique, known as the functional lower bound method, has been further refined and studied by Govindasamy, Hakoniemi, and Tzameret [20] and Hakoniemi, Limaye, and Tzameret [25]. Here, the hard system of equations $\mathcal{F}$ typically consists of boolean axioms together with a sparse polynomial that corresponds to an instance of subset sum, possibly lifted with an appropriate gadget.[3] This method is capable of proving strong lower bounds; for example, Hakoniemi, Limaye, and Tzameret [25] proved super-polynomial lower bounds on the size of constant-depth IPS refutations for such systems of equations. However, current applications of the functional lower bound method are only able to prove lower bounds against refutations of individual degree $O(\log \log n)$. This is to be expected, as functional lower bounds without the individual degree constraint, even for depth-four arithmetic circuits, would lead to new lower bounds in boolean complexity theory [17].

The second technique introduced by Forbes, Shpilka, Tzameret, and Wigderson [18] is based on hardness of multiples. This method is also capable of proving strong lower bounds: later work by Andrews and Forbes [4] proved super-polynomial lower bounds on the size of constant-depth IPS refutations for a system of equations related to matrix rank. Unlike the functional lower bound method, the lower bounds produced from hardness of multiples do not require the IPS refutation to be of small individual degree. However, the drawback to this method is that in order to prove a lower bound against IPS certificates computed by $\mathscr{C}$-circuits, there must be at least one polynomial f in the hard instance $\mathcal{F}$ which itself

[3] These hard instances are closely related to the previously-discussed example of Kayal [31] that leads to lower bounds on the degree of annihilating polynomials.

cannot be computed efficiently by $\mathscr{C}$-circuits. This limitation is inherent to the method, as the lower bound against IPS is derived from circuit lower bounds for one (or more) of the polynomials in the hard instance $\mathcal{F}$.

Strong conditional lower bounds for IPS are also known. Alekseev, Grigoriev, Hirsch, and Tzameret [1] proved super-polynomial lower bounds on the size of constant-free IPS refutations of the binary value principle, a system of equations that asserts a natural number n given in binary is negative. Their lower bound assumes the τ-conjecture of Shub and Smale, which asserts that the straight-line program complexity of computing (a multiple of) the integer $n!$ is bounded from below by $\log^{\omega(1)} n$. Santhanam and Tzameret [40] showed that under the assumption VP $\neq$ VNP, there is a sequence of 3CNF formulas that require IPS refutations of super-polynomial size. One drawback of this result is that the sequence of CNF formulas considered by Santhanam and Tzameret [40] may be satisfiable; if this were the case, then the lower bound becomes trivial, as the soundness of IPS implies that it cannot refute a system of satisfiable equations.

Despite the success so far in bringing techniques from arithmetic circuit complexity to bear on IPS, we still seem far from proving unconditional lower bounds for systems of polynomial equations that encode unsatisfiable 3CNF formulas, even against weak subsystems of IPS. Current applications of the functional lower bound method are limited to proving lower bounds against refutations of low individual degree, and some formulations of the method provably cannot lead to lower bounds for boolean instances [25]. The method of lower bounds for multiples can prove unconditional lower bounds against fragments of IPS with no restrictions on the individual degree of the refutation, but the method requires the hard instance to contain polynomials of large circuit complexity, which precludes its application to systems of equations that encode a 3CNF formula. As a step towards proving IPS lower bounds for such systems, can we prove IPS lower bounds for any system of equations where each polynomial in the system can be described by an arithmetic formula of constant size?

1.4 Hard Families of Simple Polynomials from Nullstellensatz Degree Bounds

Just as prior work on annihilating polynomials led to simple constructions of cryptographic hitting set generators, existing work on lower bounds for Nullstellensatz degree – in particular, examples demonstrating the tightness of these bounds – easily leads to families of simple polynomials that are hard for fragments of IPS. Recall that the Nullstellensatz says that if $f_1 = \cdots = f_m = 0$ is an unsatisfiable system of equations, then there are polynomials $g_1, \ldots, g_m$ such that $\sum_{i=1}^{m} f_i g_i = 1$. Of particular relevance to our work are degree bounds for the Nullstellensatz: given the polynomials f_i, what is the smallest possible degree of the polynomials g_i that witness the identity $\sum_{i=1}^{n} f_i g_i = 1$? A long line of work [26, 7, 9, 32, 15, 45, 33, 28] has established various bounds on the degrees and heights of such polynomials. For example, we know that such polynomials g_i can always be found with degree $\deg(g_i) \leqslant d^n$, where $d = \max_i \deg(f_i)$ is the maximum degree of the f_i. An example due to Masser and Philippon (see [7]) shows that bounds of this shape are tight. In particular, if we take

