More on Noncommutative Polynomial Identity Testing

Andrej Bogdanov Hoeteck Wee*

Abstract

We continue the study of noncommutative polynomial identity testing initiated by Raz and Shpilka and present efficient algorithms for the following problems in the noncommutative model:

Polynomial identity testing: The algorithm gets as an input an arithmetic circuit with the promise that the polynomial it computes has small degree (for instance, a circuit of logarithmic depth or an arithmetic formula) and determines whether or not the output of the circuit is identically zero (as a formal expression). Unlike the algorithm by Raz and Shpilka, our algorithm is black-box (but randomized with one-sided error) and evaluates the circuit over the ring of matrices. In addition, we present query complexity lower bounds for identity testing and explore the possibility of derandomizing our algorithm. The analysis of our algorithm uses a noncommutative variant of the Schwartz-Zippel test.

Minimizing algebraic branching programs: The algorithm gets as an input an algebraic branching program (ABP) and outputs a smallest equivalent ABP. The algorithm is based on Nisan's characterization of ABP complexity, and uses as a sub-routine an algorithm for computing linear dependencies amongst arithmetic formulas, a problem previously studied by the authors.

1 Introduction

The problem of polynomial identity testing asks the following: Given a finite field $\mathbb{F}$ and a polynomial $p \in \mathbb{F}[x_1, \ldots, x_n]$ computable by an arithmetic circuit, can we determine whether p is identically zero?[1] The celebrated lemma of Schwartz and Zippel [12, 14] shows that if p is nonzero, then $p(a_1, \ldots, a_n)$ is unlikely to evaluate to zero if $a_1, \ldots, a_n$ are chosen at random from a set of size slightly larger than the degree of p. No polynomial-time deterministic algorithm for the problem is known, even in the special case when p is computable by an arithmetic formula rather than a circuit. A formula is a circuit where every gate has fan-out one.

Recently Raz and Shpilka [10] considered the noncommutative case of this problem. In the noncommutative scenario, p is a polynomial in the ring $\mathbb{F}\{x_1, \ldots, x_n\}$ of polynomials over noncommuting variables $x_1, \ldots, x_n$. Syntactically, a noncommutative circuit is exactly like a commutative arithmetic circuit, except that the inputs to the multiplication gates are ordered. Such a circuit represents a formal expression in $\mathbb{F}\{x_1, \ldots, x_n\}$, and we ask whether this expression is identically zero. If p is identically zero as a noncommutative polynomial, then p is identically zero also when viewed as a commutative polynomial. However, the converse is not true; for example the "commutator polynomial" $x_1x_2 - x_2x_1$ is identically zero in the commutative case, but nonzero in the noncommutative case.

Raz and Shpilka showed that the class of noncommutative *formulas* admits a deterministic polynomial-time algorithm for polynomial identity testing by reducing the problem to identity testing for algebraic branching programs (ABPs) using a technique of Nisan's and then solving the problem for ABPs. Unlike the Schwartz-Zippel algorithm for the commutative case, the algorithm of Raz and Shpilka is "non-black-box": While Schwartz-Zippel only uses the ability to evaluate the formula over its chosen input, Raz and Shpilka rely on the structure of the formula to determine whether it is identically zero.

To our knowledge, for the more general case of noncommutative *circuits* very little is known; there seems to be no known algorithm, deterministic, randomized, or even nondeterministic, to test if the circuit computes the identically zero polynomial.

1.1 Our Contributions and Perspective

We present a polynomial-time randomized algorithm for polynomial identity testing of noncommutative circuits, with the promise that the degree of the circuit is polynomial in its size. More generally, the running time is poly-

*Computer Science Division, University of California, Berkeley. {adib, hoeteck}@cs.berkeley.edu. Work supported by US-Israel BSF Grant 2002246.

[1] As in [10], we reserve the term "identically zero" to refer to a polynomial being identically zero as a formal expression. If a polynomial p evaluates to zero everywhere on the input domain, we say that *p vanishes* over the input domain. For instance, the bivariate polynomial $p(x, y) = (x^p - 1)(y^p - 1)$ vanishes over $\mathbb{F}_p^2$, but is not identically zero.

nomial in the circuit size, the degree and $\log |\mathbb{F}|$. Our algorithm is "black-box" and is based on a noncommutative variant of the Schwartz-Zippel test: We show that a polynomial p in $\mathbb{F}\{x_1, \ldots, x_n\}$ of degree d is unlikely to evaluate to zero when the formal variables are replaced with sufficiently large matrices over $\mathbb{F}$ or an extension of $\mathbb{F}$.

