Algebraic Hardness Versus Randomness in Low Characteristic

Robert Andrews
Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL, USA
rgandre2@illinois.edu

Abstract

We show that lower bounds for explicit constant-variate polynomials over fields of characteristic $p > 0$ are sufficient to derandomize polynomial identity testing over fields of characteristic p. In this setting, existing work on hardness-randomness tradeoffs for polynomial identity testing requires either the characteristic to be sufficiently large or the notion of hardness to be stronger than the standard syntactic notion of hardness used in algebraic complexity. Our results make no restriction on the characteristic of the field and use standard notions of hardness.

We do this by combining the Kabanets-Impagliazzo generator with a white-box procedure to take p^{th} roots of circuits computing a p^{th} power over fields of characteristic p. When the number of variables appearing in the circuit is bounded by some constant, this procedure turns out to be efficient, which allows us to bypass difficulties related to factoring circuits in characteristic p.

We also combine the Kabanets-Impagliazzo generator with recent "bootstrapping" results in polynomial identity testing to show that a sufficiently-hard family of explicit constant-variate polynomials yields a near-complete derandomization of polynomial identity testing. This result holds over fields of both zero and positive characteristic and complements a recent work of Guo, Kumar, Saptharishi, and Solomon, who obtained a slightly stronger statement over fields of characteristic zero.

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

Keywords and phrases Polynomial identity testing, hardness versus randomness, low characteristic

Digital Object Identifier 10.4230/LIPIcs.CCC.2020.37

Funding Supported by NSF grant CCF-1755921.

Acknowledgements We would like to thank Michael A. Forbes for many useful comments which helped improve the presentation of this work.

1 Introduction

The interaction between computational hardness and pseudorandomness is a central theme of computational complexity. The goal of this vein of work is to show that a class $\mathcal{C}$ of problems that are solvable by randomized algorithms can in fact be solved by deterministic algorithms which are not much slower than the known randomized algorithm, assuming lower bounds for a related class $\mathcal{D}$. When trying to derandomize BPP, the class of problems solvable in polynomial time by a randomized Turing machine with failure probability at most $1/3$, we understand this problem quite well. A series of works culminated in that of Impagliazzo and Wigderson [20], which showed that BPP = P if there are problems in E which require boolean circuits of exponential size. Subsequent work by Shaltiel and Umans [36] and Umans [40] further tightened the quantitative tradeoffs obtainable for derandomizing BPP.

In this work, we focus on the question of hardness versus randomness in the more restricted computational model of algebraic circuits, which naturally compute multivariate polynomials over a specified base field $\mathbb{F}$. Here, the algorithmic problem of interest is *polynomial identity testing* (PIT), which is the problem of determining if a given algebraic circuit computes the

identically zero polynomial. We typically consider identity testing of circuits whose size and degree are bounded by a polynomial function in the number of variables. This low-degree regime captures polynomials of interest to computer scientists, such as the determinant and permanent, and corresponds to typical algorithmic applications of PIT. In this regime, the problem of PIT is easily solved with randomness by evaluating the circuit at a randomly chosen point of a large enough grid. The correctness of this algorithm follows from the Schwartz-Zippel lemma, which roughly says that a low-degree multivariate polynomial cannot vanish at many points of a sufficiently large grid. To date, no deterministic algorithm for PIT is known that substantially improves on the naïve derandomization of the Schwartz-Zippel lemma.

Polynomial identity testing has widespread applications in theoretical computer science and has led to randomized algorithms for perfect matching [29, 23, 30], primality testing [1, 3], and equivalence testing of read-once branching programs [6], among other problems. In light of the utility of PIT as an algorithmic primitive, it is worth understanding to what extent PIT can be derandomized. There is a large body of work concerned with unconditional derandomization of PIT for various sub-classes of algebraic circuits. For more on this, we refer the reader to the surveys of Shpilka and Yehudayoff [38] and Saxena [34, 35]. In this work, we will focus on conditional derandomization of PIT under suitable hardness assumptions.

1.1 Prior Work

The first instantiation of the hardness-randomness paradigm for polynomial identity testing was given by Kabanets and Impagliazzo [21]. Their work implemented the design-based approach of Nisan and Wigderson [31] in the algebraic setting, showing that lower bounds for an explicit family of multivariate polynomials can be used to derandomize PIT.

Subsequent work by Dvir, Shpilka, and Yehudayoff [13] and Chou, Kumar, and Solomon [12] extended this to the setting of bounded-depth circuits, roughly showing that lower bounds against depth-$(\Delta + O(1))$ circuits suffice to derandomize identity testing of depth-Δ circuits, for any constant Δ. The result of Dvir, Shpilka, and Yehudayoff [13] works with any hard polynomial, but scales poorly with the individual degree of the circuit being tested. Chou, Kumar, and Solomon [12] refined the approach of Dvir, Shpilka, and Yehudayoff [13] and showed that if the family of hard polynomials has sufficiently low degree, then this dependence on the individual degree of the circuit being tested can be avoided. Implementing the hardness-randomness paradigm in low-depth is motivated in part by a host of depth-reduction results in algebraic complexity [4, 24, 39, 18] which show that polynomials computable by small circuits can be computed by non-trivially small low-depth circuits.

Returning to the setting of unrestricted circuits, recent work of Guo, Kumar, Saptharishi, and Solomon [17] uses a stronger hardness assumption than that of Kabanets and Impagliazzo [21] and obtains a stronger derandomization of PIT. Specifically, Guo, Kumar, Saptharishi, and Solomon [17] obtain a polynomial-time derandomization of PIT using lower bounds against an explicit family of constant-variate polynomials. For comparison, Kabanets and Impagliazzo [21] only obtain quasipolynomial-time algorithms for PIT under multivariate hardness assumptions. In Section 6 of this work, we further discuss the relationship between these hardness assumptions and provide evidence for the strength of constant-variate hardness compared to multivariate hardness.

A separate line of work by Agrawal, Ghosh, and Saxena [2] and Kumar, Saptharishi, and Tengse [27] shows that PIT exhibits a "bootstrapping" phenomenon. That is, if one can obtain a barely non-trivial derandomization of PIT for circuits of size and degree which are unbounded in the number of variables, then it follows that there is a near-complete derandomization of PIT for circuits of polynomial size and degree.

From these works, we have a relatively good understanding of what derandomization of PIT is possible under hardness assumptions. However, excluding the bootstrapping results of Agrawal, Ghosh, and Saxena [2] and Kumar, Saptharishi, and Tengse [27], all previous work on hardness-randomness tradeoffs for PIT requires the underlying field to be of zero or large characteristic (for the definition of the characteristic of a field, see Section 2). That is, we can derandomize PIT under hardness assumptions over the complex numbers $\mathbb{C}$ or the finite field of p^m elements $\mathbb{F}_{p^m}$ when p is sufficiently large, but we do not know how to do the same over a field of low characteristic like $\mathbb{F}_{2^m}$.

A partial exception to this deficiency is the work of Kabanets and Impagliazzo [21]. Their results yield derandomization of PIT over a finite field $\mathbb{F}_{p^m}$ assuming an explicit polynomial which is hard to compute *as a function* over $\mathbb{F}_{p^m}$. Over infinite fields, two polynomials are equal if and only if they compute the same function. However, this no longer holds over finite fields. For example, over $\mathbb{F}_2$, the polynomial $x^2 - x$ computes the zero function but is decidedly not the zero polynomial. It is more common in the study of algebraic circuits to prove lower bounds on the task of computing a polynomial as a syntactic object, not as a function. Functional lower bounds imply syntactic lower bounds, but the reverse direction does not hold, which makes proving functional lower bounds a harder task.

If one inspects the proof of Kabanets and Impagliazzo [21], the functional hardness assumption can be replaced with a slightly weaker, albeit non-standard, syntactic hardness assumption. Namely, it suffices to assume the existence of an explicit family of n-variate polynomials $\{f_n : n \in \mathbb{N}\}$ such that $f_n^{p^k}$ is hard in the syntactic sense for $1 \leqslant p^k \leqslant 2^{O(n)}$. Over characteristic zero fields, the factoring algorithm of Kaltofen [22] implies that if f is hard to compute, then f^d is comparably hard to compute as long as d is not too large. Over fields of characteristic p, it is not clear if hardness of f^p is implied by hardness of f. For example, it is consistent with our current state of knowledge that the $n \times n$ permanent $\mathrm{perm}_n(\overline{x})$ is $2^{\Omega(n)}$-hard over $\mathbb{F}_3$, but that $\mathrm{perm}_n(\overline{x})^3$ is computable by circuits of size $O(n^2)$ over $\mathbb{F}_3$. Understanding the relationship between the complexity of f and f^p over fields of characteristic $p > 0$ in general remains a challenging open problem.

For further exposition on hardness-randomness tradeoffs for PIT, see the recent survey of Kumar and Saptharishi [26].

1.2 Identity Testing in Low Characteristic

Before describing our contributions, we take a detour to look more closely at the question of derandomizing PIT over fields of low characteristic. Known techniques for derandomizing PIT over fields of small characteristic under hardness assumptions fail due to the fact that over a field of positive characteristic, the derivative of a non-constant polynomial may be zero. For example, over $\mathbb{F}_2$, we have $\frac{\partial}{\partial x}(x^2) = 2x = 0$, since $2 = 0$ in $\mathbb{F}_2$. Thus, techniques which are in some sense "analytic" break in low characteristic. Given that the problem of polynomial identity testing is entirely algebraic, it would be nice to find an "algebraic" approach that does not succumb to this flaw. Indeed, derandomizing PIT in low characteristic fields under hardness assumptions is listed as an open problem in the recent survey of Kumar and Saptharishi [26] on algebraic derandomization.

The problem of derandomizing PIT in low characteristic fields also has interesting algorithmic applications. Consider, for example, the randomized algorithm of Lovász [29] to detect whether a bipartite graph has a perfect matching. Let $G = (V_1 \sqcup V_2, E)$ be a balanced bipartite graph on $2n$ vertices with partite sets V_1 and V_2. We form the $n \times n$ symbolic matrix A given by

$$A_{i,j} = \begin{cases} x_{i,j} & \{i,j\} \in E \\ 0 & \text{otherwise.} \end{cases}$$

It is not hard to see that $\det(A) \neq 0$ if and only if G has a perfect matching. We can then check if G has a perfect matching by evaluating A at a random point chosen from a suitably large grid of integers.

In evaluating $\det(A)$, we may encounter large numbers of size $\Omega(n!)$. Arithmetic on such numbers is expensive, requiring at least $\Omega(n \log n)$ time. We could instead implement this algorithm over a finite field of size $\text{poly}(n)$. As the determinant is a polynomial of degree n, the Schwartz-Zippel lemma guarantees that this modification yields an algorithm with low error probability. What we have gained is the fact that elements of such a finite field can be represented in $O(\log n)$ bits, so our arithmetic becomes more efficient. In principle, one could choose the field so that the characteristic is large enough for the the hardness-randomness paradigm to apply, but there may be other considerations which motivate picking, say, an extension field of $\mathbb{F}_2$. Derandomizing such an algorithm (under hardness assumptions) requires extending the hardness-randomness paradigm to fields of low characteristic.

Alternatively, one can reduce the bit complexity by using a derandomized polynomial identity testing algorithm over the rational numbers, but with the arithmetic performed modulo a small prime number. This approach also achieves logarithmic bit complexity. However, we are now in the position of having to derandomize the selection of the prime number. It is not obvious how to do this much faster than brute force, so the benefits of reducing the bit complexity are negated by the need to try many different primes.

While the previous example may seem somewhat artificial, we remark that there are instances of algorithms which explicitly rely on polynomial identity testing over fields of low characteristic. For example, the randomized algorithm of Williams [41] for the k-path problem makes use of polynomial identity testing over fields of characteristic 2. If one wanted to derandomize this algorithm under a hardness assumption, prior work on hardness-randomness tradeoffs for PIT would not suffice.

1.3 Our Results

In this work, we instantiate the hardness-randomness paradigm for PIT over fields of low characteristic under standard syntactic hardness assumptions. That is, we obtain derandomization of PIT from the existence of an explicit family of hard polynomials $\{f_n : n \in \mathbb{N}\}$ without assuming hardness of p^{th} powers of f_n. At the heart of our results is a new technique for computing the map $f^p \mapsto f$ over $\mathbb{F}[\overline{x}]$ when the polynomial f^p is given by an algebraic circuit. When f depends on a small number of variables, the circuit computing f is not too much larger than the circuit which computes f^p.

▶ **Lemma 1.1** (informal version of Corollary 3.6). *Suppose $f(\overline{x})^p$ is a polynomial on $O(1)$ variables and can be computed by a circuit of size s over a field of characteristic $p > 0$. Then $f(\overline{x})$ can be computed by a circuit of size $O(s)$.*

Using this, we are able to extend the techniques of Kabanets and Impagliazzo [21] to fields of low characteristic. To do so, we need stronger hardness assumptions than those made by Kabanets and Impagliazzo [21] for the case of zero characteristic fields. In algebraic complexity, lower bounds are typically proved for families of polynomials parameterized by the number of variables, as this captures the regime of interest for algorithmic applications. To prove our results, we assume lower bounds against a family of constant-variate polynomials which are parameterized by degree.

For the sake of exposition, we focus on the case of lower bounds for univariate polynomials. A univariate polynomial of degree d can easily be computed by circuits of size $O(d)$ using Horner's rule. It is not hard to show that every such polynomial also requires size $\Omega(\log d)$

to compute. However, improving on this $\Omega(\log d)$ lower bound for an explicit family of polynomials is a long-standing open problem. Standard dimension arguments show that most univariate polynomials of degree d require circuits of size $d^{\Omega(1)}$ to compute.

When comparing statements regarding degree d univariates and degree $n^{O(1)}$ multivariate polynomials on n variables, it is instructive to think of n and $\log d$ as comparable. In this sense, our results achieve the same hardness-randomness tradeoffs as those of Kabanets and Impagliazzo [21], but require translating their hardness assumptions to the comparable statement for univariate polynomials.

Using Lemma 1.1, we can extend the analysis of Kabanets and Impagliazzo to work over fields of low characteristic. We now give two concrete examples of the derandomization we can obtain using this extension.

▶ **Theorem 1.2** (informal version of Theorem 4.3 and Corollary 4.5)**.** *Let $\mathbb{F}$ be a field of characteristic $p > 0$. Let $\{f_d(x) : d \in \mathbb{N}\}$ be an explicit family of univariate polynomials which cannot be computed by circuits of size less than $s(d)$ over $\mathbb{F}$.*

1. *If $s(d) = \log^{\omega(1)} d$, then there is a deterministic algorithm for identity testing of polynomial-size, polynomial-degree circuits over $\mathbb{F}$ in n variables which runs in time $2^{n^{o(1)}}$.*
2. *If $s(d) = 2^{\log^{\Omega(1)} d}$, then there is a deterministic algorithm for identity testing of polynomial-size, polynomial-degree circuits over $\mathbb{F}$ in n variables which runs in time $2^{\log^{O(1)} n}$.*

For comparison, from an $n^{\omega(1)}$ lower bound against a family of explicit multilinear polynomials, Kabanets and Impagliazzo [21] give a deterministic algorithm for PIT over fields of characteristic zero which runs in time $2^{n^{o(1)}}$. If instead one has a $2^{n^{\Omega(1)}}$ lower bound, then their techniques yield a deterministic algorithm which runs in time $2^{\log^{O(1)} n}$. Viewing $\log d$ and n as (roughly) equivalent, we see that our derandomization obtains the same tradeoff between hardness and pseudorandomness as Kabanets and Impagliazzo [21], modulo the difference between univariate and multivariate lower bounds.

It is not hard to show that lower bounds in the constant-variate regime imply comparable lower bounds in the multivariate regime (see Lemma 2.6), but the reverse implication is not known. In Section 6, we investigate the possibility of using known techniques to prove univariate lower bounds from multivariate lower bounds.

As the assumption of a hard univariate family seems strong, it raises the question of whether or not one can obtain a stronger derandomization of PIT over fields of positive characteristic under a univariate hardness assumption. There is evidence this can be done, as Guo, Kumar, Saptharishi, and Solomon [17] use univariate lower bounds to obtain a complete derandomization of PIT over fields of characteristic zero. With a more careful instantiation of the Kabanets-Impagliazzo result, we are able to derandomize PIT in a way that suffices for the bootstrapping results of Agrawal, Ghosh, and Saxena [2] and Kumar, Saptharishi, and Tengse [27] to take effect. This allows us to prove nearly-optimal hardness-randomness tradeoffs for PIT over fields of positive characteristic, which comes close to matching the characteristic zero result of Guo, Kumar, Saptharishi, and Solomon [17]. More concretely, we prove the following.