$$\begin{aligned} f_1(\overline{x}) &= x_1^d \\ f_2(\overline{x}) &= x_1 - x_2^d \\ &\vdots \\ f_{n-1}(\overline{x}) &= x_{n-2} - x_{n-1}^d \\ f_n(\overline{x}) &= 1 - x_{n-1} x_n^{d-1}, \end{aligned}$$

then in any expression $\sum_{i=1}^{n} f_i g_i = 1$, the polynomial g_1 must satisfy $\deg(g_1) \geqslant d^n - d^{n-1}$. This degree lower bound implies similar lower bounds on the size of IPS refutations of the above system when the refutation is written as an arithmetic formula or arithmetic branching program. The lower bound follows from the fact that an arithmetic formula or branching program of size s can only compute polynomials of degree at most s, so no formula or branching program of size less than $d^n - d^{n-1}$ can compute a refutation of this system.

At first glance, it appears that we have made progress towards IPS lower bounds for refuting 3CNF formulas. The example of Masser and Philippon above gives us a system of polynomial equations, each of which can be encoded by a constant-size arithmetic formula, that requires large refutations in powerful subsystems of IPS. Unfortunately, this line of reasoning cannot lead to IPS lower bounds for systems of polynomials that encode 3CNF formulas. An unsatisfiable 3CNF formula on n variables always admits a degree-$O(n)$ refutation [23], so degree bounds will not result in new lower bounds on the complexity of refuting 3CNF formulas. To make progress towards proving lower bounds for IPS refutations of 3CNF formulas, we need techniques rooted in finer complexity measures than degree.

1.5 Our Contributions

We now describe our results. As we have seen in Sections 1.2 and 1.4, prior work on degree bounds can be used to construct cryptographic hitting set generators and hard instances for the Ideal Proof System. The main contribution of our work is a new construction of cryptographic hitting set generators whose correctness is based on arithmetic circuit lower bounds, not degree lower bounds. As a corollary, we will obtain families of simple polynomials that are hard to refute in subsystems of the the Geometric Ideal Proof System, itself a restricted form of the Ideal Proof System [23, Appendix B]. Although the theorem statements below already follow from prior work, the constructions appearing in our proofs do not, and we believe there is value in developing techniques in algebraic pseudorandomness and proof complexity that go beyond degree lower bounds.

Our first result is an explicit construction of a low-complexity hitting set generator that hits VAC^0, the class of polynomials computed by constant-depth circuits of polynomial size. Each output of our generator is computable in VNC^0, i.e., can be computed by an arithmetic formula of constant size.

▶ **Theorem 1.1.** *Let $\mathbb{F}$ be a field of characteristic zero.*[4] *There is a VNC^0-computable hitting set generator with stretch $n^{0.99}$ that hits VAC^0.*

Under strong but reasonable hardness assumptions, the same techniques yield VNC^0-computable hitting set generators that hit larger circuit classes. Our first conditional construction yields a VNC^0-computable generator that hits VF, which corresponds to families of polynomials that are computable by polynomial-size formulas. The correctness of this construction relies on the assumption that the border formula complexity of the determinant is super-polynomial.

▶ **Theorem 1.2.** *Let $\mathbb{F}$ be a field of characteristic zero. If the border formula complexity of the $n \times n$ determinant is $n^{\omega(1)}$, then there is a VNC^0-computable hitting set generator that hits VF.*

[4] For technical reasons, we can only prove that our generator is pseudorandom over fields of characteristic zero or sufficiently large characteristic. See [3, Section 4] for an explanation of the underlying issue.

Although it is commonly conjectured that the formula complexity of the determinant is $n^{\omega(1)}$, it is less clear whether we should expect the same lower bound to hold for border formulas, as border computation is poorly understood even in very weak settings. Despite this gap in our understanding, we find it conceivable that the determinant does require border formulas of super-polynomial size. The reason for this is that recent progress on arithmetic circuit lower bounds [36, 46, 34, 35, 19, 16] has relied on the use of complexity measures based on matrix rank. Because matrix rank is lower semi-continuous, these lower bounds often directly imply lower bounds on border complexity.[5] It is unclear if these methods will prove lower bounds against arithmetic formulas in the near future, and if they do, the hard polynomial may not be the determinant. However, in light of the recent success of rank-based methods in proving lower bounds, coupled with the fact that many of these lower bounds apply to the determinant, it is not out of the question that when we manage to prove lower bounds against arithmetic formulas, the methods used will be rank-based and will apply to the determinant. In that case, we will have corresponding lower bounds on the border formula complexity of the determinant.

Our second conditional construction produces a VNC^0-computable generator that hits polynomials that are computable by polynomial-size arithmetic branching programs, a class commonly known as VBP. This result assumes there is a family of polynomials that can be computed by polynomial-size arithmetic circuits but not by polynomial-size arithmetic branching programs.