When the input is a noncommutative formula, our algorithm can be implemented in randomized NC using the standard technique of parallel formula evaluation. (However, this implementation is no longer black-box.) Parallel evaluation of arithmetic formulas is well known in the commutative setting [3, 4]; the approach easily extends for arithmetic formulas over rings of matrices.[2]

We suspect that when the input to our algorithm is a noncommutative formula, it may be possible to obtain a partial derandomization of our algorithm that uses only a polylogarithmic (in the formula size) number of random bits. Roughly, if one can show that no bivariate polynomial $p \in \mathbb{F}\{x, y\}$ computed by a formula of size s is a polynomial identity for matrices of dimension $\omega(\log s)$, then our randomized algorithm will use only $O(\log^3 s)$ bits of randomness. We do not know of an example that violates this conjecture, and in Section 6 we show that certain well known classes of identities satisfy it.

For the case of general noncommutative circuits (with no promise on the degree of the circuit), our argument merely yields that the problem is in coREXP. While this observation is not trivial (since a circuit can compute a noncommutative polynomial with a doubly exponential number of terms), it is far from satisfactory, as no hardness result about this problem is known. We show lower bounds (Section 4.2) which imply that any evaluation-based algorithm over algebras of dimension $2^{o(s)}$ whose proof of correctness treats the circuit as a "black box" computing a polynomial of degree $2^{O(s)}$ cannot do better. Therefore, to obtain an improved black-box algorithm for identity testing of general noncommutative circuits, one needs to either evaluate over an algebra of exponential dimension that admits some efficient implicit representation (compatible with addition and multiplication), or use properties of noncommutative circuits other than bounds on the degree and the number of variables of the polynomial computed by the circuit in the analysis.

This may be compared with a result by Kabanets and Impagliazzo [7] in the commutative setting, where they show a quasipolynomial derandomization of polynomial identity testing under the assumption that there exists an exponential-time computable family of multilinear polynomials that requires subexponential size arithmetic circuits. The difficulty in resolving such an assumption stems from our inability to prove arithmetic lower bounds for general circuits. In contrast, lower bounds for noncommutative formulas are easy to prove; however we were unable to show a good bound for the class of all polynomial identities of $k \times k$ matrices, or to find a counterexample.

We also present a new application of noncommutative polynomial identity testing in the context of minimizing algebraic branching programs (ABPs), starting from the Raz-Shpilka identity testing algorithm for ABPs. Although a minimization algorithm generalizes an identity testing algorithm in the context of ABPs[3], our work does not subsume the Raz-Shpilka algorithm, since we use their algorithm as a subroutine to compute linear dependencies amongst arithmetic formulas. (The latter problem previously surfaced in our work on hypercube linearity testing [2].) Our minimization algorithm builds on Nisan's characterization of ABP complexity [9]. In particular, Nisan's characterization is constructive, as it yields an ABP of size equal to the lower bound. Our algorithm may be seen as an efficient realization of Nisan's construction. To our knowledge, this is the first polynomial-time minimization algorithm for models outside of finite automata.

1.2 Additional Related Work

Motivated by the study of algebraic algorithms for approximating the permanent, Chien and Sinclair [5] extended Nisan's formula lower bound methods to obtain exponential ABP lower bounds for the permanent and the determinant in a very large class of noncommutative algebras with polynomial identities, or PI-algebras. They do not address the question of polynomial identity testing in PI-algebras, namely whether a given polynomial with coefficients in some field $\mathbb{F}$ vanishes over a PI-algebra containing $\mathbb{F}$. For polynomial identity testing in the black box model, the case of PI-algebras is easier than the case of the free noncommutative algebra. Our methods extend trivially to give randomized identity testing algorithms over any PI-algebra in which addition and multiplication of algebra elements is computable in time polynomial in the representation of the elements. As an interesting open problem, we note that there is no known deterministic algorithm for arithmetic formula identity testing in PI-algebras (even for special cases like the algebra of 2×2 matrices). In particular, the Raz-Shpilka algorithm does not extend to this case.