▶ **Theorem 1.3** (informal version of Theorem 5.3)**.** *Let $\mathbb{F}$ be a field of characteristic $p > 0$. Let $\{f_d(x) : d \in \mathbb{N}\}$ be an explicit family of univariate polynomials which cannot be computed by circuits of size less than d^δ for some constant $\delta > 0$. Then there is a deterministic algorithm for identity testing of polynomial-size, polynomial-degree algebraic circuits in n variables over $\mathbb{F}$ which runs in time $n^{\exp \circ \exp(O(\log^\star n))}$.*

The rest of this work is organized as follows. In Section 2, we establish notation, definitions, and relevant background necessary to state and prove our results. In Section 3, we prove our main technical lemma on computing p^{th} roots of algebraic circuits over fields of characteristic

$p > 0$. We then use this in Section 4 to extend the work of Kabanets and Impagliazzo to the low characteristic setting. We combine our techniques with the bootstrapping results to obtain near-complete derandomization of PIT over fields of positive characteristic in Section 5. Section 6 investigates the relationship between univariate and multivariate circuit lower bounds. We conclude in Section 7 with a collection of problems left open by this work.

2 Preliminaries

For $n \in \mathbb{N}$, we write $[n] \coloneqq \{1, \ldots, n\}$ and $[\![n]\!] \coloneqq \{0, \ldots, n-1\}$. If A is an $n \times m$ matrix, we write $A_{i,\bullet}$ and $A_{\bullet,j}$ for the i^{th} row and j^{th} column of A, respectively. We abbreviate a vector of variables $(x_1, \ldots, x_n)$, numbers $(a_1, \ldots, a_n)$, or field elements $(\alpha_1, \ldots, \alpha_n)$ by $\overline{x}$, $\overline{a}$, and $\overline{\alpha}$, respectively, where the length is usually clear from context. We also abbreviate the product $\prod_{i=1}^{n} x_i^{a_i} =: \overline{x}^{\overline{a}}$. Given a polynomial $f(\overline{x}) = \sum_{\overline{a}} \alpha_{\overline{a}} \overline{x}^{\overline{a}}$, we write $\deg(f)$ and $\operatorname{ideg}(f)$ for the *total degree* and *individual degree* of f, respectively. The total degree of f is given by $\deg(f) \coloneqq \max\{\|\overline{a}\|_1 : \alpha_{\overline{a}} \neq 0\}$, while the individual degree of f is given by $\operatorname{ideg}(f) \coloneqq \max\{\|\overline{a}\|_\infty : \alpha_{\overline{a}} \neq 0\}$.

For a field $\mathbb{F}$, the *characteristic* of $\mathbb{F}$, denoted $\operatorname{char}\mathbb{F}$, is the smallest positive integer p such that $p \cdot 1 = 0$ in $\mathbb{F}$. In the case that there is no such p, we say that $\mathbb{F}$ has characteristic zero. Alternatively, $\operatorname{char}\mathbb{F}$ is the number p such that the ring homomorphism $\mathbb{Z} \to \mathbb{F}$ induced by $1 \mapsto 1$ has kernel $p\mathbb{Z}$. The set $\mathcal{C}_{\mathbb{F}}(s, n, d) \subseteq \mathbb{F}[\overline{x}]$ denotes the set of all n-variate degree d polynomials which can be computed by an algebraic circuit of size at most s over $\mathbb{F}$.

2.1 Algebraic Computation and Polynomial Identity Testing

We assume familiarity with the models of algebraic circuits, formulae, and branching programs. When we refer to the *size* of a circuit, formula, or branching program, we mean the number of nodes in the computational device. An introduction to this area can be found in the survey of Shpilka and Yehudayoff [38]. Throughout this work, we analyze our algorithms under the assumption that arithmetic over the base field $\mathbb{F}$ can be performed in constant time.

We now collect basic definitions and results needed for the study of deterministic black-box algorithms for polynomial identity testing. More in-depth exposition is available in the recent survey of Kumar and Saptharishi [26].

We start with the notion of a hitting set, the basic object used to construct deterministic black-box algorithms for polynomial identity testing.

▶ **Definition 2.1.** *Let $\mathcal{C} \subseteq \mathbb{F}[\overline{x}]$ be a set of n-variate polynomials. We say that a set $\mathcal{H} \subseteq \mathbb{F}^n$ is a* hitting set for $\mathcal{C}$ *if for every non-zero $f(\overline{x}) \in \mathcal{C}$, there is a point $\overline{\alpha} \in \mathcal{H}$ such that $f(\overline{\alpha}) \neq 0$. If $\mathcal{H}$ can be computed in $t(n)$ time, then we say that $\mathcal{H}$ is* $t(n)$-explicit.

We now introduce hitting set generators, the analogue of pseudorandom generators in the context of algebraic derandomization.

▶ **Definition 2.2.** *Let $\mathcal{C} \subseteq \mathbb{F}[\overline{x}]$ be a set of n-variate polynomials. Let $\mathcal{G} : \mathbb{F}^m \to \mathbb{F}^n$ be a mapping given by*

$$\mathcal{G}(\overline{y}) = (\mathcal{G}_1(\overline{y}), \ldots, \mathcal{G}_n(\overline{y})),$$

where $\mathcal{G}_i \in \mathbb{F}[\overline{y}]$. We say that $\mathcal{G}$ is a hitting set generator for $\mathcal{C}$ *if for every non-zero $f(\overline{x}) \in \mathcal{C}$, we have $f(\mathcal{G}(\overline{y})) \neq 0$. The* seed length *of $\mathcal{G}$ is m. The* degree *of $\mathcal{G}$ is $\max_{i \in [n]} \deg(\mathcal{G}_i)$. We say $\mathcal{G}$ is* $t(n)$-explicit *if, given $\overline{\alpha} \in \mathbb{F}^m$, we can compute $\mathcal{G}(\overline{\alpha})$ in $t(n)$ time.*

It is a well-known result that an explicit, low-degree hitting set generator for $\mathcal{C}$ with small seed length yields an explicit hitting set for $\mathcal{C}$ of small size. The hitting set is constructed by evaluating the generator on a grid of large enough size. Correctness follows from the Schwartz-Zippel lemma.

▶ **Lemma 2.3.** *Let $\mathcal{C}$ be a set of n-variate degree d polynomials. Let $\mathcal{G} : \mathbb{F}^m \to \mathbb{F}^n$ be a $t(n)$-explicit hitting set generator for $\mathcal{C}$ of degree D. Then there is a $(dD+1)^m t(n)$-explicit hitting set $\mathcal{H}$ for $\mathcal{C}$ of size $(dD+1)^m$.*

We also need a notion of explicitness for a family of polynomials. In previous works on hardness-randomness tradeoffs for polynomial identity testing, a family of n-variate polynomials $\{f_n \in \mathbb{F}[\overline{x}] : n \in \mathbb{N}\}$ is considered explicit if f_n is computable in $\exp(O(n))$ time. However, we will need a slightly different notion of explicitness. Instead of an exponential-time algorithm to compute f_n, we require an exponential-time algorithm to compute the coefficient of a given monomial in f_n. This different notion of explicitness will be used to transition between the constant-variate and multivariate regimes later on in Section 4 and Section 5.

▶ **Definition 2.4.** *Let $\{f_{n,d}(\overline{x}) \in \mathbb{F}[\overline{x}] : n, d \in \mathbb{N}\}$ be a family of n-variate degree d polynomials. We say that this family is* strongly $t(n,d)$-explicit *if there is an algorithm which on input $(n, d, \overline{a})$ outputs the coefficient of $\overline{x}^{\overline{a}}$ in $f_{n,d}(\overline{x})$ in $t(n,d)$ time.*

▶ Remark 2.5. The preceding definition is reminiscent of Valiant's criterion for membership in VNP. Briefly, Valiant's criterion says that if the coefficient of $\overline{x}^{\overline{a}}$ can be computed in #P/poly, then the polynomial $f(\overline{x})$ is in VNP, an algebraic analogue of NP. We refer the reader to Bürgisser [8, Chapters 1 and 2] for further exposition on VNP and Valiant's criterion.

We will repeatedly build explicit families of hard multivariate polynomials out of explicit families of hard constant-variate polynomials. By "a family of hard multivariate polynomials," we mean a family of polynomials $\{f_n(\overline{x}) \in \mathbb{F}[\overline{x}] : n \in \mathbb{N}\}$, where f_n is an n-variate polynomial of degree $n^{O(1)}$. When we say "a family of hard constant-variate polynomials," we mean a family $\{f_d(\overline{x}) \in \mathbb{F}[\overline{x}] : d \in \mathbb{N}\}$, where f_d is a degree d polynomial on $k = O(1)$ variables. That is, when we consider multivariate polynomials, we parameterize the family by the number of variables and primarily consider families of small degree; when we look at constant-variate polynomials, we fix the number of variables in all polynomials and parameterize the family by the degree of the polynomial.

To illustrate how we can obtain hard multivariate polynomials from hard constant-variate polynomials, suppose $g_d(x) = \sum_{i=0}^{d} \alpha_i x^i$ is a hard degree d univariate polynomial. We will define a new polynomial $f_n(\overline{y})$ on $n := \lfloor \log d \rfloor + 1$ variables, where the monomials of f_n correspond to writing each term of g_d "in base 2." More precisely, for each $\overline{e} \in \{0,1\}^n$, let $j(\overline{e})$ be the number whose representation in binary corresponds to $\overline{e}$. We assign the coefficient $\alpha_{j(\overline{e})}$ to the monomial $\overline{y}^{\overline{e}}$ in f_n. To show that f_n is hard, we show the contrapositive: a small circuit for f_n implies a small circuit for g_d, which contradicts the hardness of g_d. The proof of this is relatively straightforward, as we simply find a way to substitute powers of x for each y_i so that the monomial $\overline{y}^{\overline{e}}$ is mapped to $x^{j(\overline{e})}$.

In the case where g_d is a polynomial in multiple variables, we simultaneously write each variable appearing in g_d "in base 2." We remark that there is nothing *a priori* special about our use of base 2. However, doing so yields polynomials which are multilinear, a fact which will be useful later on.

We now make the preceding sketch precise, showing that lower bounds in the constant-variate regime imply comparable lower bounds in the multivariate regime.

▶ **Lemma 2.6.** *Let $g_{m,d}(\overline{x}) = \sum_{\overline{a}} \alpha_{\overline{a}} \overline{x}^{\overline{a}}$ be a strongly $t(m,d)$-explicit m-variate degree d polynomial which requires circuits of size s to compute. Let $j : \{0,1\}^{\lfloor \log d \rfloor + 1} \to [\![2^{\lfloor \log d \rfloor + 1}]\!]$ be given by $j(\overline{e}) = \sum_{i=1}^{\lfloor \log d \rfloor + 1} \overline{e}_i 2^{i-1}$, that is, $j(\overline{e})$ is the number whose binary representation corresponds to $\overline{e}$. Let $\overline{y} = (y_{1,1}, \ldots, y_{1,\lfloor \log d \rfloor + 1}, \ \cdots \ , y_{m,1}, \ldots, y_{m,\lfloor \log d \rfloor + 1})$ and define*

$$f_{m,d}(\overline{y}) = \sum_{\overline{e} \in \{0,1\}^{m \times \lfloor \log d \rfloor + 1}} \alpha_{(j(\overline{e}_{1,\bullet}),\ldots,j(\overline{e}_{m,\bullet}))} \overline{y}^{\overline{e}}.$$

Then $f_{m,d}$ is a strongly $t(m,d)$-explicit multilinear polynomial on $m(\lfloor \log d \rfloor + 1)$ variables which requires circuits of size $s - \Theta(m \log d)$ to compute.

Proof. The fact that $f_{m,d}$ is multilinear is clear from the definition.

To see that $f_{m,d}$ is hard to compute, suppose Φ is a circuit of size t which computes $f_{m,d}$. By applying the Kronecker substitution $y_{i,j} \mapsto x_i^{2^j}$, we can recover a circuit which computes $g_{m,d}(\overline{x})$. This mapping can be computed in size $\Theta(m \log d)$ by repeated squaring, so we obtain a circuit for $g_{m,d}$ of size $t + \Theta(m \log d)$. By assumption, $t + \Theta(m \log d) \geqslant s$, so $t \geqslant s - \Theta(m \log d)$, which proves the lower bound on the circuit complexity of $f_{m,d}$.

Finally, remark that the binary description of a monomial in $f_{m,d}$ is exactly the same as the binary description of a monomial in $g_{m,d}$. This implies we can use the $t(m,d)$-time algorithm to compute the coefficients of $f_{m,d}$, so $f_{m,d}$ inherits the explicitness of $g_{m,d}$. ◀

Whether lower bounds in the multivariate regime imply lower bounds in the constant-variate regime is an open question. In Section 6, we give complexity-theoretic evidence that suggests the technique used to prove the preceding lemma does not suffice to prove constant-variate lower bounds from multivariate lower bounds.

In Section 5, we will run into some technical issues concerning circuits which are defined over a low-degree extension of the base field $\mathbb{F}$. The next lemma says that whenever a circuit Φ is defined over an extension $\mathbb{K} \supseteq \mathbb{F}$ of low degree, such a circuit can in fact be defined over $\mathbb{F}$ without increasing its size too much. A related result was proved in Bürgisser, Clausen, and Shokrollahi [10, §4.3], where the authors considered extensions $\mathbb{K} \supseteq \mathbb{F}$ such that circuits defined over $\mathbb{K}$ have no computational advantage compared to circuits defined over $\mathbb{F}$ when computing a polynomial in $\mathbb{F}[\overline{x}]$.

▶ **Lemma 2.7** ([8, Proposition 4.1(iii)], [19], see also [10, §4.3])**.** *Let $\mathbb{F}$ be a field and let $\mathbb{K} \supseteq \mathbb{F}$ be an extension of degree k. Suppose $f(\overline{x})$ can be computed by a circuit of size s over $\mathbb{K}$. Then there is a circuit of size $O(k^3 s)$ which computes f over $\mathbb{F}$.*

We conclude our preliminaries on algebraic complexity by quoting a celebrated result of Kaltofen which shows that algebraic circuits may be factored without a large increase in size.

▶ **Theorem 2.8** ([22])**.** *Let $f(\overline{x}) \in \mathbb{F}[\overline{x}]$ be a polynomial of degree d computable by an algebraic circuit of size s. Let $g(\overline{x}) \in \mathbb{F}[\overline{x}]$ be a factor of $f(\overline{x})$. Then there is an algebraic circuit of size $s' \leqslant O((snd)^4)$ which computes*

1. *$g(\overline{x})$, in the case that $\operatorname{char} \mathbb{F} = 0$, and*
2. *$g(\overline{x})^{p^k}$ where $k \geqslant 0$ is the largest integer such that $g(\overline{x})^{p^k}$ divides $f(\overline{x})$, in the case that $\operatorname{char} \mathbb{F} = p > 0$.*

2.2 Combinatorial Designs

We will make use of the designs of Nisan and Wigderson [31], specifically as they are used by Kabanets and Impagliazzo [21] to prove hardness-randomness tradeoffs for polynomial identity testing. Nisan and Wigderson [31] gave two constructions of designs: one via

Reed-Solomon codes, and one via a greedy algorithm. We first quote their construction using Reed-Solomon codes, which was also recently described in work by Kumar, Saptharishi, and Tengse [27].

▶ **Lemma 2.9** ([31], see also [27]). *Let $c \geqslant 2$ be a positive integer, and let $n, m, \ell, r \in \mathbb{N}$ be such that (i) $\ell = m^c$, (ii) $r \leqslant m$, (iii) m is a prime power, and (iv) $n \leqslant m^{(c-1)r}$. Then there is a collection of sets $S_1, \ldots, S_n \subseteq [\ell]$ such that*

- *for each $i \in [n]$, we have $|S_i| = m$; and*
- *for all distinct $i, j \in [n]$, we have $|S_i \cap S_j| \leqslant r$.*

Additionally, such a family can be deterministically constructed in $\mathrm{poly}(n)$ *time.*

We now cite the designs obtained by Nisan and Wigderson [31] via a greedy algorithm. In the regime where $m = O(\log n)$, this improves on the previous construction by taking the size ℓ of the ground set to be $O(\log n)$ as opposed to $O(\log^2 n)$.

▶ **Lemma 2.10** ([31]). *Let n and m be integers such that $n < 2^m$. There exists a family of sets $S_1, \ldots, S_n \subseteq [\ell]$ such that*

1. *$\ell = O(m^2/\log(n))$,*
2. *for each $i \in [n]$, we have $|S_i| = m$; and*
3. *for all distinct $i, j \in [n]$, we have $|S_i \cap S_j| \leqslant \log(n)$.*

Such a family of sets can be deterministically constructed in time $\mathrm{poly}(n, 2^\ell)$.

In extending the analysis of the Kabanets-Impagliazzo generator to low characteristic fields, we will make use of Lemma 2.10. Our use of Lemma 2.9 will arise when we combine the hardness versus randomness paradigm with the bootstrapping phenomenon. In that setting, we will apply Lemma 2.9 with $c = O(1)$ and $r = O(1)$. Compared to Lemma 2.10, this yields sets with much smaller intersection size, though the number of sets is only $m^{O(1)}$ as opposed to 2^m.