▶ **Theorem 1.3.** *Let $\mathbb{F}$ be a field of characteristic zero. If there is a family of polynomials in* VP *that require arithmetic branching programs of super-polynomial size, then there is a* VNC^0*-computable hitting set generator that hits* VBP*.*

As a corollary of our generator constructions, we obtain new lower bounds for subsystems of the Geometric Ideal Proof System of Grochow and Pitassi [23, Appendix B], a restricted form of the Ideal Proof System. An easy observation, already made by Grochow, Kumar, Saks, and Saraf [22], shows that hitting set generators immediately yield hard instances for the Geometric Ideal Proof System. The complexity of the generator upper bounds the complexity of the equations in the hard instance, and the hitting property of the generator is used to infer the lower bound against Geometric IPS. Because we construct VNC^0-computable generators against VAC^0, we obtain a system of equations where every equation can be written as an arithmetic formula of constant size, but no Geometric IPS refutation can be computed by a circuit of constant depth and polynomial size.

▶ **Theorem 1.4.** *Let $\mathbb{F}$ be a field of characteristic zero. There is an explicit system of polynomial equations $\mathcal{F}_n$ such that (1) each equation in $\mathcal{F}_n$ depends on at most 3 variables, (2) the system $\mathcal{F}_n$ can be refuted by Geometric IPS, and (3) any Geometric IPS refutation of $\mathcal{F}_n$ cannot be computed by a circuit of constant depth and polynomial size.*

We also prove conditional lower bounds against Geometric IPS refutations computed by arithmetic formulas and arithmetic branching programs. The conditions used to prove these lower bounds are the same ones used to construct VNC^0-computable generators that hit arithmetic formulas and arithmetic branching programs, respectively.

Although our hard instances do not correspond to encodings of 3CNF formulas, we view these results as progress towards proving lower bounds for refutations of unsatisfiable boolean formulas. As mentioned in Section 1.4, unsatisfiable 3CNF formulas on n variables can

[5] See e.g. [4, Section 6.1] for an explanation of how the lower bound of Limaye, Srinivasan, and Tavenas [36] implies a lower bound on border complexity.

always be refuted in degree $O(n)$, so any method of proving lower bounds on the complexity of refuting 3CNF's must tolerate the existence of low-degree refutations. In all of our lower bounds for Geometric IPS, the hard systems of equations admit refutations of degree $n^{O(1)}$. To the best of our knowledge, this is the first example of a system of equations where each equation can be expressed by an arithmetic formula of constant size, the system admits low-degree refutations, and the system is provably hard to refute in a nontrivial subsystem of IPS.

1.6 Our Techniques

Due to space constraints, this extended abstract only presents a high-level overview of our techniques. We refer the reader to the full version of this work [3] for further details.

We obtain our VNC^0-computable hitting set generators through a transformation that takes a known hitting set generator $\mathcal{G}$ and compiles it into a related generator $\mathcal{G}'$ that has much lower complexity, yet retains the hitting property of $\mathcal{G}$. A similar high-level approach was taken by Applebaum, Ishai, and Kushilevitz [5] to obtain NC^0-computable one-way functions and pseudorandom generators in the boolean setting. There are some superficial similarities between our work and theirs, since both works obtain the new generator $\mathcal{G}'$ through an encoding of the computation of $\mathcal{G}$.

However, one difference between our work and [5] is the set of requirements placed on the high-complexity generator $\mathcal{G}$. Applebaum, Ishai, and Kushilevitz [5] require $\mathcal{G}$ to be of sufficiently low complexity (say, NC^1-computable), but are agnostic to the particular choice of the generator $\mathcal{G}$. In contrast, our work does not place a requirement on the complexity of the starting generator, but we are only able to handle generators $\mathcal{G}$ that are of the specific form

$$\mathcal{G} : (x_1, \ldots, x_n) \mapsto (x_1, \ldots, x_n, f(\overline{x})),$$

where $f(\overline{x})$ is a polynomial.

We require the starting generator to be of this form because it provides sufficient algebraic structure to completely analyze the resulting low-complexity generator $\mathcal{G}'$. To every generator $\mathcal{G} : \mathbb{F}^\ell \to \mathbb{F}^n$, one can associate its *annihilator ideal* $\mathrm{Ann}(\mathcal{G})$, the set of polynomials $h \in \mathbb{F}[z_1, \ldots, z_n]$ such that $h \circ \mathcal{G} = 0$, i.e., the composition of h and $\mathcal{G}$ results in the identically zero polynomial. To prove that a generator $\mathcal{G}$ hits a circuit class $\mathscr{C}$, it is both necessary and sufficient to show that every nonzero polynomial in the annihilator $\mathrm{Ann}(\mathcal{G})$ cannot be computed within the resource bounds of $\mathscr{C}$.