2 Preliminaries

Let $\mathbb{F}$ be a field, and we write $\mathbb{F}\{x_1, \ldots, x_n\}$ to denote the set of polynomials with coefficients from $\mathbb{F}$ over

[2] Brent's paper references a report by Maruyama [8] that extends his results over rings of matrices.

[3] A minimal ABP computing the identity must have some edge labelled with the zero function and therefore we can always use a minimization algorithm to solve identity testing for ABPs, even if the minimization algorithm does not output a canonical minimal ABP.

noncommuting variables $x_1, \ldots, x_n$. A polynomial $p \in \mathbb{F}\{x_1, \ldots, x_n\}$ is *homogeneous* if all its terms have the same total degree. We say p is *multilinear* if it is homogeneous of degree n with n variables, and the individual degree of all its variables is one. This means that a multilinear polynomial p has the form $p(x_1, \ldots, x_n) = \sum_{\sigma \in \mathrm{Per}(n)} \alpha_\sigma x_{\sigma(1)} \ldots x_{\sigma(n)}$, where $\mathrm{Per}(n)$ denotes the set of all permutations $\sigma : [n] \to [n]$.

Arithmetic circuits. An *arithmetic circuit* over a field $\mathbb{F}$ represents a multivariate polynomial in the obvious way: a multiplication (addition) gate outputs the product (sum) of the formal polynomials represented by its input. (We may also allow generalized addition gates that, on input f, g, compute arbitrary linear combinations $af + bg$, where $a, b \in \mathbb{F}$.) The input wires to the circuit correspond to the input variables of the polynomial and the output of the circuit computes some polynomial of the input variables. We measure the *size* of an arithmetic circuit as the number of gates. An arithmetic circuit is called a *formula* if the fan-out of every gate in the circuit is at most one.

Black-box evaluations of polynomials. Given a polynomial p over noncommuting variables over $\mathbb{F}$, we allow evaluations p over extension fields of $\mathbb{F}$, and more generally, any associative algebra over $\mathbb{F}$. Evaluating p over a noncommutative algebra is in fact necessary for "black-box" noncommutative polynomial identity testing. We stress that it is reasonable to ask for values of the polynomials where the input comes from an algebra over $\mathbb{F}$ because the polynomial is given to the algorithm in the form of an arithmetic circuit.

3 Polynomial Identities

Let $\mathbb{F}$ be a field and A be an associative algebra with identity over $\mathbb{F}$. The *dimension* of A over $\mathbb{F}$ is the dimension of A as an $\mathbb{F}$-vector space. Given a set $\mathcal{S} \subseteq A$ and a polynomial $p \in \mathbb{F}\{x_1, \ldots, x_n\}$, we call p a *polynomial identity* (in short, PI) for $\mathcal{S}$ if for all $a_1, \ldots, a_n \in \mathcal{S}$, $p(a_1, \ldots, a_n) = 0$. We say $\mathcal{S}$ *satisfies* a PI if some nonzero p is a PI for $\mathcal{S}$.

We will be particularly interested in the special case when A is the set of $k \times k$ matrices over $\mathbb{F}$ itself, or over some field extension $\mathbb{F}'$ over $\mathbb{F}$. We denote the space of $k \times k$ matrices over $\mathbb{F}$ by $\mathbb{F}^{k \times k}$. The (i, j)*th unit matrix* e_{ij}, where $1 \leq i, j \leq k$ is the matrix in $\mathbb{F}^{k \times k}$ whose (i, j)th entry is 1 and whose other entries are zero.

Let $S_k(\mathbb{F})$ denote the linear subspace of $\mathbb{F}^{k \times k}$ consisting of all matrices whose entries are all zero except along the main and second upper diagonal. Alternatively, $S_k(\mathbb{F})$ is the linear space spanned by the unit matrices $e_{ii}, 1 \leq i \leq k$ and $e_{i(i+1)}, 1 \leq i \leq k-1$. For $T \subseteq \mathbb{F}$ and $L(\mathbb{F})$ a space of matrices generated by some of the unit matrices e_{ij}, we denote by $L(T)$ the set of matrices in $L(\mathbb{F})$ all of whose entries are in T.

We state two lemmas that are standard in the theory of polynomial identities. Similar statements can be found, for example, in the books by Rowen [11] or Drensky and Formanek [6].