2.3 Field Theory

To cleanly state some of our results, we need the notion of a perfect field. Namely, given a circuit Φ which computes $f(\overline{x})^p \in \mathbb{F}[\overline{x}]$, we will construct in Section 3 a circuit Ψ which computes $f(\overline{x})$. This construction takes p^{th} roots of field elements $\alpha \in \mathbb{F}$, which are not always guaranteed to exist in $\mathbb{F}$. To ensure Ψ is defined over the base field $\mathbb{F}$, we require that $\mathbb{F}$ is closed under taking p^{th} roots, which is equivalent to requiring that $\mathbb{F}$ is perfect.

▶ **Definition 2.11.** *A field $\mathbb{F}$ is called* perfect *if either $\mathbb{F}$ has characteristic 0 or $\mathbb{F}$ has characteristic $p > 0$ and the map $\alpha \mapsto \alpha^p$ is an automorphism of $\mathbb{F}$. If $\mathbb{F}$ has characteristic $p > 0$, then the* perfect closure *of $\mathbb{F}$, denoted $\mathbb{F}^{p^{-\infty}}$, is the smallest field containing $\mathbb{F}$ which is closed under taking p^{th} roots.*

It is a basic fact that perfect closures exist.

▶ **Fact 2.12.** *Every field $\mathbb{F}$ of characteristic $p > 0$ has a perfect closure $\mathbb{F}^{p^{-\infty}}$.*

Informally, one can prove this by adjoining "enough" p^{th} roots to the field $\mathbb{F}$. That is, for each $\alpha \in \mathbb{F}$, we introduce a countable collection of new field elements denoted by (α, n) for $n \in \mathbb{N}$, where the element (α, n) is meant to represent $\alpha^{p^{-n}}$. We then take a quotient by a suitable equivalence relation; for example, if $\alpha^p = \beta$, then we regard (α, n) and $(\beta, n+1)$ as equivalent for all $n \in \mathbb{N}$. One must then verify that the resulting object is in fact a field and is (up to isomorphism) the perfect closure of $\mathbb{F}$. More formally, the perfect closure can be constructed as the *direct limit* of a particular *direct system* of fields. We refer the reader to Bourbaki [7, Chapter 5, §1] for the details of this construction.

Examples of perfect fields of positive characteristic include all finite fields and all algebraically closed fields of positive characteristic. A non-example is given by $\mathbb{F}_{p^m}(\overline{x})$, the field of rational functions in n variables with coefficients in $\mathbb{F}_{p^m}$, where $\mathbb{F}_{p^m}$ is the finite field of size p^m. The field $\mathbb{F}_{p^m}(\overline{x})$ fails to be perfect due to the fact that $x_1^{1/p} \notin \mathbb{F}_{p^m}(\overline{x})$, so x_1 is not in the image of the map $\alpha \mapsto \alpha^p$.

For more details on perfect fields, we refer the reader to any text on field theory, e.g., Roman [33, Chapter 3].

3 p^{th} Roots of Algebraic Computation

Suppose $\mathbb{F}$ is a field of characteristic $p > 0$ and Φ is a circuit which computes $f(\overline{x})^p$ for a polynomial $f(\overline{x})$. If we want to obtain a circuit which computes $f(\overline{x})$, then Theorem 2.8 does not suffice. In this section, we will describe a simple transformation of Φ which yields a circuit computing $f(\overline{x})$. This is the main technical step that will allow us to obtain hardness-randomness tradeoffs over fields of low characteristic.

In general, this transformation will incur an exponential blow-up in the size of Φ. If the original circuit computes a polynomial on n variables, then the new circuit we build will be larger in size by a factor of about p^{2n}. In particular, if our input is a circuit on a constant number of variables, then we only increase the size of the circuit by a constant factor. The fact that this transformation is efficient in the constant-variate regime is exactly the reason we need to use hardness of constant-variate families of polynomials as opposed to a family of hard multilinear polynomials.

Before describing the construction for circuits on an arbitrary number of variables, we first examine the case of univariate polynomials. Let $\mathbb{F}$ be a field of characteristic $p > 0$ and let $f(x) \in \mathbb{F}[x]$ be a univariate polynomial. We start by grouping the monomials of f by their degree modulo p, which allows us to write

$$f(x) = \sum_{i=0}^{p-1} \widetilde{f}_i(x) x^i,$$

where each $\widetilde{f}_i(x)$ is a univariate polynomial in x which is only supported on p^{th} powers of x. That is, the term $\widetilde{f}_i(x)x^i$ corresponds exactly to the monomials in $f(x)$ whose degree in x is congruent to i modulo p. Recall that over a field of characteristic $p > 0$, we have the identity $(a+b)^p = a^p + b^p$. Since $\widetilde{f}_i(x)$ is a sum of p^{th} powers of x, we can write

$$\widetilde{f}_i(x) = \sum_{j=0}^{d_i} \alpha_{i,j} x^{jp} = \left(\sum_{j=0}^{d_i} \alpha_{i,j}^{1/p} x^j \right)^p.$$

This expresses $\widetilde{f}_i(x)$ as a p^{th} power of the polynomial $f_i(x) \coloneqq \sum_{j=0}^{d_i} \alpha_{i,j}^{1/p} x^j$. In general, f_i may not be well-defined over $\mathbb{F}$, as the coefficients $\alpha_{i,j}^{1/p}$ may not exist in $\mathbb{F}$. However, $\alpha_{i,j}^{1/p} \in \mathbb{F}^{p^{-\infty}}$, the perfect closure of $\mathbb{F}$, so f_i is well-defined over $\mathbb{F}^{p^{-\infty}}$.

With this, we can write

$$f(x) = \sum_{i=0}^{p-1} f_i(x)^p x^i.$$

We refer to such an expression as the mod-p decomposition of f. This motivates the following definition, which generalizes this decomposition to the case of multivariate polynomials.

▶ **Definition 3.1.** *Let $f(\overline{x}) \in \mathbb{F}[\overline{x}]$. The* mod-$p$ decomposition of $f(\overline{x})$ *is the collection of polynomials $\{f_{\overline{a}}(\overline{x}) : \overline{a} \in [\![p]\!]^n\}$ such that*

$$f(\overline{x}) = \sum_{\overline{a} \in [\![p]\!]^n} f_{\overline{a}}(\overline{x})^p \overline{x}^{\overline{a}}.$$

Over a perfect field $\mathbb{F}$ of characteristic $p > 0$, the existence of the mod-p decomposition follows from the fact that any polynomial of the form $\sum_{\overline{a}} \alpha_{\overline{a}} \overline{x}^{p \cdot \overline{a}}$ has a p^{th} root, given by $\sum_{\overline{a}} \alpha_{\overline{a}}^{1/p} \overline{x}^{\overline{a}}$. Here, we use the fact that $\mathbb{F}$ is perfect to guarantee the constants $\alpha_{\overline{a}}^{1/p}$ exist in $\mathbb{F}$. Uniqueness of the decomposition follows from the fact that the monomials $\{\overline{x}^{\overline{a}} : \overline{a} \in \mathbb{N}^n\}$ form a basis for $\mathbb{F}[\overline{x}]$. We record this observation as a lemma.

▶ **Lemma 3.2.** *Let $\mathbb{F}$ be a field of characteristic $p > 0$ and let $f, g \in \mathbb{F}[\overline{x}]$. Let $\{f_{\overline{a}} : \overline{a} \in [\![p]\!]^n\}$ and $\{g_{\overline{a}} : \overline{a} \in [\![p]\!]^n\}$ be the mod-p decompositions of f and g, respectively. Then $f = g$ if and only if $f_{\overline{a}} = g_{\overline{a}}$ for all $\overline{a} \in [\![p]\!]^n$.*

The utility of the mod-p decomposition becomes apparent when $f(\overline{x})$ is itself a p^{th} power. In this case, f itself is a sum of p^{th} powers of monomials in the variables $x_1, \ldots, x_n$, so we have $f(\overline{x}) = f_{\overline{0}}(\overline{x})^p$. Given a circuit Φ which computes f, suppose we could transform Φ into a new circuit Ψ which computes the mod-p decomposition of f. Then to compute $f(\overline{x})^{1/p}$, we simply construct the circuit Ψ and set $f_{\overline{0}}(\overline{x}) = f(\overline{x})^{1/p}$ to be the output.

Before continuing on, we record a straightforward lemma about how the mod-p decomposition behaves with respect to addition and multiplication.

▶ **Lemma 3.3.** *Let $\mathbb{F}$ be a perfect field of characteristic $p > 0$. Let $f, g \in \mathbb{F}[\overline{x}]$, and let $\{f_{\overline{a}} : \overline{a} \in [\![p]\!]^n\}$ and $\{g_{\overline{a}} : \overline{a} \in [\![p]\!]^n\}$ be the mod-p decompositions of f and g, respectively. Let $h = \alpha f + \beta g$ and $q = \gamma f g$ for $\alpha, \beta, \gamma \in \mathbb{F}$. Let $\{h_{\overline{a}} : \overline{a} \in [\![p]\!]^n\}$ and $\{q_{\overline{a}} : \overline{a} \in [\![p]\!]^n\}$ be the mod-p decompositions of h and q. Then for all $\overline{a} \in [\![p]\!]^n$, we have*

$$h_{\overline{a}} = \alpha^{1/p} f_{\overline{a}} + \beta^{1/p} g_{\overline{a}}$$

and

$$q_{\overline{a}} = \gamma^{1/p} \sum_{\substack{\overline{b}, \overline{c} \in [\![p]\!]^n \\ \overline{b} + \overline{c} \equiv \overline{a} \bmod p}} f_{\overline{b}} g_{\overline{c}} \overline{x}^{\frac{\overline{b} + \overline{c} - \overline{a}}{p}},$$

where the sum and congruence $\overline{b} + \overline{c} \equiv \overline{a} \bmod p$ are performed component-wise.

Proof. By expanding the equality $h = \alpha f + \beta g$ in the mod-p decomposition and using the fact that $(a + b)^p = a^p + b^p$, we obtain

$$\begin{aligned} \sum_{\overline{a} \in [\![p]\!]^n} h_{\overline{a}}(\overline{x})^p \overline{x}^{\overline{a}} &= \alpha \sum_{\overline{a} \in [\![p]\!]^n} f_{\overline{a}}(\overline{x})^p \overline{x}^{\overline{a}} + \beta \sum_{\overline{a} \in [\![p]\!]^n} g_{\overline{a}}(\overline{x})^p \overline{x}^{\overline{a}} \\ &= \sum_{\overline{a} \in [\![p]\!]^n} (\alpha^{1/p} f_{\overline{a}}(\overline{x}) + \beta^{1/p} g_{\overline{a}}(\overline{x}))^p \overline{x}^{\overline{a}}. \end{aligned}$$

Lemma 3.2 implies that $h_{\overline{a}} = \alpha^{1/p} f_{\overline{a}} + \beta^{1/p} g_{\overline{a}}$ as claimed.

For $q(\overline{x})$, we again expand the equality $q = \gamma f g$ in the mod-p decomposition to obtain

$$\begin{aligned}\sum_{\overline{a}\in[\![p]\!]^n} q_{\overline{a}}(\overline{x})^p \overline{x}^{\overline{a}} &= \gamma \left(\sum_{\overline{a}\in[\![p]\!]^n} f_{\overline{a}}(\overline{x})^p \overline{x}^{\overline{a}}\right)\left(\sum_{\overline{a}\in[\![p]\!]^n} g_{\overline{a}}(\overline{x})^p \overline{x}^{\overline{a}}\right) \\ &= \gamma \sum_{\overline{b},\overline{c}\in[\![p]\!]^n} f_{\overline{b}}(\overline{x})^p g_{\overline{c}}(\overline{x})^p \overline{x}^{\overline{b}+\overline{c}} \\ &= \sum_{\overline{a}\in[\![p]\!]^n} \left(\gamma^{1/p} \sum_{\substack{\overline{b},\overline{c}\in[\![p]\!]^n \\ \overline{b}+\overline{c}\equiv\overline{a} \bmod p}} f_{\overline{b}}(\overline{x}) g_{\overline{c}}(\overline{x}) \overline{x}^{\frac{\overline{b}+\overline{c}-\overline{a}}{p}}\right)^p \overline{x}^{\overline{a}}.\end{aligned}$$

Once more, Lemma 3.2 implies that

$$q_{\overline{a}} = \gamma^{1/p} \sum_{\substack{\overline{b},\overline{c}\in[\![p]\!]^n \\ \overline{b}+\overline{c}\equiv\overline{a} \bmod p}} f_{\overline{b}} g_{\overline{c}} \overline{x}^{\frac{\overline{b}+\overline{c}-\overline{a}}{p}}$$

as claimed. ◀

3.1 Circuits

We start by implementing the strategy outlined above in the case of algebraic circuits. Throughout this and subsequent sections, Φ and Ψ will denote algebraic circuits, formulae, or branching programs, and v, u, and w will denote gates in these circuits. We will frequently refer to the polynomial computed at a gate v, which we denote by $\hat{v}$. For $\overline{a} \in [\![p]\!]^n$, we write $\hat{v}_{\overline{a}}$ for the part of the mod-p decomposition of $\hat{v}$ indexed by $\overline{a}$.

▶ **Lemma 3.4.** *Let $\mathbb{F}$ be a field of characteristic $p > 0$. Let Φ be an algebraic circuit of size s which computes a polynomial $f(\overline{x}) \in \mathbb{F}[\overline{x}]$ and let $\{f_{\overline{a}} : \overline{a} \in [\![p]\!]^n\}$ be the mod-p decomposition of f. Then there is a circuit Ψ of size $3sp^{2n} + 2^n$ which simultaneously computes $\{f_{\overline{a}} : \overline{a} \in [\![p]\!]^n\}$ over $\mathbb{F}^{p^{-\infty}}$, the perfect closure of $\mathbb{F}$.*

Proof. To construct the desired circuit Ψ, we will split each gate v of Φ into pieces $\{(v, \overline{a}) : \overline{a} \in [\![p]\!]^n\}$ and wire Ψ so that $(v, \overline{a})$ computes $\hat{v}_{\overline{a}}$. As Φ computes $f(\overline{x})$, this implies that Ψ will contain gates computing $f_{\overline{a}}(\overline{x})$ for all $\overline{a} \in [\![p]\!]^n$. To wire each gate $(v, \overline{a})$ in Ψ, we consider the type of the gate v in Φ.

- First, suppose v is an input gate in Φ labeled by a constant $\alpha \in \mathbb{F}$. In this case, we set $(v, \overline{0}) = \alpha^{1/p}$ and $(v, \overline{a}) = 0$ for $\overline{a} \neq \overline{0}$. By definition, $\mathbb{F}^{p^{-\infty}}$ contains $\alpha^{1/p}$, so this is valid over $\mathbb{F}^{p^{-\infty}}$.
  It follows from the definition of $\hat{v}_{\overline{a}}$ that $(v, \overline{a})$ correctly computes $\hat{v}_{\overline{a}}$.
- If v is an input gate labeled by the variable x_i, let $\overline{e}_i$ denote the vector with a 1 in the i^{th} slot and zero elsewhere. We set $(v, \overline{e}_i) = 1$ and $(v, \overline{a}) = 0$ for $\overline{a} \neq \overline{e}_i$.
  Again, it follows immediately from the definition of $\hat{v}_{\overline{a}}$ that $(v, \overline{a})$ correctly computes $\hat{v}_{\overline{a}}$.
- Suppose now that v is an addition gate in Φ with children u and w with incoming edges labeled α_u and α_w. For each $\overline{a} \in [\![p]\!]^p$, we set $(v, \overline{a}) = \alpha_u^{1/p} \cdot (u, \overline{a}) + \alpha_w^{1/p} \cdot (w, \overline{a})$.
  By induction, $(u, \overline{a})$ and $(w, \overline{a})$ correctly compute $\hat{u}_{\overline{a}}$ and $\hat{w}_{\overline{a}}$, respectively. Lemma 3.3 then implies that $(v, \overline{a})$ correctly computes $\hat{v}_{\overline{a}}$.

- Finally, we consider the case where v is a multiplication gate in Φ with children u and w with incoming edges labeled α_u and α_w. For $\overline{a} \in [\![p]\!]^n$, we set

$$(v, \overline{a}) = \alpha_u^{1/p} \alpha_w^{1/p} \sum_{\substack{\overline{b}, \overline{c} \in [\![p]\!]^n \\ \overline{b} + \overline{c} \equiv \overline{a} \pmod{p}}} (u, \overline{b}) \cdot (w, \overline{c}) \cdot \overline{x}^{\frac{\overline{b} + \overline{c} - \overline{a}}{p}},$$

  where vector addition and congruence of vectors is performed coordinate-wise. Note that since $\overline{b} + \overline{c} \equiv \overline{a} \bmod p$, the vector $\frac{1}{p}(\overline{b} + \overline{c} - \overline{a})$ is in fact an integer vector. Moreover, since $\overline{b} + \overline{c} \in \{0, \ldots, 2(p-1)\}^n$, it follows that $\overline{b} + \overline{c} - \overline{a} \in \{0, p\}^n$, so $\frac{1}{p}(\overline{b} + \overline{c} - \overline{a}) \in \{0, 1\}^n$ is a zero-one vector.
  Via induction, $(u, \overline{b})$ and $(w, \overline{c})$ correctly compute $\hat{u}_{\overline{b}}$ and $\hat{w}_{\overline{c}}$, respectively. From this and Lemma 3.3, it follows that $(v, \overline{a})$ correctly computes $\hat{v}_{\overline{a}}$.