For some generators, we can completely characterize the ideal $\mathrm{Ann}(\mathcal{G})$, which a useful first step towards proving lower bounds for $\mathrm{Ann}(\mathcal{G})$. In the case where $\mathcal{G}$ is of the form $\mathcal{G}(\overline{x}) = (x_1, \ldots, x_n, f(\overline{x}))$, it is easy to show that the annihilator ideal is given by

$$\mathrm{Ann}(\mathcal{G}) = \langle z_{n+1} - f(z_1, \ldots, z_n) \rangle \subseteq \mathbb{F}[z_1, \ldots, z_{n+1}].$$

That is, every annihilator of $\mathcal{G}$ is a multiple of the polynomial $z_{n+1} - f(z_1, \ldots, z_n)$. Because this ideal is generated by a single polynomial (i.e., is a principal ideal), we can leverage existing techniques on polynomial factorization to prove lower bounds for $\mathrm{Ann}(\mathcal{G})$. The argument proceeds by contradiction: if there is a small $\mathscr{C}$-circuit that computes an element of $\mathrm{Ann}(\mathcal{G})$, and if $\mathscr{C}$-circuits are polynomially-closed under factorization, then there is a small $\mathscr{C}$-circuit for $z_{n+1} - f(z_1, \ldots, z_n)$. Thus, if the circuit class $\mathscr{C}$ is closed under factorization (as are VP [30] and VBP [44]), then to prove lower bounds for $\mathrm{Ann}(\mathcal{G})$, it suffices to prove a lower bound on $f(z_1, \ldots, z_n)$. This is a much easier task, since we only have to reason about

a single polynomial f and not an arbitrary multiple of f. In some cases, the circuit class $\mathscr{C}$ is not known to be polynomially-closed under factorization. Despite this, we can still make use of results that show for a *specific choice* of f, lower bounds on the $\mathscr{C}$-circuit complexity of f imply lower bounds for all multiples of f.

So far, we have seen that it is possible to completely understand generators of the form $\mathcal{G}(x_1, \ldots, x_n) = (x_1, \ldots, x_n, f(\overline{x}))$. On its own, this generator has no hope of producing a generator that is of lower complexity than the circuits it hits, since there is always an annihilator of complexity comparable to f.

To obtain a generator of lower complexity, we replace the single output $f(\overline{x})$ by a sequence of $s+1$ outputs that encode a size-s circuit Φ that computes $f(\overline{x})$.[6] For each internal gate of Φ, we introduce a fresh variable y_i, and we include the polynomial

$$y_i - y_j \odot y_k$$

in the output of the generator, where $\odot \in \{+, \times\}$ is the operation labeling the i^{th} gate and the children of the i^{th} gate are gates j and k.[7] We also include y_s as an output so that the generator stretches its $n+s$ inputs to $n+s+1$ outputs. It is clear from the definition that each output of this new generator $\mathcal{G}'$ can be computed by an arithmetic formula of constant size, but it is not obvious why $\mathcal{G}'$ preserves any pseudorandom properties the original generator $\mathcal{G}$ had.

To show that $\mathcal{G}'$ is pseudorandom, we follow the approach suggested above: we determine the annihilator ideal $\mathrm{Ann}(\mathcal{G}')$ and subsequently prove lower bounds on the complexity of all nonzero polynomials in $\mathrm{Ann}(\mathcal{G}')$. As we saw, the annihilator ideal $\mathrm{Ann}(\mathcal{G})$ of the starting generator was generated by $z_{n+1} - f(z_1, \ldots, z_n)$, so we could infer lower bounds on $\mathrm{Ann}(\mathcal{G})$ from lower bounds for f and its multiples. For $\mathcal{G}'$, the annihilator ideal $\mathrm{Ann}(\mathcal{G}')$ is again principal and is generated by a polynomial of the form

$$h(\overline{z}) = z_{n+s+1} - f(z_1, \ldots, z_n) + g(\overline{z}),$$

where g is a structured polynomial that should be thought of as an error term for the purposes of this overview. To prove that $\mathcal{G}'$ is pseudorandom, it suffices to prove lower bounds on the complexity of multiples of h. Because h is close to f, one could hope that lower bounds for multiples of f imply comparable lower bounds for multiples of h. This is precisely the route we follow to show that $\mathcal{G}'$ is pseudorandom. By instantiating the construction of $\mathcal{G}'$ with a polynomial f whose multiples are hard for a circuit class $\mathscr{C}$ (possibly under hardness assumptions), we obtain a generator that is computable in VNC^0 yet hits the much larger class $\mathscr{C}$.