As previously remarked, since Φ computes $f(\overline{x})$, for every $\overline{a} \in [\![p]\!]^n$ there is a gate in Ψ which computes $f_{\overline{a}}(\overline{x})$, so Ψ correctly computes all components of the mod-p decomposition of f. It remains to bound the size of Ψ.

For every gate in Φ, we construct p^n gates of the form $(v, \overline{a})$ in Ψ. In the case that v is a multiplication gate, we need extra intermediate hardware to compute the summation $(v, \overline{a}) = \sum_{\overline{b} + \overline{c} \equiv \overline{a} \pmod{p}} (u, \overline{b}) \cdot (w, \overline{c}) \cdot \overline{x}^{\frac{\overline{b} + \overline{c} - \overline{a}}{p}}$. This can be done with p^n summation gates and $2p^n$ multiplication gates. We also need 2^n gates to compute the products $\overline{x}^{\overline{e}}$ for $\overline{e} \in \{0, 1\}^n$. Since Ψ is a circuit, we only need to pay for these gates once, as we can reuse them for all the multiplication computations. In total, each multiplication gate incurs an extra cost of $3p^n$ gates.

This implies each gate in Φ gives rise to at most $3p^{2n}$ gates in Ψ. As there are s gates in Φ, there are at most $3sp^{2n} + 2^n$ gates in Ψ. ◀

▶ Remark 3.5. In the above construction, rather than using the perfect closure, the resulting circuit can be defined over an extension $\mathbb{K} \supseteq \mathbb{F}$ of finite degree. This can be done by adjoining to $\mathbb{F}$ all p^{th} roots of constants which appear in Φ. The degree of this extension may be exponential in s in the worst case.

We can now use the construction of Lemma 3.4 to take p^{th} roots of circuits which compute a p^{th} power over a field of characteristic p.

▶ **Corollary 3.6.** *Let $\mathbb{F}$ be a field of characteristic $p > 0$. Let Φ be an algebraic circuit of size s which computes a polynomial $f(\overline{x})^p \in \mathbb{F}[\overline{x}]$. Then there is a circuit Ψ of size $3sp^{2n} + 2^n$ which computes $f(\overline{x})$ over $\mathbb{F}^{p^{-\infty}}$, the perfect closure of $\mathbb{F}$.*

Proof. By Lemma 3.4, there is a circuit Ψ of the claimed size which computes $(f(\overline{x})^p)_{\overline{0}}$. It follows from the definition of the mod-p decomposition that $f(\overline{x}) = (f(\overline{x})^p)_{\overline{0}}$, so Ψ computes $f(\overline{x})$ as desired. ◀

▶ Remark 3.7. If $n = O(\log_p s)$, then Corollary 3.6 shows that if f^p is computable in size s, then f is computable in size $s^{O(1)}$. While the log-variate regime may appear as a somewhat artificial intermediary between the constant-variate and full multivariate regimes, it is a meaningful setting to study due to various corollaries of the bootstrapping results. For example, Forbes, Ghosh, and Saxena [14] recently studied the problem of designing explicit hitting sets for log-variate depth-three diagonal circuits.

3.2 Formulae

It is natural to ask if the mod-p decomposition allows us to efficiently take p^{th} roots in other models of algebraic computation. We address this question first in the case of algebraic formulae, and subsequently for algebraic branching programs. For the reader who is solely interested in the application of the mod-p decomposition and Corollary 3.6 to hardness-randomness tradeoffs, it is safe to skip ahead to Section 4. Before continuing on, we make an important remark regarding formulae and branching programs for univariate polynomials.

▶ Remark 3.8. In the univariate regime, our results (as stated) for formulae and branching programs are not as meaningful as the result for circuits. A formula or ABP of size s can only compute a polynomial of degree $d \leqslant s$, so any formula or ABP computing a degree d univariate polynomial must have size at least d. For univariate polynomials, Horner's rule supplies a matching $O(d)$ upper bound. Thus, the p^{th} root of a univariate polynomial which has complexity s can be computed by a device of size s/p, which is much stronger than what we will obtain in Corollary 3.10 and Corollary 3.12.

However, if one modifies the model of formulae (or branching programs) to allow leaves (or edges) labeled by a power of a variable x_i^j, then the trivial $\Omega(d)$ lower bound no longer holds. Our techniques can be adapted to this stronger model with little modification, where the upper bounds we obtain are less trivial.

We now show how one can compute the mod-p decomposition of an algebraic formula. We essentially do this by applying the transformation of Lemma 3.4 and arguing that we can convert the resulting circuit into a formula without increasing its size too much. To do this, we need some additional bookkeeping to ensure that the underlying graph of the resulting computation is a tree. We borrow this style of bookkeeping from Raz [32], who used it for improved homogenization and multilinearization of formulae. Alternatively, one can use the fact that formulae of size s can be rebalanced to have depth $O(\log s)$ and then analyze the increase in depth incurred in the proof of Lemma 3.4.

▶ **Lemma 3.9.** *Let $\mathbb{F}$ be a field of characteristic $p > 0$. Let Φ be an algebraic formula of size s and product depth d which computes a polynomial $f(\overline{x}) \in \mathbb{F}[\overline{x}]$ and let $\{f_{\overline{a}} : \overline{a} \in [\![p]\!]^n\}$ be the mod-p decomposition of f. Then there is a formula Ψ of size $3snp^{n(d+3)}$ and product depth $d + \lceil \log n \rceil$ which simultaneously computes $\{f_{\overline{a}} : \overline{a} \in [\![p]\!]^n\}$ over $\mathbb{F}^{p^{-\infty}}$, the perfect closure of $\mathbb{F}$.*

Proof. As in Lemma 3.4, we will split each gate v of Φ into pieces which compute components of the mod-p decomposition of $\hat{v}$. However, we will need a much larger number of copies of v to ensure that the resulting circuit Ψ is in fact a formula.

We first set up some notation, borrowing heavily from Raz [32]. For a gate v in Φ, let $\text{path}(v)$ denote the set of all vertices on the path from v to the root of Φ, including v itself. Let N_v denote the set of all functions $T : \text{path}(v) \to [\![p]\!]^n$ such that for all $u, w \in \text{path}(v)$ where u is a sum gate with child w, we have $T(u) = T(w)$. Informally, the map T encodes the progression of types in the mod-p decomposition seen as the computation progresses through the formula.

For each gate v in Φ, we create a collection of gates $\{(v, \overline{a}, T) : \overline{a} \in [\![p]\!]^n, T \in N_v, T(v) = \overline{a}\}$. We will wire the gates of Ψ so that $(v, \overline{a}, T)$ computes $\hat{v}_{\overline{a}}$. As before, to wire the gates of Ψ correctly, we consider what type of gate v is in Φ. The construction only differs meaningfully from that of Lemma 3.4 in the case of multiplication gates.

- If v is an input gate in Φ labeled by $\alpha \in \mathbb{F}$, then we set $(v, \overline{0}, T) = \alpha^{1/p}$ and $(v, \overline{a}, T) = 0$ for $\overline{a} \neq \overline{0}$. As $\alpha^{1/p} \in \mathbb{F}^{p^{-\infty}}$, this produces a valid circuit over $\mathbb{F}^{p^{-\infty}}$.

  It is immediate from the definition that $(v, \overline{a}, T)$ correctly computes $\hat{v}_{\overline{a}}$.

- If v is an input gate labeled by the variable x_i, let $\overline{e}_i$ denote the vector with a 1 in the i^{th} slot and zero elsewhere. We set $(v, \overline{e}_i, T) = 1$ and $(v, \overline{a}, T) = 0$ for $\overline{a} \neq \overline{e}_i$.
  Once more, it is an immediate consequence of the definition that $(v, \overline{a}, T)$ correctly computes $\hat{v}_{\overline{a}}$.
- Suppose now that v is an addition gate with children u and w with incoming edges labeled α_u and α_w. For each $\overline{a} \in \{0, \ldots, p-1\}^n$ and $T \in N_v$, we set $(v, \overline{a}, T) = \alpha_u^{1/p} \cdot (u, \overline{a}, T_u) + \alpha_w^{1/p} \cdot (w, \overline{a}, T_w)$, where $T_u \in N_u$ and $T_w \in N_w$ extend T and satisfy $T(v) = T_u(u) = T_w(w)$.
  By induction, $(u, \overline{a}, T_u)$ and $(w, \overline{a}, T_w)$ correctly compute $\hat{u}_{\overline{a}}$ and $\hat{w}_{\overline{a}}$, respectively. By Lemma 3.3, it follows that $(v, \overline{a}, T)$ correctly computes $\hat{v}_{\overline{a}}$.
- Finally, consider the case when v is a multiplication gate with children u and w with incoming edges labeled α_u and α_w. We set

$$(v, \overline{a}, T) = \alpha_u^{1/p} \alpha_w^{1/p} \sum_{\overline{b}+\overline{c} \equiv \overline{a} \pmod{p}} (u, \overline{b}, T_{u,\overline{b}}) \cdot (w, \overline{c}, T_{w,\overline{c}}) \cdot \overline{x}^{\frac{\overline{b}+\overline{c}-\overline{a}}{p}},$$

  where $T_{u,\overline{b}}$ (respectively $T_{w,\overline{c}}$) extends T and satisfies $T_{u,\overline{b}}(u) = \overline{b}$ (respectively $T_{w,\overline{c}}(w) = \overline{c}$).
  By induction, $(u, \overline{b}, T_{u,\overline{b}})$ and $(w, \overline{c}, T_{w,\overline{c}})$ compute $\hat{u}_{\overline{b}}$ and $\hat{w}_{\overline{c}}$, respectively. Lemma 3.3 implies that $(v, \overline{a}, T)$ correctly computes $\hat{v}_{\overline{a}}$.

By construction, Ψ correctly computes $\{f_{\overline{a}} : \overline{a} \in [\![p]\!]^n\}$. It remains to bound the size and product depth of Ψ and show that Ψ is indeed a formula.

Each gate v in Φ yields $p^n |N_v|$ gates of the form $(v, \overline{a}, T)$ in Ψ. If v is a multiplication gate with children u and w, we need to implement the sum over the children $(u, \overline{b}, T_u)$ and $(w, \overline{c}, T_w)$. For a given $\overline{e} \in \{0,1\}^n$, we can compute $\overline{x}^{\overline{e}}$ using a subformula of size at most n. To compute $(v, \overline{a}, T)$, we need p^n summation gates and $2p^n$ multiplication gates in addition to the gates computing $(u, \overline{b}, T_u)$, $(w, \overline{c}, T_w)$, and $\overline{x}^{\overline{e}}$. This implies that we can compute $(v, \overline{a}, T)$ using at most $3np^n$ extra gates. Thus, for every gate v in Φ, we create at most $3np^{2n}|N_v|$ gates in Ψ.

To bound the size of N_v, note that a function $T \in N_v$ can only change values along path(v) at multiplication gates. Since there are at most d multiplication gates along path(v), we can specify T by a $(d+1)$-tuple of elements of $[\![p]\!]^n$, corresponding to the values taken by T between successive multiplication gates. This implies $|N_v| \leqslant p^{n(d+1)}$. Thus Ψ contains at most $3snp^{n(d+3)}$ gates.

It follows from the definition of Ψ that the product depth of Ψ is $d + \lceil \log n \rceil$, as the number of product gates on any path from a leaf to the root increases by at most an additive $\lceil \log n \rceil$. This arises from the need to implement a product of the form $\overline{x}^{\overline{e}}$ at gates of Ψ which correspond to multiplication gates in Φ. As we need to compute a product of this form at most once along every path from the root to a leaf, we only incur an additive $\lceil \log n \rceil$ increase in product depth as opposed to a multiplicative increase.

To see that Ψ is a formula, consider the edges leaving the gate $(u, \overline{a}, T)$. Let v denote the parent of u in Ψ. If v is an addition gate, then only $(v, \overline{a}, T_v)$ receives an edge from $(u, \overline{a}, T)$ where $T_v \in N_v$ agrees with T on path(v). If v is a multiplication gate, then only $(v, T(v), T_v)$ receives an edge from $(u, \overline{a}, T)$ where $T_v \in N_v$ agrees with T on path(v). In both cases, the fan-out of the gate u is 1, so Ψ is in fact a formula. ◀

As with circuits, we can use Lemma 3.9 to compute p^{th} roots of formulae which compute a p^{th} power over a field of characteristic $p > 0$.

▶ **Corollary 3.10.** *Let $\mathbb{F}$ be a field of characteristic $p > 0$. Let Φ be an algebraic formula of size s and product depth d which computes a polynomial $f(\overline{x})^p \in \mathbb{F}[\overline{x}]$. Then there is a formula Ψ of size $3snp^{n(d+3)}$ and product depth $d + \lceil \log n \rceil$ which computes $f(\overline{x})$ over $\mathbb{F}^{p^{-\infty}}$, the perfect closure of $\mathbb{F}$.*

Proof. Analogous to the proof of Corollary 3.6. ◀

3.3 Algebraic Branching Programs

We now consider the task of taking p^{th} roots of algebraic branching programs. We consider the model of branching programs where edges may only be labeled by a constant $\alpha \in \mathbb{F}$ or a multiple of a variable αx_i. Some authors allow the edges of a branching program to be labeled by an affine form $\ell(\overline{x}) = \alpha_0 + \sum_{i=1}^{n} \alpha_i x_i$. Such a branching program can be converted to one whose edges are labeled by field constants or multiples of a variable. This transformation increases the number of vertices by a factor of $O(n)$, which is small compared to the increase in size we will incur by taking a p^{th} root. We begin by computing the mod-p decomposition of an algebraic branching program.

▶ **Lemma 3.11.** *Let $\mathbb{F}$ be a field of characteristic $p > 0$. Let Φ be an algebraic branching program on s vertices with edges labeled by variables or field constants which computes a polynomial $f(\overline{x}) \in \mathbb{F}[\overline{x}]$ and let $\{f_{\overline{a}} : \overline{a} \in [\![p]\!]^n\}$ be the mod-p decomposition of f. Then there is an algebraic branching program Ψ on sp^n vertices which simultaneously computes $\{f_{\overline{a}} : \overline{a} \in [\![p]\!]^n\}$ over $\mathbb{F}^{p^{-\infty}}$, the perfect closure of $\mathbb{F}$.*

Proof. For each node v in Φ, we create a collection of nodes $\{(v, \overline{a}) : \overline{a} \in [\![p]\!]^n\}$ in Ψ. We will wire the nodes of Ψ so that $(v, \overline{a})$ computes $\hat{v}_{\overline{a}}$.

For a pair of vertices u and v, let $\ell(u, v)$ denote the label of the edge between u and v. Let $N^{\text{in}}(v)$ denote the set of vertices w such that the edge (w, v) is present in Φ.

Let u and v be two nodes in Φ and suppose there is an edge from u to v in Φ. We consider two cases, depending on whether this edge is labeled by a constant $\alpha \in \mathbb{F}$ or a multiple of a variable αx_i.

- Suppose the edge from u to v is labeled by $\alpha \in \mathbb{F}$. For all $\overline{a} \in [\![p]\!]^n$, we add an edge between $(u, \overline{a})$ and $(v, \overline{a})$ labeled by $\alpha^{1/p}$. Since $\alpha^{1/p} \in \mathbb{F}^{p^{-\infty}}$, this construction is valid over the perfect closure $\mathbb{F}^{p^{-\infty}}$ of $\mathbb{F}$.
- Suppose the edge from u to v is labeled by αx_i, where $\alpha \in \mathbb{F}$. Denote by $\overline{e}_i$ the vector which has a 1 in the i^{th} slot and zeroes elsewhere. For all $\overline{a} \in [\![p]\!]^n$, we add an edge between $(u, \overline{a})$ and $(v, \overline{a} + \overline{e}_i)$, where the addition $\overline{a} + \overline{e}_i$ is performed modulo p. If $\overline{a}_i < p - 1$, we label this edge with $\alpha^{1/p}$. If $\overline{a}_i = p - 1$, we label this edge with $\alpha^{1/p} x_i$. Again, $\alpha^{1/p} \in \mathbb{F}^{p^{-\infty}}$ by definition, so this construction is valid.

To see that this construction is correct, let v be a node in Φ. By the definition of an algebraic branching program, we have

$$\hat{v} = \sum_{u \in N^{\text{in}}(v)} \ell(u, v) \cdot \hat{u}.$$

Repeatedly applying the addition case of Lemma 3.3 yields, for each $\overline{a} \in [\![p]\!]^n$,

$$\hat{v}_{\overline{a}} = \sum_{u \in N^{\text{in}}(v)} (\ell(u, v) \cdot \hat{u})_{\overline{a}}.$$

If $\ell(u,v) = \alpha \in \mathbb{F}$, then we have $(\ell(u,v) \cdot \hat{u})_{\overline{a}} = \alpha^{1/p} \hat{u}_{\overline{a}}$. If $\ell(u,v) = \alpha x_i$, then if $\overline{a}_i > 0$, we have $(\ell(u,v) \cdot \hat{u})_{\overline{a}} = \alpha^{1/p} \hat{u}_{\overline{a}-\overline{e}_i}$. Otherwise, $\overline{a}_i = 0$, so $(\ell(u,v) \cdot \hat{u})_{\overline{a}} = \alpha^{1/p} \hat{u}_{\overline{a}-\overline{e}_i} x_i$, where the subtraction $\overline{a} - \overline{e}_i$ is done modulo p.

By induction, $(u, \overline{a})$ correctly computes $\hat{u}_{\overline{a}}$. From our construction of Ψ, if (u,v) is an edge in Φ, then $(v, \overline{a})$ has an incoming edge which computes $(\ell(u,v) \cdot \hat{u})_{\overline{a}}$. This implies that $(v, \overline{a})$ computes the polynomial $\sum_{u \in N^{\text{in}}(v)} (\ell(u,v) \cdot \hat{u})_{\overline{a}} = \hat{v}_{\overline{a}}$, which is what we want.

Thus, Ψ simultaneously computes $\{f_{\overline{a}} : \overline{a} \in [\![p]\!]^n\}$. Every node in Φ corresponds to p^n nodes in Ψ. Unlike the cases of circuits and formulae, we do not need extra hardware to implement intermediate calculations, so Ψ consists of sp^n nodes as claimed. ◀

Again, as in the case of circuits and formulae, this immediately yields a way to compute p^{th} roots of algebraic branching programs which compute a p^{th} power over a field of characteristic $p > 0$.

▶ **Corollary 3.12.** *Let $\mathbb{F}$ be a field of characteristic $p > 0$. Let Φ be an algebraic branching program on s vertices with edges labeled by variables or field constants which computes a polynomial $f(\overline{x})^p \in \mathbb{F}[\overline{x}]$. Then there is an algebraic branching program Ψ on sp^n vertices which computes $f(\overline{x})$ over $\mathbb{F}^{p^{-\infty}}$, the perfect closure of $\mathbb{F}$.*

Proof. Analogous to the proof of Corollary 3.6. ◀

4 Extending the Kabanets-Impagliazzo Generator

With our main technical tool in hand, we move on to our first application. The hitting set generator of Kabanets and Impagliazzo [21] was the first to provide hardness-randomness tradeoffs for polynomial identity testing over fields of characteristic zero. Over fields of characteristic $p > 0$, Kabanets and Impagliazzo obtain hardness-randomness tradeoffs under non-standard hardness assumptions. Namely, they require an explicit family of polynomials $\{f_n : n \in \mathbb{N}\}$ such that $f_n^{p^k}$ is hard to compute for $1 \leqslant p^k \leqslant 2^{O(n)}$, though they do not state their results in this way. Rather, they use the assumption of a family of polynomials which are hard to compute as functions, which implies hardness of p^{th} powers over finite fields.

It is more common in algebraic complexity to prove lower bounds on the task of computing polynomials as syntactic objects. Over infinite fields, this is equivalent to computing a polynomial as a function. However, the two notions differ over finite fields. For example, the polynomial $x^2 - x$ is non-zero as a polynomial over $\mathbb{F}_2$, but computes the zero function over $\mathbb{F}_2$. It is interesting to note that examples of functional lower bounds over finite fields are known. The works of Grigoriev and Karpinski [15], Grigoriev and Razborov [16], and Kumar and Saptharishi [25] prove lower bounds against constant-depth circuits over finite fields which functionally compute an explicit polynomial.

In this section, we will extend the Kabanets-Impagliazzo generator to all perfect fields of characteristic $p > 0$ under syntactic hardness assumptions for a single family of polynomials. The perfect fields of characteristic p include all finite fields and all algebraically closed fields of positive characteristic. To do this, we need a stronger (but still syntactic) hardness assumption. In their work, Kabanets and Impagliazzo use the existence of an explicit family of hard multilinear polynomials to derandomize polynomial identity testing. Here, we need lower bounds against an explicit family of constant-variate polynomials of arbitrarily high degree. Such an assumption appears to be stronger than the assumption of a hard family of multilinear polynomials. We discuss the relationship between these hypotheses in more detail in Section 6.

4.1 The Kabanets-Impagliazzo Generator

We first describe the construction of the Kabanets-Impagliazzo generator.

▶ **Construction 4.1** ([21]). *Let n and m be integers satisfying $n < 2^m$. Let $g \in \mathbb{F}[\overline{x}]$ be a polynomial on m variables. Let $S_1, \ldots, S_n \subseteq [\ell]$ be a Nisan-Wigderson design as in Lemma 2.10. The Kabanets-Impagliazzo generator $\mathcal{G}_{\mathrm{KI},g}(\overline{z}) : \mathbb{F}^\ell \to \mathbb{F}^n$ is the polynomial map given by*

$$\mathcal{G}_{\mathrm{KI},g}(\overline{z}) \coloneqq (g(\overline{z}|_{S_1}), \ldots, g(\overline{z}|_{S_n})),$$

where $\overline{z}|_{S_i}$ denotes the restriction of $\overline{z}$ to the variables with indices in S_i.

We now quote the main lemma used by Kabanets and Impagliazzo in the analysis of their generator.

▶ **Lemma 4.2** ([21]). *Let $\mathbb{F}$ be any field and $n, m \in \mathbb{N}$ such that $n < 2^m$. Let $f \in \mathbb{F}[y_1, \ldots, y_n]$ and $g \in \mathbb{F}[x_1, \ldots, x_m]$ be non-zero polynomials of degree d_f and d_g, respectively. Let $f(\overline{y})$ be computable by an algebraic circuit of size s. Let $S \subseteq \mathbb{F}$ be any set of size at least $d_f d_g + 1$ and let $\ell = O(m^2/\log n)$ be as in Lemma 2.10. Let $\mathcal{G}_{\mathrm{KI},g}$ be as in Construction 4.1.*

Suppose that $f(\mathcal{G}_{\mathrm{KI},g}(\overline{\alpha})) = 0$ for all $\overline{\alpha} \in S^\ell$. Then there is an algebraic circuit Φ of size $s' \leqslant \mathrm{poly}(n, m, d_f, d_g, s, (1+\mathrm{ideg}\, g)^{\log n})$ which computes the following. If $\mathbb{F}$ has characteristic zero, then Φ computes $g(\overline{x})$. If $\mathbb{F}$ has characteristic $p > 0$, then Φ computes $g(\overline{x})^{p^k}$ for some $k \in \mathbb{N}$ such that $p^k \leqslant d_f$.

If $f(\mathcal{G}_{\mathrm{KI},g}(\overline{z})) = 0$, then using Lemma 4.2, we can reconstruct a circuit for g using the circuit for f. By taking g from a family of hard polynomials, we obtain a contradiction if there is a small circuit which computes f. This proves that $\mathcal{G}_{\mathrm{KI},g}$ is a hitting set generator for the class of small circuits. The explicitness of $\mathcal{G}_{\mathrm{KI},g}$ follows from the explicitness of the family from which g is taken. The hardness-randomness tradeoffs of Kabanets and Impagliazzo [21] then follow by setting parameters according to the hardness of g.

Over a field of characteristic $p > 0$, Lemma 4.2 provides a circuit computing $g(\overline{x})^{p^k}$. Suppose we are working over $\mathbb{F}_q$, the finite field of $q = p^a$ elements. By taking p^{th} powers of $g(\overline{x})^{p^k}$ if necessary, we can obtain a circuit which computes $g(\overline{x})^{p^{ar}} = g(\overline{x})^{q^r}$ for some $r \in \mathbb{N}$. The map $\alpha \mapsto \alpha^q$ is the identity over $\mathbb{F}_q$, so the circuit which computes $g(\overline{x})^{q^r}$ in fact computes the same function as $g(\overline{x})$. This is why, without further work, we need a polynomial which is hard to compute as a function to obtain hardness-randomness tradeoffs over finite fields.

If we could factor the circuit for $g(\overline{x})^{p^k}$ to obtain a not-too-much-larger circuit for $g(\overline{x})$, then we could derive hardness-randomness tradeoffs from the assumption of an explicit family of multilinear polynomials which are hard to compute. It remains an open problem to show that if $g(\overline{x})^p$ has a small circuit, then $g(\overline{x})$ has a small circuit. However, in the constant-variate regime, Corollary 3.6 resolves this problem in the affirmative. This is the main fact which drives our extension of the Kabanets-Impagliazzo generator.

4.2 Extension to Fields of Low Characteristic

We now show how to use the Kabanets-Impagliazzo generator to obtain hardness-randomness tradeoffs over all perfect fields of characteristic $p > 0$. Recall that $\mathcal{C}_{\mathbb{F}}(s, n, d)$ denotes the set of n-variate degree d polynomials computable by circuits of size at most s.

▶ **Theorem 4.3.** *Let $\mathbb{F}$ be a field of characteristic $p > 0$ and let $c, k \in \mathbb{N}$ be positive constants. Let $\{g_d(\overline{x}) : d \in \mathbb{N}\}$ be a strongly $t(k,d)$-explicit family of k-variate degree d polynomials. Let $s : \mathbb{N} \to \mathbb{N}$ be a function such that g_d cannot be computed by algebraic circuits of size smaller than $s(d)$ over $\mathbb{F}^{p^{-\infty}}$. Then there is a hitting set generator $\mathcal{G} : \mathbb{F}^\ell \to \mathbb{F}^n$ for $\mathcal{C}_\mathbb{F}(n^c, n, n^c)$ which*

1. *is $\left(\mathrm{poly}(n, 2^\ell) + t(k, n^{3ck+\Omega(c)}) \cdot s^{-1}(n^{3ck+\Omega(c)})^{O(k)}\right)$-explicit,*
2. *has seed length $\ell = O\left(\frac{k^2 \log^2(s^{-1}(n^{3ck+O(c)}))}{\log n}\right)$, and*
3. *has degree $O(k \log(s^{-1}(n^{3ck+O(c)})))$.*

Proof. We will obtain our generator by using $\{g_d : d \in \mathbb{N}\}$ to construct a family of hard multilinear polynomials. We then set parameters and instantiate the Kabanets-Impagliazzo generator with this hard multilinear family.

By Lemma 2.6, there is a strongly $t(k,d)$-explicit family of multilinear polynomials $h_d(\overline{y})$ on $m := k(\lfloor \log d \rfloor + 1)$ variables such that any circuit which computes h_d must be of size $s(d) - O(k \log d)$. The construction of h_d also yields the identity

$$g_d(\overline{x}) = h_d(x_1^{2^0}, x_1^{2^1}, \ldots, x_1^{2^{\lfloor \log d \rfloor}}, \ \ldots \ , x_k^{2^0}, x_k^{2^1}, \ldots, x_k^{2^{\lfloor \log d \rfloor}}),$$

which allows us to obtain a circuit for g_d from a circuit for h_d. As h_d is multilinear, we have $\deg(h_d) \leqslant m$ and $\mathrm{ideg}(h_d) = 1$.

Set $d = s^{-1}(n^e)$ for a large enough constant $e \geqslant 1$ to be specified later. Since g_d is a k-variate degree d polynomial, we trivially have $s(d) \leqslant d^{O(k)}$, so $s^{-1}(d) \geqslant d^{\Omega(1/k)}$. This gives us

$$2^m \geqslant d^k = s^{-1}(n^e)^k \geqslant (n^{\Omega(e/k)})^k = n^{\Omega(e)}.$$

Taking e to be large enough guarantees $2^m > n$. Let $S_1, \ldots, S_n \subseteq [\ell]$ be the Nisan-Wigderson design guaranteed by Lemma 2.10. Our generator $\mathcal{G} : \mathbb{F}^\ell \to \mathbb{F}^n$ is given by instantiating the Kabanets-Impagliazzo generator with h_d. That is,

$$\mathcal{G}(\overline{z}) := \mathcal{G}_{\mathrm{KI},h_d}(\overline{z}) = (h_d(\overline{z}|_{S_1}), \ldots, h_d(\overline{z}|_{S_n})).$$

We now verify the claimed properties of $\mathcal{G}$.

Correctness. To see that $\mathcal{G}$ is indeed a hitting set generator for $\mathcal{C}_\mathbb{F}(n^c, n, n^c)$, suppose there is some non-zero $f \in \mathcal{C}_\mathbb{F}(n^c, n, n^c)$ such that $f(\mathcal{G}(\overline{z})) = 0$. Then by Lemma 4.2, there is a circuit of size

$$s' \leqslant \mathrm{poly}(n, m, n^c, 2^{\log n}) \leqslant n^{O(c)}$$

which computes $h_d(\overline{y})^{p^a}$ for $p^a \leqslant \deg(f) \leqslant n^c$. Via the Kronecker substitution $y_{i,j} \mapsto x_i^{2^j}$, we obtain a circuit of size $s' + O(k \log d) \leqslant n^{O(c)}$ which computes $g_d(\overline{x})^{p^a}$. We now apply Corollary 3.6 a total of a times to obtain a circuit which computes $g_d(\overline{x})$ and has size $s'' \leqslant (3 \cdot 2^k \cdot p^{2k})^a n^{O(c)}$. Since $p^a \leqslant n^c$ and $2 \leqslant p$, we obtain $s'' \leqslant n^{3kc+O(c)}$. By setting $e = 3ck + \Theta(c)$ where the hidden constant on the $\Theta(c)$ term is large enough, we obtain a contradiction as follows. By assumption, any circuit which computes g_d must be of size at least $s(d) = n^e$. However, we have a circuit of size $n^{3ck+O(c)} \ll n^e = s(d)$ which computes g_d, a contradiction. Thus, it must be the case that $f(\mathcal{G}(\overline{z})) \neq 0$. Hence $\mathcal{G}$ is a hitting set generator for $\mathcal{C}_\mathbb{F}(n^c, n, n^c)$.

Explicitness. Given a point $\overline{\alpha} \in \mathbb{F}^\ell$, we can evaluate $\mathcal{G}$ as follows. First, we construct the Nisan-Wigderson design $S_1, \ldots, S_n \subseteq [\ell]$ in time $\mathrm{poly}(n, 2^\ell)$. We then compute all $d^{O(k)}$ coefficients of h_d, each in $t(k,d)$ time. Finally, for each $i \in [\ell]$, we evaluate h_d on $\overline{\alpha}|_{S_i}$ in time $d^{O(k)}$. Using the fact that $d = s^{-1}(n^{3ck+O(c)})$, we can evaluate $\mathcal{G}$ in $\mathrm{poly}(n, 2^\ell) + t(k, n^{3ck+O(c)}) \cdot s^{-1}(n^{3ck+O(c)})^{O(k)}$ time as claimed.

Seed length. It follows from Lemma 2.10 that $\mathcal{G}$ has seed length $\ell = O(m^2/\log n) = O\left(\frac{k^2 \log^2 d}{\log n}\right)$. By our choice of $d = s^{-1}(n^{3ck+O(c)})$, we obtain the claimed seed length of $O\left(\frac{k^2 \log^2(s^{-1}(n^{3ck+O(c)}))}{\log n}\right)$.

Degree. By construction, $\mathcal{G}$ is a map of degree $\deg(h_d) \leqslant m = k(\lfloor \log d \rfloor + 1)$. Once more, plugging in our choice of d yields the claimed bound of $O(k \log(s^{-1}(n^{3ck+O(c)})))$. ◀

By applying Lemma 2.3, we obtain the following construction of explicit hitting sets for $\mathcal{C}_{\mathbb{F}}(n^c, n, n^c)$.

▶ **Corollary 4.4.** *Assume the setup of Theorem 4.3. Let T, ℓ, and Δ be the explicitness, seed length, and degree of the generator of Theorem 4.3, respectively. Then there is a hitting set $\mathcal{H}$ for $\mathcal{C}_{\mathbb{F}}(n^c, n, n^c)$ which*

1. *has size $|\mathcal{H}| = (n^c\Delta + 1)^\ell$, and*
2. *has explicitness $|\mathcal{H}| \cdot T = (n^c\Delta + 1)^\ell \cdot T$.*

Proof. This is Lemma 2.3 applied to Theorem 4.3. ◀

We conclude this section with some concrete hardness-randomness tradeoffs obtainable via Theorem 4.3 and Corollary 4.4. Recall that for constant k, a k-variate polynomial of degree d consists of at most $\binom{k+d}{k} \leqslant d^{O(k)}$ monomials. In this regime, a polynomial which is strongly $d^{O(k)}$-explicit is "exponential time explicit," as the description of a single monomial consists of $O(k \log d)$ bits.