In particular, Theorem 1.3 follows by taking f to be a family of polynomials that can be computed by polynomial-size arithmetic circuits, but not by polynomial-size arithmetic branching programs. Because branching programs are closed under factorization [44], the hardness of f against branching programs implies that all multiples of f are likewise hard for branching programs. This ultimately implies that the corresponding generator $\mathcal{G}'$ hits polynomial-size arithmetic branching programs. To prove Theorems 1.1 and 1.2, we take the hard polynomial f to be the determinant. Although low-depth arithmetic circuits and

[6] This transformation is similar to the reduction of Circuit-SAT to 3CNF-SAT and was recently used by Grochow [21] to study the PIT instances that arise from verification of IPS refutations.

[7] If the children of the i^{th} gate are not internal gates, we use a different polynomial, but this is a technical point that does not meaningfully impact the overview here. See [3, Definition 3.1] for the precise definition.

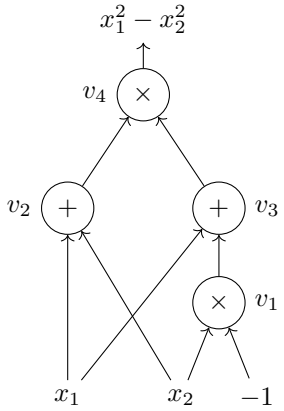

Figure 1 An arithmetic circuit computing the polynomial $x_1^2 - x_2^2$.

arithmetic formulas are not known to be closed under factorization, Andrews and Forbes [4] showed that multiples of the determinant are essentially as hard as the determinant for both of these circuit classes. We then obtain Theorems 1.1 and 1.2 using known lower bounds on the size of low-depth circuits that compute the determinant [36] and the assumed lower bound on the border formula complexity of the determinant, respectively.

We end this overview with a concrete example of the new generator $\mathcal{G}'$ when the polynomial f is taken to be $f(x_1, x_2) = x_1^2 - x_2^2$. This polynomial is too simple to yield a generator with useful pseudorandom properties, but its simplicity allows us to explicitly write down the resulting generator and its annihilator ideal.

▶ **Example 1.5.** Let Φ be the arithmetic circuit depicted in Figure 1 that computes the polynomial $x_1^2 - x_2^2$. The internal gates of Φ are labeled as v_1, v_2, v_3, and v_4, corresponding to a topological ordering of the internal gates. Consider the polynomial map $\mathcal{G} : \mathbb{F}^6 \to \mathbb{F}^7$ given by

$$\mathcal{G}(x_1, x_2, y_1, y_2, y_3, y_4) = (x_1, x_2, y_1 + x_2, y_2 - x_1 - x_2, y_3 - x_1 - y_1, y_4 - y_2 y_3, y_4).$$

What do the annihilators of $\mathcal{G}$ look like? One annihilator can be constructed by writing y_4 as a polynomial combination of the first 6 outputs of $\mathcal{G}$ and then taking the difference of this polynomial and z_7. To do this, we iteratively find polynomials $h_i \in \mathbb{F}[z_1, \ldots, z_7]$ so that $(h_i \circ \mathcal{G})(\overline{x}, \overline{y}) = y_i$. The desired polynomials h_1, h_2, h_3, and h_4 are given by

$$\begin{aligned} h_1(\overline{z}) &= z_3 - z_2 \\ h_2(\overline{z}) &= z_4 + z_1 + z_2 \\ h_3(\overline{z}) &= z_5 + z_1 + h_1(\overline{z}) \\ h_4(\overline{z}) &= z_6 + h_2(\overline{z}) h_3(\overline{z}). \end{aligned}$$

Because $(h_4 \circ \mathcal{G})(\overline{x}, \overline{y}) = y_4$, it follows that

$$h(\overline{z}) \coloneqq h_4(\overline{z}) - z_7$$

is a nonzero annihilator of $\mathcal{G}$. Because $\mathrm{Ann}(\mathcal{G})$ is an ideal, every multiple of h is also an annihilator of $\mathcal{G}$. Using computer software, such as Macaulay2, one can verify that these are the only annihilators of $\mathcal{G}$: the ideal $\mathrm{Ann}(\mathcal{G})$ is precisely the principal ideal generated by h.