▶ **Corollary 4.5.** *Let $\mathbb{F}$ be a field of characteristic $p > 0$. Let $c, k \in \mathbb{N}$ be fixed constants. Let $\{g_d(\overline{x}) : d \in \mathbb{N}\}$ be a strongly $d^{O(k)}$-explicit family of k-variate degree d polynomials which cannot be computed by circuits of size smaller than $s(d)$ over $\mathbb{F}^{p^{-\infty}}$. Then the following results hold regarding hitting sets for $\mathcal{C}_{\mathbb{F}}(n^c, n, n^c)$.*

1. *If $s(d) = \log^{\omega(1)} d$, then there is a $2^{n^{o(1)}}$-explicit hitting set for $\mathcal{C}_{\mathbb{F}}(n^c, n, n^c)$ of size $2^{n^{o(1)}}$.*
2. *If $s(d) = 2^{\log^{\Omega(1)} d}$, then there is a $2^{\log^{O(1)} n}$-explicit hitting set for $\mathcal{C}_{\mathbb{F}}(n^c, n, n^c)$ of size $2^{\log^{O(1)} n}$.*
3. *If $s(d) = d^{\Omega(1)}$, then there is a $n^{O(\log n)}$-explicit hitting set for $\mathcal{C}_{\mathbb{F}}(n^c, n, n^c)$ of size $n^{O(\log n)}$.*

Proof. Each statement follows by setting parameters in Theorem 4.3 and Corollary 4.4 and using the fact that c and k are fixed constants independent of n and d. We omit the straightforward calculations. ◀

5 Bootstrapping from Constant-Variate Hardness

Given that we use the seemingly stronger assumption of constant-variate hardness in our extension of the Kabanets-Impagliazzo generator, one may wonder if we can push the hardness-randomness connection further and obtain a better derandomization of identity testing for $\mathcal{C}_{\mathbb{F}}(n^c, n, n^c)$. Perhaps surprisingly, this is possible by going through the recent development of "bootstrapping" for hitting sets.

5.1 A Non-Trivial Hitting Set from Constant-Variate Hardness

Let n be a constant and let s be arbitrarily large. Suppose we have an explicit, slightly non-trivial hitting set for $\mathcal{C}_{\mathbb{F}}(s, n, s)$. Then we can "bootstrap" the advantage this hitting set has over the trivial one in order to obtain an explicit hitting set of very small size for $\mathcal{C}_{\mathbb{F}}(s, s, s)$. That is, in order to almost completely derandomize polynomial identity testing

for the class of polynomials of polynomial degree computed by polynomial-size circuits, it suffices to find a non-trivial derandomization of polynomial identity testing for circuits on a constant number of variables but of arbitrary size and degree.

We remark that, throughout this section, one should read $\mathcal{C}_{\mathbb{F}}(s,s,s)$ as a stand-in for $\mathcal{C}_{\mathbb{F}}(n^c,n,n^c)$, where c is a fixed constant. This follows by taking $s = n^c$ and noting that $\mathcal{C}_{\mathbb{F}}(n^c,n,n^c) \subseteq \mathcal{C}_{\mathbb{F}}(n^c,n^c,n^c) = \mathcal{C}_{\mathbb{F}}(s,s,s)$. While the following results are stated for $\mathcal{C}_{\mathbb{F}}(s,s,s)$, changing s by at most a polynomial factor will not qualitatively affect the results we obtain.

We now formally state the bootstrapping result. Let $\log^\star s$ denote the iterated logarithm of s. That is,

$$\log^\star s := \begin{cases} 1 + \log^\star(\log s) & s > 1 \\ 0 & s \leqslant 1. \end{cases}$$

This version of the bootstrapping theorem is due to Kumar, Saptharishi, and Tengse [27] and improves upon the initial work of Agrawal, Ghosh, and Saxena [2]. Note that this theorem holds over all fields, including those of positive characteristic.

▶ **Theorem 5.1** ([27]). *Let $\mathbb{F}$ be any field and let $\varepsilon > 0$ and $n \geqslant 2$ be constants. Suppose that for all sufficiently large s, there is an $s^{O(n)}$-explicit hitting set of size $s^{n-\varepsilon}$ for $\mathcal{C}_{\mathbb{F}}(s,n,s)$. Then there is an $s^{\exp\circ\exp(O(\log^\star s))}$-explicit hitting set of size $s^{\exp\circ\exp(O(\log^\star s))}$ for $\mathcal{C}_{\mathbb{F}}(s,s,s)$.*

In this section, we will use Theorem 5.1 to obtain a stronger derandomization of polynomial identity testing over fields of characteristic $p > 0$ under appropriate hardness assumptions. Suppose $\{g_d(\overline{x}) : d \in \mathbb{N}\}$ is a family of strongly $d^{O(k)}$-explicit k-variate degree d polynomials which require algebraic circuits of size $d^{\Omega(k)}$. Using Corollary 4.5, we can obtain a $s^{O(\log s)}$-explicit hitting set for $\mathcal{C}_{\mathbb{F}}(s,s,s)$ of size $s^{O(\log s)}$. By a more careful instantiation of the Kabanets-Impagliazzo generator, we can use the hardness assumption on g_d to design an explicit hitting set which satisfies the hypotheses of Theorem 5.1. This yields an explicit hitting set for $\mathcal{C}_{\mathbb{F}}(s,s,s)$ of size $s^{\exp\circ\exp(O(\log^\star s))}$, which greatly improves upon the size $s^{O(\log s)}$ hitting set of Corollary 4.5.

Our argument also works for fields of characteristic zero, giving us a general theorem which converts near-optimal constant-variate hardness into near-optimal derandomization of polynomial identity testing for $\mathcal{C}_{\mathbb{F}}(s,s,s)$.

First, we need a technical lemma regarding lower bounds against constant-variate polynomials. Roughly, we will show that d^δ lower bounds against degree d constant-variate polynomials can be magnified to d^c lower bounds against constant-variate polynomials for arbitrary $\delta, c > 0$.

▶ **Lemma 5.2.** *Let $\mathbb{F}$ be any field. Let $k \in \mathbb{N}$ and $c, \delta > 0$ be fixed constants. Let $\{g_d(\overline{x}) : d \in \mathbb{N}\}$ be a strongly $d^{O(k)}$-explicit family of k-variate polynomials of degree d. Suppose that for d sufficiently large, g_d cannot be computed by algebraic circuits of size smaller than d^δ over $\mathbb{F}$. Then there is a constant $m \in \mathbb{N}$ and a family $\{h_\Delta(\overline{y}) : \Delta \in \mathbb{N}\}$ of strongly $\Delta^{O(m)}$-explicit m-variate degree Δ polynomials such that for Δ sufficiently large, h_Δ cannot be computed by algebraic circuits of size smaller than Δ^c over $\mathbb{F}$.*

Proof. We follow the approach of Lemma 2.6, but in base $d^{\delta/2c}+1$ as opposed to base 2.

Without loss of generality, assume that $\delta \leqslant 1 \leqslant c$. Let $m := \frac{2ck}{\delta}$ and let $\overline{y} = (y_{1,1}, \ldots, y_{k,2c/\delta})$. Let $\sigma(y_{i,j}) = x_i^{(d^{\delta/2c}+1)^j}$. We will take $h_\Delta(\overline{y})$ to be the polynomial of individual degree $d^{\delta/2c}$ which satisfies the equation $h(\sigma(\overline{y})) = g_d(\overline{x})$. More explicitly, let

$g_d(\overline{x}) = \sum_{\overline{a} \in \mathbb{N}^k} \alpha_{\overline{a}} \overline{x}^{\overline{a}}$ be the expression of g_d as a sum of monomials. Let $\varphi : [\![d^{\delta/2c}+1]\!]^{2c/\delta} \to [\![d+1]\!]$ be the map which takes the base-$(d^{\delta/2c}+1)$ expansion of a number $t \in [\![d+1]\!]$ and returns t. Then we define $h_\Delta(\overline{y})$ as

$$h_\Delta(\overline{y}) = \sum_{A \in [\![d^{\delta/2c}+1]\!]^{k \times 2c/\delta}} \alpha_{\varphi(A_{1,\bullet}),\ldots,\varphi(A_{k,\bullet})} \prod_{i,j \in [\![d^{\delta/2c}+1]\!]} y_{i,j}^{A_{i,j}}.$$

It is clear from the construction of h_Δ that $h_\Delta(\sigma(\overline{y})) = g_d(\overline{x})$. The polynomial h_Δ is of individual degree at most $d^{\delta/2c}$, so $\Delta \coloneqq \deg(h_\Delta)$ can be bounded as

$$\Delta \leqslant m d^{\delta/2c} = \frac{2ckd^{\delta/2c}}{\delta}.$$

Since k and δ are fixed constants, for d large enough, we obtain $\Delta \leqslant d^{2\delta/3c}$.

To show that h_Δ has the claimed hardness, suppose we are given a circuit of size s which computes h_Δ. By repeated squaring, we may compute the map $\sigma(\overline{y})$ using a circuit of size $O(k \log d) = O(m \log \Delta) = O(\log \Delta)$. This yields a circuit of size $s' \leqslant s + O(\log \Delta)$ which computes g_d. By the assumed hardness of g_d, we have $s' \geqslant d^\delta$. Putting things together gives us

$$s \geqslant d^\delta - O(\log \Delta).$$

Since $\Delta \leqslant d^{2\delta/3c}$ for d large enough, we obtain

$$s \geqslant \Delta^{3c/2} - O(\log \Delta).$$

For Δ (and hence d) large enough, we have $s \geqslant \Delta^c$, which yields the desired lower bound on h_Δ.

It remains to verify the explicitness of h_Δ. We can compute a coefficient of h_Δ by computing the corresponding coefficient of g_d, so h_Δ inherits the strong $d^{O(k)}$-explicitness of g_d. We need to show that $d^{O(k)} \leqslant \Delta^{O(m)}$ in order to conclude that h_Δ is strongly $\Delta^{O(m)}$-explicit. By writing h_Δ as a sum of monomials, there is a circuit of size $\Delta^{O(m)}$ which computes h_Δ. Combined with the argument above, this yields a circuit of size $\Delta^{O(m)} + O(\log \Delta) = \Delta^{O(m)}$ which computes g_d. Since any circuit which computes g_d must have size d^δ, we obtain $\Delta^{O(m)} \geqslant d^\delta$. As c, k, δ, and m are all fixed constants, this yields $d^{O(k)} \leqslant \Delta^{O(m)}$ as desired. ◀

Now we are ready to state and prove our hardness-randomness tradeoff.

▶ **Theorem 5.3.** *Let $\mathbb{F}$ be any field and let $k \in \mathbb{N}$ and $\delta > 0$ be fixed constants. Let $\mathbb{K} = \mathbb{F}^{p^{-\infty}}$ if $\operatorname{char} \mathbb{F} = p > 0$ and $\mathbb{K} = \mathbb{F}$ otherwise. Let $\{g_d(\overline{x}) \in \mathbb{F}[\overline{x}] : d \in \mathbb{N}\}$ be a family of strongly $d^{O(k)}$-explicit k-variate degree d polynomials. Suppose that for all d sufficiently large, g_d cannot be computed by algebraic circuits of size smaller than d^δ over $\mathbb{K}$. Then for all sufficiently large s, there is an $s^{\exp \circ \exp(O(\log^\star s))}$-explicit hitting set of size $s^{\exp \circ \exp(O(\log^\star s))}$ for $\mathcal{C}_\mathbb{F}(s, s, s)$.*

Proof. Using Lemma 5.2, we may assume without loss of generality that $\delta \geqslant 30$.

By Theorem 5.1, it suffices to provide an explicit hitting set of size $s^{n-\varepsilon}$ for $\mathcal{C}_\mathbb{F}(s, n, s)$ for constants ε, n and all s sufficiently large. We will instantiate the Kabanets-Impagliazzo generator with g_d as the hard polynomial, using the finer-grained designs of Lemma 2.9.

Let s be given. By adding auxiliary variables if necessary, we may assume that k is a prime power. Note there is always a power of 2 between k and $2k$, so this at most doubles the number of variables in g_d. We set parameters as follows:

- $c := 3$,
- $n := 2k^{c+1} = 2k^4$,
- $r := 2$, and
- $d := s^k$.

By Lemma 2.9, we can construct in $\text{poly}(n)$ time a collection of sets $S_1, \ldots, S_n \subseteq [k^c]$ such that $|S_i| = k$ and $|S_i \cap S_j| \leqslant r$.

Consider the generator $\mathcal{G} : \mathbb{F}^{k^c} \to \mathbb{F}^n$ given by

$$\mathcal{G}(\overline{z}) = (g_d(\overline{z}|_{S_1}), \ldots, g_d(\overline{z}|_{S_n})).$$

By construction, $\mathcal{G}$ has seed length k^c and degree $d = s^k$. Since g_d is strongly $d^{O(k)}$-explicit, we can evaluate $\mathcal{G}$ by constructing the design $S_1, \ldots, S_n$, computing the coefficients of g_d, and evaluating each of the n copies of g_d. Constructing the design takes $n^{O(1)}$ time and computing the coefficients of g_d takes $d^{O(k)}$ time. To evaluate g_d, we use the expression of g_d as a sum of monomials, which requires $d^{O(k)}$ time for each of the n evaluations. In total, we can evaluate $\mathcal{G}$ in time

$$n^{O(1)} \cdot d^{O(k)} = n^{O(1)} \cdot s^{O(k^2)} = n^{O(1)} \cdot s^{O(\sqrt{n})},$$

so $\mathcal{G}$ is $s^{O(\sqrt{n})}$-explicit for s sufficiently large.

If $\mathcal{G}$ is in fact a hitting set generator for $\mathcal{C}_{\mathbb{F}}(s, n, s)$, then using Lemma 2.3, we obtain a hitting set $\mathcal{H}$ for $\mathcal{C}_{\mathbb{F}}(s, n, s)$ of size

$$(s \cdot d)^{k^c} = (s^{k+1})^{k^3} = s^{k^4+k^3} \leqslant s^{2k^4-\varepsilon} = s^{n-\varepsilon}$$

for some $\varepsilon > 0$ when s is large enough. Moreover, $\mathcal{H}$ is $s^{O(\sqrt{n})} \cdot |\mathcal{H}| \leqslant s^{O(n)}$-explicit. We now apply Theorem 5.1 to obtain the claimed $s^{\exp \circ \exp(O(\log^\star s))}$-explicit hitting set for $\mathcal{C}_{\mathbb{F}}(s, s, s)$ of size $s^{\exp \circ \exp(O(\log^\star s))}$. It remains to show that $\mathcal{G}$ is indeed a hitting set generator for $\mathcal{C}_{\mathbb{F}}(s, n, s)$.

To show this, suppose for the sake of contradiction that $\mathcal{G}$ is not a hitting set generator for $\mathcal{C}_{\mathbb{F}}(s, n, s)$. Then there is some $f(\overline{y}) \in \mathcal{C}_{\mathbb{F}}(s, n, s)$ such that $f(\overline{y}) \neq 0$ and $f(\mathcal{G}(\overline{z})) = 0$. We define the hybrid polynomials $f_0, \ldots, f_n$ by

$$\begin{aligned}
f_0(\overline{y}, \overline{z}) &= f(y_1, \ldots, y_n) \\
f_1(\overline{y}, \overline{z}) &= f(g_d(\overline{z}|_{S_1}), y_2, \ldots, y_n) \\
&\vdots \\
f_{n-1}(\overline{y}, \overline{z}) &= f(g_d(\overline{z}|_{S_1}), \ldots, g_d(\overline{z}|_{S_{n-1}}), y_n) \\
f_n(\overline{y}, \overline{z}) &= f(g_d(\overline{z}|_{S_1}), \ldots, g_d(\overline{z}|_{S_n})) = f(\mathcal{G}(\overline{z})).
\end{aligned}$$

Since $f_0 \neq 0$ and $f_n = 0$, there is some $i \in [n]$ such that $f_{i-1} \neq 0$ and $f_i = 0$. Assuming $|\mathbb{F}| > sd \geqslant \deg(f_i)$, we can find an assignment to the variables $\{y_j : j \neq i\}$ and $\{z_j : j \notin S_i\}$ such that f_i remains non-zero under this partial evaluation. If $\mathbb{F}$ is too small, we may find such an assignment using values from some finite extension $\mathbb{F}' \supseteq \mathbb{F}$ of size at least $sd + 1$ (and hence degree $O(\log(sd))$). After renaming variables, denote this non-zero restriction of f_i by $\overline{f}(z_1, \ldots, z_k, y)$.

We can compute $\overline{f}$ by composing the circuit for f with at most $n - 1$ copies of the partial evaluation of $g_d(\overline{z}|_{S_j})$ for $j < i$. By assumption, we can compute f with a circuit of size s. Since $|S_j \cap S_i| \leqslant 2$ for $j \neq i$, at most 2 variables in $\overline{z}|_{S_j}$ are unset. This implies each restriction of $g_d(\overline{z}|_{S_j})$ is a polynomial of degree d on 2 variables and thus can be computed

by a depth-two circuit of size at most $d \cdot (d+1)^2$. This yields a circuit for $\overline{f}$ of size at most $s + nd \cdot (d+1)^2$. Note that the degree of $\overline{f}$ is bounded by sd, since $\overline{f}$ is the composition of two polynomials of degrees at most s and d.

By assumption, we have that $\overline{f}(z_1, \ldots, z_k, y) \neq 0$ and $\overline{f}(z_1, \ldots, z_k, g_d(\overline{z})) = 0$. This implies that $y - g_d(\overline{z})$ is a factor of $\overline{f}$. We now apply Theorem 2.8 to factor the circuit for $\overline{f}$.

- If $\operatorname{char} \mathbb{F} = p > 0$, we obtain a circuit for $(y - g_d(\overline{z}))^{p^t} = y^{p^t} - g_d(\overline{z})^{p^t}$ for some $t \in \mathbb{N}$. Since $y^{p^t} - g_d(\overline{z})^{p^t}$ is a factor of $\overline{f}(z_1, \ldots, z_k, y)$, we must have

  $$dp^t = \deg(y^{p^t} - g_d(\overline{z})^{p^t}) \leqslant \deg(\overline{f}) \leqslant sd.$$

  This implies $p^t \leqslant s$. Since $\overline{f}$ has degree sd and is computable in size $s + O(nd^3)$, the circuit computing $y^{p^t} - g_d(\overline{z})^{p^t}$ has size at most $O((nsd)^{12})$. By setting $y = 0$ and negating the output of the circuit, we obtain a circuit for $g_d(\overline{z})^{p^t}$ of size $O((nsd)^{12})$.
  We now apply Corollary 3.6 a total of t times. This produces a circuit which computes $g_d(\overline{z})$ and has size $O((nsd)^{12} p^{2kt} 2^{kt} 3^t) = O((nsd)^{12} s^{3k+2})$. Here we use the fact that $p \geqslant 2$, so $2^{kt} \leqslant p^{kt} \leqslant s^k$ and $3^t \leqslant 4^t \leqslant p^{2t} \leqslant s^2$.
  In the case where $|\mathbb{F}| > sd$, the circuit for $\overline{f}$ was defined over $\mathbb{F}$, so the circuit for g_d is defined over $\mathbb{K} = \mathbb{F}^{p^{-\infty}}$. If instead $|\mathbb{F}| \leqslant sd$, the circuit for $\overline{f}$ was defined over a finite extension $\mathbb{F}' \supseteq \mathbb{F}$ of degree $O(\log(sd))$. As $\mathbb{F}'$ is a finite field, $\mathbb{F}'$ is perfect, so the circuit obtained from Corollary 3.6 is defined over $\mathbb{F}'$. We apply Lemma 2.7 to simulate this circuit over $\mathbb{F}$, incurring an extra $O(\log^3(sd))$ factor in the circuit size.
  In total, we now have a circuit which computes g_d over $\mathbb{K} = \mathbb{F}^{p^{-\infty}}$ and has size bounded by $O((nsd)^{12} s^{3k+2} \log^3(sd))$.
- If $\operatorname{char} \mathbb{F} = 0$, the previous case applies, but without the need to take a p^{th} root or simulate a field extension. This yields a circuit which computes $g_d(\overline{z})$ over $\mathbb{K} = \mathbb{F}$ and has size $O((nsd)^{12})$.

In both cases, we obtain a circuit which computes $g_d(\overline{z})$ over $\mathbb{K}$ and has size at most $O((nsd)^{12} s^{3k+2} \log^3(sd))$. Restating in terms of k and d, we have a circuit for g_d of size

$$O((nsd)^{12} s^{3k+2} \log^3(sd)) = O(k^{48} s^{14+3k} d^{12} \log^3(d)) = O(k^{48} d^{15+14/k} \log^3(d)).$$

Since $k \geqslant 1$ and k is a constant, we can bound the size of the circuit computing g_d by $O(d^{29} \log^3(d))$. This contradicts the fact that g_d requires circuits over $\mathbb{K}$ of size $d^\delta \geqslant d^{30}$ for sufficiently large d. Hence $\mathcal{G}$ is in fact a hitting set generator for $\mathcal{C}_{\mathbb{F}}(s, n, s)$. ◀

5.2 Comparison to Characteristic Zero

Over fields of characteristic zero, the recent work of Guo, Kumar, Saptharishi and Solomon [17] obtained what is currently the best-known derandomization of polynomial identity testing for $\mathcal{C}_{\mathbb{F}}(s, s, s)$ under a hardness assumption. From an explicit family of k-variate degree d polynomials of hardness $d^{\Omega(1)}$, they obtain an explicit hitting set for $\mathcal{C}_{\mathbb{F}}(s, s, s)$ of size $s^{O(1)}$. Specifically, they prove the following theorem.

▶ **Theorem 5.4** ([17]). *Let $\mathbb{F}$ be a field of characteristic zero. Let $k \in \mathbb{N}$ be large enough and let $\delta > 0$ be a fixed constant. Suppose $\{P_{k,d} \in \mathbb{F}[\overline{x}] : d \in \mathbb{N}\}$ is a family of $d^{O(k)}$-explicit k-variate polynomials of degree d such that $P_{k,d}$ cannot be computed by algebraic circuits of size smaller than d^δ. Then there is an $s^{(k/\delta)^{O(1)}}$-explicit hitting set for $\mathcal{C}_{\mathbb{F}}(s, s, s)$ of size $s^{O(k^2/\delta^2)}$.*

We remark that Guo, Kumar, Saptharishi, and Solomon [17] do not define the notion of explicitness they use in their result, but it is enough for $P_{k,d}$ to be computable by a uniform algorithm which runs in time $d^{O(k)}$. This is slightly different from our notion of strong

explicitness, where we require the coefficients of $P_{k,d}$ to be computable in $d^{O(k)}$ time. It is clear that one can pass from strong explicitness to the standard notion of explicitness by computing a polynomial as a sum of monomials. Via polynomial interpolation, one can show that polynomials which are "evaluation-explicit" are strongly explicit. In both cases, the explicitness parameter may degrade considerably, as the number of terms in a polynomial may be exponentially larger than the amount of time required to compute the polynomial or one of its coefficients. In general, one cannot hope to do better than this: in one direction, the coefficients of the permanent are easy to compute, but the permanent is widely conjectured to be hard to compute; in the other direction, there are examples of polynomials which are easy to compute but which have the permanent of a large matrix embedded in their coefficients (see, for example, Bürgisser [8, §2.3]).

In the context of Theorem 5.3 and Theorem 5.4, however, the two notions of explicitness coincide. When working with k-variate polynomials of degree d, we incur an overhead of $d^{O(k)}$ in passing between the two notions of explicitness. As the hypotheses of these theorems are already in the regime of (strong) $d^{O(k)}$-explicitness, the explicitness parameter changes by a polynomial factor, which is small enough to not affect the asymptotics of the results obtained.

The fact that the underlying field has characteristic zero is used in a key part of the proof of Theorem 5.4, and it is not clear how to adapt the proof to fields of positive characteristic. The generator used to design the hitting set in the conclusion of Theorem 5.4 is notably not a variation on the Kabanets-Impagliazzo generator, but instead a new generator whose construction is more algebraic than combinatorial in flavor.

Note that Theorem 5.3 and Theorem 5.4 require the same hardness assumption. This gives a second proof of derandomization of polynomial identity testing from an explicit family of hard constant-variate polynomials, although the derandomization we obtain is slightly weaker compared to Theorem 5.4. However, our construction does not require the characteristic of the underlying field to be zero. It is tempting to conjecture that one can recover the conclusion of Theorem 5.4 in positive characteristic by improving the bootstrapping process used to prove Theorem 5.1. It is unclear whether such a result is possible.

6 Relating Constant-Variate and Multivariate Lower Bounds

This work and the work of Guo, Kumar, Saptharishi, and Solomon [17] have shown that lower bounds against (strongly) explicit constant-variate polynomials yield very strong derandomizations of polynomial identity testing. We are able to give an explicit hitting set of size $s^{\exp \circ \exp(O(\log^\star s))}$ for $\mathcal{C}_{\mathbb{F}}(s, s, s)$ for any field $\mathbb{F}$ (this is Theorem 5.3), while Guo, Kumar, Saptharishi, and Solomon [17] obtain explicit hitting sets of size $s^{O(1)}$ for the same class when $\operatorname{char} \mathbb{F} = 0$. However, if one instead assumes the existence of a (strongly) explicit family of maximally-hard multivariate polynomials of low degree (specifically, degree $n^{O(1)}$ where n is the number of variables), it is not clear how to obtain similar derandomization results. The best-known derandomization from multivariate lower bounds is that of Kabanets and Impagliazzo [21], who gave an explicit hitting set of size $s^{O(\log s)}$ for $\mathcal{C}_{\mathbb{F}}(s, s, s)$.

The fact that we can obtain strong derandomizations of polynomial identity testing from constant-variate hardness raises the question of whether or not such derandomization is possible under multivariate hardness assumptions. A natural first approach to this would be to show that lower bounds for a (strongly) explicit family of multivariate polynomials imply comparable lower bounds against a (strongly) explicit family of constant-variate polynomials. Such an implication is known in the setting of non-commutative circuits and is due to Carmosino, Impagliazzo, Lovett, and Mihajlin [11].

It is not hard to show a connection in the other direction; that is, lower bounds against strongly explicit families of constant-variate polynomials can be translated into comparable lower bounds against strongly explicit families of multivariate polynomials. An easy way to do this is via the approach of Lemma 2.6.

In this section, we investigate to what extent a converse to Lemma 2.6 may hold. Unconditionally refuting the converse of Lemma 2.6 requires proving circuit lower bounds that seem far out of reach, so we have little hope to fully resolve this question. However, we can give some complexity-theoretic evidence which shows a converse to Lemma 2.6 is unlikely to hold. To do this, we take a detour into the arithmetic complexity of integers.

6.1 Complexity of Computing Integers

We start by defining the model we use to compute sequences of integers.

▶ **Definition 6.1.** *For a natural number $n \in \mathbb{N}$, let $\tau(n)$ denote the size of the smallest circuit which computes n using the constant 1 and the operations of addition, subtraction, and multiplication. Let $(a_n)_{n\in\mathbb{N}}$ be a sequence of natural numbers. If $\tau(a_n) \leqslant \log^{O(1)} n$, then we say $(a_n)_{n\in\mathbb{N}}$ is* easy to compute. *Otherwise, we say $(a_n)_{n\in\mathbb{N}}$ is* hard to compute.

As an example, the sequence $(2^n)_{n\in\mathbb{N}}$ is easy to compute, as we can compute 2^n in $O(\log n)$ arithmetic steps by repeated squaring. A major open problem in this area is to understand $\tau(n!)$, the complexity of the sequence of factorials. The following conjecture regarding $\tau(n!)$ appears to be folklore.

▶ **Conjecture 6.2.** The sequence of factorials $(n!)_{n\in\mathbb{N}}$ is hard to compute.

Prior work has established relationships between Conjecture 6.2 and other prominent conjectures in computational complexity. Blum, Cucker, Shub, and Smale [5, page 126] gave an argument that shows if $\tau(n!) \leqslant \log^{O(1)} n$, then there are circuits of $\log^{O(1)} n$ size to factor n. A related work by Shamir [37] reduces factorization to computing factorials, albeit in a slightly different model. Bürgisser [9] showed that Conjecture 6.2 implies that the $n \times n$ permanent cannot be computed by constant-free division-free algebraic circuits of size $n^{O(1)}$. Work by Lipton [28] shows that average-case hardness of factoring implies a slightly weaker form of Conjecture 6.2; namely, that the polynomial $\prod_{i=1}^{n}(x-i)$ is hard to compute by constant-free algebraic circuits.

Before moving on to address the question of a converse to Lemma 2.6, we present a reduction due to Shamir [37] which reduces the task of computing $n!$ to the task of computing $\binom{2n}{n}$.

▶ **Lemma 6.3** ([37])**.** *If $(\binom{2n}{n})_{n\in\mathbb{N}}$ is easy to compute, then $(n!)_{n\in\mathbb{N}}$ is easy to compute.*

Proof. Suppose $\tau\big(\binom{2n}{n}\big) \leqslant O(\log^c n)$. Recall the identity

$$n! = \begin{cases} ((n/2)!)^2 \cdot \binom{n}{n/2} & n \text{ is even} \\ n \cdot ((\frac{n-1}{2})!)^2 \cdot \binom{n-1}{(n-1)/2} & n \text{ is odd.} \end{cases}$$

This implies

$$\tau(n!) \leqslant \tau(n) + \tau((\lfloor n/2 \rfloor !)^2) + \tau\left(\binom{2 \cdot \lfloor n/2 \rfloor}{\lfloor n/2 \rfloor}\right).$$

Expanding out the recurrence and using the fact that $\tau((\lfloor n/2 \rfloor !)^2) \leqslant \tau(\lfloor n/2 \rfloor !) + 1$, we get

$$\begin{aligned}\tau(n!) &\leqslant \sum_{i=1}^{\log n}\left[\tau(\lfloor n/2^i\rfloor)+\tau\left(\binom{2\cdot\lfloor n/2^{i+1}\rfloor}{\lfloor n/2^{i+1}\rfloor}\right)\right]+1\\ &\leqslant \log n\cdot(O(\log n)+O(\log^c n)+1)\\ &\leqslant O(\log^{c+1} n).\end{aligned}$$

Hence $(n!)_{n\in\mathbb{N}}$ is easy to compute. ◀

6.2 The Inverse Kronecker Map and Constant-Free Circuits

Here, we show that two forms of a converse to Lemma 2.6 refute Conjecture 6.2 to varying degrees. Our first argument shows that a straightforward converse of Lemma 2.6 implies that Conjecture 6.2 fails infinitely often. That is, suppose $g(x)$ is a univariate degree d polynomial and $f(\overline{y})$ is a multilinear polynomial which simplifies to $g(x)$ under the mapping $y_i \mapsto x^{2^i}$. Lemma 2.6 says that hardness of $g(x)$ implies hardness of $f(\overline{y})$. The following conjecture, which we wish to conditionally refute, says that hardness of $f(\overline{y})$ implies hardness of $g(x)$.

▶ **Conjecture 6.4.** Let $g_{m,d}(\overline{x}) = \sum_{\overline{a}} \alpha_{\overline{a}}\overline{x}^{\overline{a}}$ be an m-variate degree d polynomial. Let $j : \{0,1\}^{\lfloor \log d\rfloor+1} \to [\![2^{\lfloor \log d\rfloor+1}]\!]$ be given by $j(\overline{e}) = \sum_{i=1}^{\lfloor \log d\rfloor+1} \overline{e}_i 2^{i-1}$. That is, $j(\overline{e})$ is the number whose binary representation corresponds to $\overline{e}$. Let

$$\overline{y} = (y_{1,1},\ldots,y_{1,\lfloor \log d\rfloor+1},\ \ldots\ ,y_{m,1},\ldots,y_{m,\lfloor \log d\rfloor+1})$$

and define

$$f_{m,d}(\overline{y}) = \sum_{\overline{e}\in\{0,1\}^{m\times\lfloor \log d\rfloor+1}} \alpha_{(j(\overline{e}_{1,\bullet}),\ldots,j(\overline{e}_{m,\bullet}))}\overline{y}^{\overline{e}}.$$

Suppose $f_{m,d}$ requires constant-free circuits of size s to compute. Then $g_{m,d}$ requires constant-free circuits of size $s^{\Omega(1)} - \Theta(m\log d)$ to compute.

We now show that Conjecture 6.4 implies the factorials are easy to compute infinitely often.

▶ **Theorem 6.5.** *Suppose Conjecture 6.4 holds over $\mathbb{Q}$. Then the sequence of factorials $(n!)_{n\in\mathbb{N}}$ is easy to compute infinitely often.*

Proof. It is easy to see that $\sum_{i=0}^{2^n}\binom{2^n}{i}x^i = (x+1)^{2^n}$ is computable by a constant-free algebraic circuit of size $O(n)$ via repeated squaring. Let

$$f_n(\overline{y}) = \sum_{\overline{e}\in\{0,1\}^{n+1}} \binom{2^n}{j(\overline{e})}\overline{y}^{\overline{e}}.$$

The contrapositive of Conjecture 6.4 yields a constant-free circuit of size $O(n^c)$ which computes f_n for some absolute constant c. Let $a_{n-1} = 1$ and $a_0 = \cdots = a_{n-2} = a_n = 0$. Then $f_n(\overline{a}) = \binom{2^n}{2^{n-1}} + 1$. By evaluating the circuit for f_n at $\overline{a}$ and subtracting 1, we obtain a circuit of size $O(n^c)$ which computes $\binom{2^n}{2^{n-1}}$.