To prove that this generator is pseudorandom, we need to understand the complexity of its annihilators. As a first step towards this, let's understand how the complexity of h relates to the complexity of $x_1^2 - x_2^2$, the polynomial computed by Φ. Expanding out $h(\overline{z})$ as

$$h(\overline{z}) = z_1^2 - z_2^2 + z_1z_3 + z_2z_3 + z_1z_4 - z_2z_4 + z_3z_4 + z_1z_5 + z_2z_5 + z_4z_5 + z_6 - z_7,$$

we rewrite $h(\overline{z})$ as

$$h(\overline{z}) = \left(z_1^2 - z_2^2\right) + g(\overline{z}) - z_7,$$

where

$$g(\overline{z}) \coloneqq z_1z_3 + z_2z_3 + z_1z_4 - z_2z_4 + z_3z_4 + z_1z_5 + z_2z_5 + z_4z_5 + z_6.$$

Importantly, the polynomial g is an element of the ideal $\langle z_3, z_4, z_5, z_6\rangle \subseteq \mathbb{F}[\overline{z}]$. In other words, every monomial of g is divisible by one of the variables z_3, z_4, z_5, or z_6. By setting $z_3 = \cdots = z_7 = 0$, we see that h projects to $z_1^2 - z_2^2$, the polynomial computed by the circuit Φ. This transformation of h has low complexity, so a small circuit that computes h can be used to compute $z_1^2 - z_2^2$ with similar complexity. In the contrapositive, a lower bound on the complexity of $z_1^2 - z_2^2$ implies a comparable lower bound on h. This reduction is not particularly useful for the specific example at hand, as the polynomial $z_1^2 - z_2^2$ is easy to compute, so the specific generator $\mathcal{G}$ considered above will not be meaningfully pseudorandom.

Although this example did not lead to a useful generator, it illustrates some features of the general case. The ideal $\mathrm{Ann}(\mathcal{G})$ is always principal and is generated by a polynomial with structure similar to that of h above. In the general case, we can transform h to the polynomial computed by Φ by zeroing out some variables, shifting other variables by a constant, and adding an appropriate constant to h. Because this transformation relating h and the polynomial computed by Φ is of low complexity, we can prove lower bounds on the complexity of h and its multiples by appealing to lower bounds on the complexity of the polynomial computed by Φ and its multiples. In turn, if we select h to be a polynomial whose multiples are all hard for a circuit class $\mathscr{C}$, the resulting generator $\mathcal{G}$ will hit $\mathscr{C}$, since no annihilator of $\mathcal{G}$ can be computed within the resource bounds of $\mathscr{C}$.

References

1. Yaroslav Alekseev, Dima Grigoriev, Edward A. Hirsch, and Iddo Tzameret. Semialgebraic proofs, ips lower bounds, and the τ-conjecture: Can a natural number be negative? *SIAM Journal on Computing*, 53(3):648–700, 2024. doi:10.1137/20M1374523.
2. Robert Andrews. Algebraic Hardness Versus Randomness in Low Characteristic. In Shubhangi Saraf, editor, *35th Computational Complexity Conference (CCC 2020)*, volume 169 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 37:1–37:32, Dagstuhl, Germany, 2020. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.CCC.2020.37.
3. Robert Andrews. Algebraic Pseudorandomness in VNC^0, 2025. arXiv:2505.10675.
4. Robert Andrews and Michael A. Forbes. Ideals, determinants, and straightening: Proving and using lower bounds for polynomial ideals. In *Proceedings of the 54th Annual ACM Symposium on Theory of Computing (STOC 2022)*, pages 389–402, 2022. doi:10.1145/3519935.3520025.
5. Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. Cryptography in NC^0. *SIAM Journal on Computing*, 36(4):845–888, 2006. doi:10.1137/S0097539705446950.
6. Paul Beame, Russell Impagliazzo, Jan Krajíček, Toniann Pitassi, and Pavel Pudlák. Lower bounds on Hilbert's Nullstellensatz and propositional proofs. *Proceedings of the London Mathematical Society*, 73(3):1–26, 1996. Preliminary version in the *35th Annual IEEE Symposium on Foundations of Computer Science (FOCS 1994)*. doi:10.1112/plms/s3-73.1.1.

7 W. Dale Brownawell. Bounds for the degrees in the nullstellensatz. *Annals of Mathematics*, 126(3):577–591, 1987. doi:10.2307/1971361.

8 Sam Buss, Russell Impagliazzo, Jan Krajíček, Pavel Pudlák, Alexander A. Razborov, and Jiři Sgall. Proof complexity in algebraic systems and bounded depth Frege systems with modular counting. *Computational Complexity*, 6:256–298, 1996. doi:10.1007/BF01294258.

9 Léandro Caniglia, André Galligo, and Joos Heintz. Borne simple exponentielle pour les degrés dans le théorème des zéros sur un corps de caractéristique quelconque. *C. R. Acad. Sci. Paris Sér. I Math.*, 307(6):255–258, 1988.

10 Prerona Chatterjee, Mrinal Kumar, Adrian She, and Ben Lee Volk. Quadratic lower bounds for algebraic branching programs and formulas. *Comput. Complex.*, 31(2):8, 2022. doi:10.1007/S00037-022-00223-8.

11 Prerona Chatterjee and Anamay Tengse. Lower bounds from succinct hitting sets, 2025. doi:10.48550/arXiv.2309.07612.

12 Chi-Ning Chou, Mrinal Kumar, and Noam Solomon. Closure results for polynomial factorization. *Theory of Computing*, 15(13):1–34, 2019. Preliminary version in the *33rd Annual Computational Complexity Conference (CCC 2018)*. doi:10.4086/toc.2019.v015a013.

13 Pranjal Dutta and Sumanta Ghosh. Advances in polynomial identity testing. *SIGACT News*, 55(2):53–88, 2024. doi:10.1145/3674159.3674165.

14 Zeev Dvir, Amir Shpilka, and Amir Yehudayoff. Hardness-randomness tradeoffs for bounded depth arithmetic circuits. *SIAM J. Comput.*, 39(4):1279–1293, 2009. doi:10.1137/080735850.

15 Noaï Fitchas and André Galligo. Nullstellensatz effectif et conjecture de Serre (théorème de Quillen-Suslin) pour le calcul formel. *Math. Nachr.*, 149:231–253, 1990. doi:10.1002/mana.19901490118.

16 Michael A. Forbes. Low-Depth Algebraic Circuit Lower Bounds over Any Field. In Rahul Santhanam, editor, *39th Computational Complexity Conference (CCC 2024)*, volume 300 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 31:1–31:16, Dagstuhl, Germany, 2024. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.CCC.2024.31.

17 Michael A. Forbes, Mrinal Kumar, and Ramprasad Saptharishi. Functional Lower Bounds for Arithmetic Circuits and Connections to Boolean Circuit Complexity. In Ran Raz, editor, *Proceedings of the 31st Annual Computational Complexity Conference (CCC 2016)*, volume 50 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 33:1–33:19, Dagstuhl, Germany, 2016. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.CCC.2016.33.

18 Michael A. Forbes, Amir Shpilka, Iddo Tzameret, and Avi Wigderson. Proof complexity lower bounds from algebraic circuit complexity. *Theory of Computing*, 17(10):1–88, 2021. doi:10.4086/toc.2021.v017a010.

19 Hervé Fournier, Nutan Limaye, Srikanth Srinivasan, and Sébastien Tavenas. On the power of homogeneous algebraic formulas. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing (STOC 2024)*, STOC 2024, pages 141–151, New York, NY, USA, 2024. Association for Computing Machinery. doi:10.1145/3618260.3649760.

20 Nashlen Govindasamy, Tuomas Hakoniemi, and Iddo Tzameret. Simple hard instances for low-depth algebraic proofs. In *Proceedings of the 63rd Annual IEEE Symposium on Foundations of Computer Science (FOCS 2022)*, pages 188–199, 2022. doi:10.1109/FOCS54457.2022.00025.

21 Joshua A. Grochow. Polynomial identity testing and the ideal proof system: Pit is in np if and only if ips can be p-simulated by a cook-reckhow proof system, 2023. doi:10.48550/arXiv.2306.02184.

22 Joshua A. Grochow, Mrinal Kumar, Michael Saks, and Shubhangi Saraf. Towards an algebraic natural proofs barrier via polynomial identity testing, 2017. doi:10.48550/arXiv.1701.01717.

23 Joshua A. Grochow and Toniann Pitassi. Circuit complexity, proof complexity, and polynomial identity testing: The ideal proof system. *J. ACM*, 65(6):37:1–37:59, November 2018. doi:10.1145/3230742.

24 Zeyu Guo, Mrinal Kumar, Ramprasad Saptharishi, and Noam Solomon. Derandomization from algebraic hardness. *SIAM Journal on Computing*, 51(2):315–335, 2022. doi:10.1137/20M1347395.

25 Tuomas Hakoniemi, Nutan Limaye, and Iddo Tzameret. Functional lower bounds in algebraic proofs: Symmetry, lifting, and barriers. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing (STOC 2024)*, pages 1396–1404, New York, NY, USA, 2024. Association for Computing Machinery. doi:10.1145/3618260.3649616.

26 Grete Hermann. Die frage der endlich vielen schritte in der theorie der polynomideale. (unter benutzung nachgelassener sätze von k. hentzelt). *Mathematische Annalen*, 95:736–788, 1926. doi:10.1007/BF01206635.

27 Russell Impagliazzo, Pavel Pudlák, and Jiří Sgall. Lower bounds for the polynomial calculus and the Gröbner basis algorithm. *Computational Complexity*, 8:127–144, 1999. doi:10.1007/s000370050024.