We now follow the argument of Lemma 6.3 to construct circuits of size $O(n^{c+1})$ to compute $(2^n!)_{n\in\mathbb{N}}$. By definition, we have

$$\begin{aligned}
2^n! &= \binom{2^n}{2^{n-1}}(2^{n-1}!)^2 \\
&= \binom{2^n}{2^{n-1}}\binom{2^{n-1}}{2^{n-2}}^2(2^{n-2}!)^4 \\
&\vdots \\
&= \prod_{i=0}^{n-1}\binom{2^{n-i}}{2^{n-i-1}}^{2^i}.
\end{aligned}$$

Using the fact that we fact that we can compute $\binom{2^n}{2^{n-1}}$ by a circuit of size $O(n^c)$, we obtain

$$\tau(2^n!) \leqslant \sum_{i=0}^{n-1}\tau\left(\binom{2^{n-i}}{2^{n-i-1}}^{2^i}\right) \leqslant \sum_{i=0}^{n-1} O(n^{c+1}) \leqslant O(n^{c+2}).$$

Hence the factorials are easy to compute infinitely often. ◀

It is unclear whether there is meaningful evidence to suggest that the factorials are not easy to compute at numbers of the form 2^n. Because of this, Theorem 6.5 may be best viewed as evidence that if Conjecture 6.4 is true, the proof will not be straightforward.

Conjecture 6.4 can be seen as a base-two converse to Lemma 2.6. Instead, we might consider the following strengthening of Conjecture 6.4 to all number bases.

▶ **Conjecture 6.6.** Let $g_{m,d}(\overline{x}) = \sum_{\overline{a}} \alpha_{\overline{a}}\overline{x}^{\overline{a}}$ be an m-variate degree d polynomial. Let $k \in \mathbb{N}$ and let $j : [\![k]\!]^{\lfloor \log_k d\rfloor+1} \to [\![k^{\lfloor \log_k d\rfloor+1}]\!]$ be given by $j(\overline{e}) = \sum_{i=1}^{\lfloor \log_k d\rfloor+1}\overline{e}_i k^{i-1}$, that is, $j(\overline{e})$ is the number whose base-k representation corresponds to $\overline{e}$. Let $\overline{y} = (y_{1,1},\dots,y_{1,\lfloor \log_k d\rfloor+1},\ \dots\ ,y_{m,1},\dots,y_{m,\lfloor \log_k d\rfloor+1})$ and define

$$f_{m,d}(\overline{y}) = \sum_{\overline{e}\in[\![k]\!]^{m\times\lfloor \log_k d\rfloor+1}} \alpha_{(j(\overline{e}_{1,\bullet}),\dots,j(\overline{e}_{m,\bullet}))}\overline{y}^{\overline{e}}.$$

Suppose $f_{m,d}$ requires constant-free circuits of size s to compute. Then $g_{m,d}$ requires constant-free circuits of size $s^{\Omega(1)} - \Theta(m\log d)$ to compute.

We can show that this stronger conjecture is less likely to hold than Conjecture 6.4.

▶ **Theorem 6.7.** *Suppose Conjecture 6.6 holds over $\mathbb{Q}$. Then $(n!)_{n\in\mathbb{N}}$ is easy to compute.*

Proof. By Lemma 6.3, it suffices to show that the central binomial coefficients $\binom{2n}{n}_{n\in\mathbb{N}}$ are easy to compute. Let $n \in \mathbb{N}$ be given. There is constant-free circuit of size $O(\log n)$ which computes $g(x) = (x+1)^{2n}$. Consider the polynomial

$$f(y_1, y_n) = \sum_{i=0}^{n-1}\sum_{j=0}^{n-1}\binom{2n}{i+jn}y_1^i y_n^j,$$

where by convention $\binom{n}{k} = 0$ when $n < k$. Note that

$$f(x, x^n) = \sum_{i=0}^{n-1}\sum_{j=0}^{n-1}\binom{2n}{i+jn}x^{i+jn} = \sum_{k=0}^{n^2-1}\binom{2n}{k}x^k = \sum_{k=0}^{2n}\binom{2n}{k}x^k = (x+1)^{2n}.$$

The contrapositive of Conjecture 6.6 implies that f is computable by a constant-free circuit of size $O(\log^c n)$ for some absolute constant c. We now evaluate $f(0,1)$ to obtain

$$f(0,1) = \sum_{j=0}^{n-1} \binom{2n}{jn} = \binom{2n}{0} + \binom{2n}{n} + \binom{2n}{2n} = \binom{2n}{n} + 2.$$

By computing $f(0,1) - 2$, we obtain a constant-free circuit of size $O(\log^c n)$ which computes $\binom{2n}{n}$. Hence the central binomial coefficients are easy to compute. ◀

Note that the results of this section only give evidence that Conjecture 6.4 and Conjecture 6.6 do not hold over fields of characteristic zero. Over fields of positive characteristic, it is unclear whether these conjectures are likely to be true or false. This is somewhat interesting, as if Conjecture 6.4 holds over fields of positive characteristic, then we can replace constant-variate hardness with multivariate hardness in our extension of the Kabanets-Impagliazzo generator to fields of small characteristic.

7 Conclusion and Open Problems

In this work, we gave the first instantiation of the algebraic hardness-randomness paradigm over fields of small characteristic. Our main tool was the mod-p decomposition, which we used to efficiently compute p^{th} roots of circuits which depend on a small number of variables. This allowed us to extend known hardness-randomness tradeoffs due to Kabanets and Impagliazzo [21] to fields of small characteristic under seemingly stronger hardness assumptions. We also constructed a hitting set generator which, under suitable hardness assumptions, provides a near-complete derandomization of polynomial identity testing. As our hardness assumptions are somewhat atypical, we compared them to more standard hardness assumptions and gave a conditional result which says that our hardness assumptions are not implied by standard ones.

A number of problems in low-characteristic derandomization remain open, some of which we have pointed out earlier in this work. Here, we mention some challenges which our techniques are not able to resolve.

1. Is it possible to obtain hardness-randomness tradeoffs over fields of small characteristic using a strongly explicit family of hard multilinear polynomials as opposed to constant-variate polynomials?
2. Let $\mathbb{F}$ be a field of characteristic $p > 0$, where p is some fixed constant. Suppose $f(\overline{x})^p \in \mathbb{F}[\overline{x}]$ is an n-variate polynomial which can be computed by a circuit of size s over $\mathbb{F}$. Is there a circuit of size $s^{O(1)}$ which computes $f(\overline{x})$ in the case that $n = \omega(\log s)$?
3. In the conclusion of Theorem 5.1, is it possible to obtain a hitting set of size $s^{O(1)}$? If so, this would give a construction of a hitting set generator over low characteristic fields which qualitatively matches the parameters of the generator of Guo, Kumar, Saptharishi, and Solomon [17].
4. Is it possible to lift lower bounds from the multivariate regime to the constant-variate regime? It seems like the answer may be "no," but our evidence thus far only applies to constant-free circuits over fields of characteristic zero. What can we say if we remove the constant-free restriction? What about fields of positive characteristic?

References

1. Manindra Agrawal and Somenath Biswas. Primality and identity testing via Chinese remaindering. *J. ACM*, 50(4):429–443, 2003. Preliminary version in the *40th Annual IEEE Symposium on Foundations of Computer Science (FOCS 1999)*. doi:10.1145/792538.792540.
2. Manindra Agrawal, Sumanta Ghosh, and Nitin Saxena. Bootstrapping variables in algebraic circuits. *Proc. Natl. Acad. Sci. USA*, 116(17):8107–8118, 2019. Preliminary version in the *50th Annual ACM Symposium on Theory of Computing (STOC 2018)*. doi:10.1073/pnas.1901272116.
3. Manindra Agrawal, Neeraj Kayal, and Nitin Saxena. PRIMES is in P. *Ann. of Math. (2)*, 160(2):781–793, 2004. doi:10.4007/annals.2004.160.781.
4. Manindra Agrawal and V. Vinay. Arithmetic circuits: A chasm at depth four. In *Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2008)*, pages 67–75, 2008. doi:10.1109/FOCS.2008.32.
5. Lenore Blum, Felipe Cucker, Michael Shub, and Steve Smale. *Complexity and real computation*. Springer-Verlag, New York, 1998. With a foreword by Richard M. Karp. doi:10.1007/978-1-4612-0701-6.
6. Manuel Blum, Ashok K. Chandra, and Mark N. Wegman. Equivalence of free Boolean graphs can be decided probabilistically in polynomial time. *Inform. Process. Lett.*, 10(2):80–82, 1980. doi:10.1016/S0020-0190(80)90078-2.
7. Nicolas Bourbaki. *Algebra. II. Chapters 4–7*. Elements of Mathematics (Berlin). Springer-Verlag, Berlin, 1990. Translated from the French by P. M. Cohn and J. Howie.
8. Peter Bürgisser. *Completeness and reduction in algebraic complexity theory*, volume 7 of *Algorithms and Computation in Mathematics*. Springer-Verlag, Berlin, 2000. doi:10.1007/978-3-662-04179-6.
9. Peter Bürgisser. On defining integers and proving arithmetic circuit lower bounds. *Comput. Complexity*, 18(1):81–103, 2009. Preliminary version in the *24th Symposium on Theoretical Aspects of Computer Science (STACS 2007)*. doi:10.1007/s00037-009-0260-x.
10. Peter Bürgisser, Michael Clausen, and M. Amin Shokrollahi. *Algebraic complexity theory*, volume 315 of *Grundlehren der Mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences]*. Springer-Verlag, Berlin, 1997. With the collaboration of Thomas Lickteig. doi:10.1007/978-3-662-03338-8.
11. Marco L. Carmosino, Russell Impagliazzo, Shachar Lovett, and Ivan Mihajlin. Hardness amplification for non-commutative arithmetic circuits. In *Proceedings of the 33rd Annual Computational Complexity Conference (CCC 2018)*, volume 102 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 12:1–12:16. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2018. doi:10.4230/LIPIcs.CCC.2018.12.
12. Chi-Ning Chou, Mrinal Kumar, and Noam Solomon. Hardness vs randomness for bounded depth arithmetic circuits. In *Proceedings of the 33rd Annual Computational Complexity Conference (CCC 2018)*, volume 102 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 13:1–13:17. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2018. doi:10.4230/LIPIcs.CCC.2018.13.
13. Zeev Dvir, Amir Shpilka, and Amir Yehudayoff. Hardness-randomness tradeoffs for bounded depth arithmetic circuits. *SIAM J. Comput.*, 39(4):1279–1293, 2009. Preliminary version in the *40th Annual ACM Symposium on Theory of Computing (STOC 2008)*. doi:10.1137/080735850.
14. Michael A. Forbes, Sumanta Ghosh, and Nitin Saxena. Towards blackbox identity testing of log-variate circuits. In *Proceedings of the 45th International Colloquium on Automata, Languages and Programming (ICALP 2018)*, volume 107 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 54:1–54:16. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2018. doi:10.4230/LIPIcs.ICALP.2018.54.

15 Dima Grigoriev and Marek Karpinski. An exponential lower bound for depth 3 arithmetic circuits. In *Proceedings of the 30th Annual ACM Symposium on Theory of Computing (STOC 1998)*, pages 577–582. ACM, New York, 1998.

16 Dima Grigoriev and Alexander Razborov. Exponential lower bounds for depth 3 arithmetic circuits in algebras of functions over finite fields. *Appl. Algebra Engrg. Comm. Comput.*, 10(6):465–487, 2000. Preliminary version in the *39th Annual IEEE Symposium on Foundations of Computer Science (FOCS 1998)*. doi:10.1007/s002009900021.

17 Zeyu Guo, Mrinal Kumar, Ramprasad Saptharishi, and Noam Solomon. Derandomization from algebraic hardness: Treading the borders. In *Proceedings of the 60th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2019)*, pages 147–157, 2019. doi:10.1109/FOCS.2019.00018.

18 Ankit Gupta, Pritish Kamath, Neeraj Kayal, and Ramprasad Saptharishi. Arithmetic circuits: a chasm at depth 3. *SIAM J. Comput.*, 45(3):1064–1079, 2016. Preliminary version in the *54th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2013)*. doi:10.1137/140957123.

19 Pavel Hrubeš and Amir Yehudayoff. Arithmetic complexity in ring extensions. *Theory of Computing*, 7(8):119–129, 2011. doi:10.4086/toc.2011.v007a008.

20 Russell Impagliazzo and Avi Wigderson. P = BPP if E requires exponential circuits: derandomizing the XOR lemma. In *Proceedings of the 29th Annual ACM Symposium on Theory of Computing (STOC 1997)*, pages 220–229. ACM, New York, 1997.

21 Valentine Kabanets and Russell Impagliazzo. Derandomizing polynomial identity tests means proving circuit lower bounds. *Comput. Complexity*, 13(1-2):1–46, 2004. Preliminary version in the *35th Annual ACM Symposium on Theory of Computing (STOC 2003)*. doi:10.1007/s00037-004-0182-6.

22 Erich Kaltofen. Factorization of polynomials given by straight-line programs. *Advances in Computing Research*, 5, 1989.

23 Richard M. Karp, Eli Upfal, and Avi Wigderson. Constructing a perfect matching is in Random NC. *Combinatorica*, 6(1):35–48, 1986. Preliminary version in the *17th Annual ACM Symposium on Theory of Computing (STOC 1985)*. doi:10.1007/BF02579407.

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

25 Mrinal Kumar and Ramprasad Saptharishi. An exponential lower bound for homogeneous depth-5 circuits over finite fields. In *Proceedings of the 32nd Annual Computational Complexity Conference (CCC 2017)*, volume 79 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 31:1–30:30. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2017. doi:10.4230/LIPIcs.CCC.2017.31.

26 Mrinal Kumar and Ramprasad Saptharishi. Hardness-randomness tradeoffs for algebraic computation. *Bull. Eur. Assoc. Theor. Comput. Sci.*, 129:56–87, 2019.

27 Mrinal Kumar, Ramprasad Saptharishi, and Anamay Tengse. Near-optimal bootstrapping of hitting sets for algebraic circuits. In *Proceedings of the 30th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2019)*, pages 639–646. SIAM, Philadelphia, PA, 2019. doi:10.1137/1.9781611975482.40.

28 Richard J. Lipton. Straight-line complexity and integer factorization. In *Algorithmic number theory (Ithaca, NY, 1994)*, volume 877 of *Lecture Notes in Comput. Sci.*, pages 71–79. Springer, Berlin, 1994. doi:10.1007/3-540-58691-1_45.

29 László Lovász. On determinants, matchings, and random algorithms. In *Fundamentals of computation theory (Proc. Conf. Algebraic, Arith. and Categorical Methods in Comput. Theory, Berlin/Wendisch-Rietz, 1979)*, volume 2 of *Math. Res.*, pages 565–574. Akademie-Verlag, Berlin, 1979.

30 Ketan Mulmuley, Umesh V. Vazirani, and Vijay V. Vazirani. Matching is as easy as matrix inversion. *Combinatorica*, 7(1):105–113, 1987. Preliminary version in the *19th Annual ACM Symposium on Theory of Computing (STOC 1987)*. doi:10.1007/BF02579206.

31 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.

32 Ran Raz. Tensor-rank and lower bounds for arithmetic formulas. *J. ACM*, 60(6):Art. 40, 15, 2013. Preliminary version in the *42nd Annual ACM Symposium on Theory of Computing (STOC 2010)*. doi:10.1145/2535928.

33 Steven Roman. *Field theory*, volume 158 of *Graduate Texts in Mathematics*. Springer, New York, 2 edition, 2006.

34 Nitin Saxena. Progress on polynomial identity testing. *Bull. Eur. Assoc. Theor. Comput. Sci.*, 99:49–79, 2009.

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

36 Ronen Shaltiel and Christopher Umans. Simple extractors for all min-entropies and a new pseudorandom generator. *J. ACM*, 52(2):172–216, 2005. Preliminary version in the *42nd Annual IEEE Symposium on Foundations of Computer Science (FOCS 2001)*. doi:10.1145/1059513.1059516.

37 Adi Shamir. Factoring numbers in $O(\log n)$ arithmetic steps. *Inform. Process. Lett.*, 8(1):28–31, 1979. doi:10.1016/0020-0190(79)90087-5.

38 Amir Shpilka and Amir Yehudayoff. Arithmetic circuits: a survey of recent results and questions. *Found. Trends Theor. Comput. Sci.*, 5(3-4):207–388, 2010. doi:10.1561/0400000039.

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

40 Christopher Umans. Pseudo-random generators for all hardnesses. *J. Comput. System Sci.*, 67(2):419–440, 2003. Preliminary version in the *34th Annual ACM Symposium on Theory of Computing (STOC 2002)*. doi:10.1016/S0022-0000(03)00046-1.

41 Ryan Williams. Finding paths of length k in $O^*(2^k)$ time. *Inform. Process. Lett.*, 109(6):315–318, 2009. doi:10.1016/j.ipl.2008.11.004.