28 Zbigniew Jelonek. On the effective Nullstellensatz. *Invent. Math.*, 162(1):1–17, 2005. doi:10.1007/s00222-004-0434-8.

29 Valentine Kabanets and Russell Impagliazzo. Derandomizing polynomial identity tests means proving circuit lower bounds. *Computational Complexity*, 13(1-2):1–46, 2004. doi:10.1007/s00037-004-0182-6.

30 Erich Kaltofen. Single-factor hensel lifting and its application to the straight-line complexity of certain polynomials. In *Proceedings of the 19th Annual ACM Symposium on Theory of Computing, 1987, New York, New York, USA*, pages 443–452, 1987. doi:10.1145/28395.28443.

31 Neeraj Kayal. The complexity of the annihilating polynomial. In *Proceedings of the 24th Annual IEEE Conference on Computational Complexity (CCC 2009)*, pages 184–193, 2009. doi:10.1109/CCC.2009.37.

32 János Kollár. Sharp effective Nullstellensatz. *J. Amer. Math. Soc.*, 1(4):963–975, 1988. doi:10.2307/1990996.

33 Teresa Krick, Luis Miguel Pardo, and Martín Sombra. Sharp estimates for the arithmetic Nullstellensatz. *Duke Mathematical Journal*, 109(3):521–598, 2001. doi:10.1215/S0012-7094-01-10934-4.

34 Deepanshu Kush and Shubhangi Saraf. Improved Low-Depth Set-Multilinear Circuit Lower Bounds. In Shachar Lovett, editor, *37th Computational Complexity Conference (CCC 2022)*, volume 234 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 38:1–38:16, Dagstuhl, Germany, 2022. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.CCC.2022.38.

35 Deepanshu Kush and Shubhangi Saraf. Near-Optimal Set-Multilinear Formula Lower Bounds. In Amnon Ta-Shma, editor, *38th Computational Complexity Conference (CCC 2023)*, volume 264 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 15:1–15:33, Dagstuhl, Germany, 2023. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.CCC.2023.15.

36 Nutan Limaye, Srikanth Srinivasan, and Sébastien Tavenas. Superpolynomial lower bounds against low-depth algebraic circuits. In *Proceedings of the 62nd Annual IEEE Symposium on Foundations of Computer Science (FOCS 2021)*, pages 804–814, 2021. doi:10.1109/FOCS52979.2021.00083.

37 Noam Nisan and Avi Wigderson. Hardness vs. randomness. *J. Comput. System Sci.*, 49(2):149–167, 1994. doi:10.1016/S0022-0000(05)80043-1.

38 Alexander A. Razborov. Lower bounds for the polynomial calculus. *Computational Complexity*, 7:291–324, 1998. doi:10.1007/s000370050013.

39 Hanlin Ren, Rahul Santhanam, and Zhikun Wang. On the range avoidance problem for circuits. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 640–650, 2022. doi:10.1109/FOCS54457.2022.00067.

40 Rahul Santhanam and Iddo Tzameret. Iterated lower bound formulas: A diagonalization-based approach to proof complexity. In *Proceedings of the 53rd Annual ACM Symposium on Theory of Computing (STOC 2021)*, pages 234–247, New York, NY, USA, 2021. Association for Computing Machinery. doi:10.1145/3406325.3451010.

41 Nitin Saxena. Progress on polynomial identity testing. *Bulletin of the EATCS*, 99:49–79, 2009.

42 Nitin Saxena. Progress on polynomial identity testing ii. In *Proceedings of the Workshop celebrating Somenath Biswas' 60th Birthday*, pages 131–146, 2014.

43 Jacob T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *J. ACM*, 27(4):701–717, 1980. doi:10.1145/322217.322225.

44 Amit Sinhababu and Thomas Thierauf. Factorization of polynomials given by arithmetic branching programs. *Computational Complexity*, 30(15), 2021. doi:10.1007/s00037-021-00215-0.

45 Martín Sombra. A sparse effective Nullstellensatz. *Adv. in Appl. Math.*, 22(2):271–295, 1999. doi:10.1006/aama.1998.0633.

46 Sébastien Tavenas, Nutan Limaye, and Srikanth Srinivasan. Set-multilinear and non-commutative formula lower bounds for iterated matrix multiplication. In *Proceedings of the 54th Annual ACM Symposium on Theory of Computing (STOC 2022)*, STOC 2022, pages 416–425, New York, NY, USA, 2022. Association for Computing Machinery. doi:10.1145/3519935.3520044.

47 Richard Zippel. Probabilistic algorithms for sparse polynomials. In *Proceedings of the International Symposium on Symbolic and Algebraic Computation, EUROSAM 1979*, pages 216–226, 1979. doi:10.1007/3-540-09519-5_73.