Lemma 3.1. *Let $\mathbb{F}$ be an arbitrary field, $\mathbb{F}'$ an extension of $\mathbb{F}$, S a linear subspace of $\mathbb{F}'^{k \times k}$. If there exists a PI for S of degree d over $\mathbb{F}$ then there exists a multilinear identity for S of degree at most d over $\mathbb{F}$.*

Proof. For an arbitrary polynomial $p \in \mathbb{F}'\{x_1, \ldots, x_n\}$, the *linearization* of p with respect to x_1 is the polynomial (in $n+1$ variables)

$$\begin{aligned} p_1(y_{11}, y_{12}, x_2, \ldots, x_n) = {} & p(y_{11} + y_{12}, x_2, \ldots, x_n) \\ & - p(y_{11}, x_2, \ldots, x_n) \\ & - p(y_{12}, x_2, \ldots, x_n). \end{aligned}$$

The linearization p_i of p with respect to x_i is defined likewise. Note that p_i has the following properties:

1. The total degree of p_i does not exceed the total degree of p.
2. If x_i has degree $d_i > 1$ in p, then y_{i1}, y_{i2} have degrees exactly $d_i - 1$ in p_i. (In particular, if $p \neq 0$ then $p_i \neq 0$.)
3. If x_i has degree one in p, then p_i consists of those terms of p that do not contain x_i, with a change in sign.

In addition, if p is a PI for S, then p_i is a PI for S as well. This suggests the following procedure: Starting with p, find a variable x_i in p such that either its degree is strictly greater than one or x_i does not appear in all terms of p. Replace p with p_i and repeat. By the above properties, the procedure must terminate with a multilinear identity for S of degree at most d. In particular, property 2 ensures that the individual degree of every variable in the final polynomial is one, and property 3 ensures that every variable appears in every monomial. □

The Amistur-Levitzki Theorem [1] states that over any field $\mathbb{F}$, the matrix algebra $\mathbb{F}^{k \times k}$ satisfies no PI of (total) degree less than $2k$, and satisfies exactly one (up to constant factor) multilinear identity of degree $2k$. We use a slight generalization of the first part of the Theorem, which we state below.

Lemma 3.2. *For arbitrary $\mathbb{F}$, $S_k(\mathbb{F})$ does not satisfy any PI of degree $2k-1$ or less.*

Notice that the statement holds for PIs with an arbitrary number of variables.

Proof. Suppose that $S_k(\mathbb{F})$ satisfies an identity of degree $2k-1$ or less. By Lemma 3.1, $S_k(\mathbb{F})$ must satisfy a multilinear identity of degree $n \leq 2k-1$. This identity has the form

$$p(x_1, \ldots, x_n) = \sum\nolimits_{\sigma \in \mathrm{Per}(n)} \alpha_\sigma x_{\sigma(1)} \ldots x_{\sigma(n)},$$

where $\alpha_\sigma \in \mathbb{F}$. Fix σ and consider the substitution

$$x_{\sigma(1)} = e_{11}, x_{\sigma(2)} = e_{12}, x_{\sigma(3)} = e_{22}, x_{\sigma(4)} = e_{23}, \ldots$$

where we choose consecutive entries by "walking along a staircase." The only term of p that does not vanish under this substitution is the term $\alpha_\sigma x_{\sigma(1)} \ldots x_{\sigma(n)} = \alpha_\sigma e_{1r}$, where $r = \lceil n/2 \rceil \leq k$. Since p is a PI for $S_k(\mathbb{F})$, we must have $\alpha_\sigma = 0$. It follows that p is identically zero. □

4 Noncommutative Identity Testing

Let $\mathcal{P}_{n,d}(\mathbb{F})$ denote the class of (noncommutative) polynomials over $\mathbb{F}$ of degree d in variables $x_1, \ldots, x_n$, excluding the zero polynomial. Note that in the problem of polynomial identity testing, we are interested in whether the polynomial is identically zero as a formal expression.

4.1 A black-box randomized algorithm

Our algorithm gets as an input an arithmetic circuit with the promise that the polynomial p it computes has small degree (for instance, a circuit of logarithmic depth or an arithmetic formula). The high-level idea is as follows: we evaluate p on random $k \times k$ matrices over some field $\mathbb{F}'$ and for a sufficiently large k. $\mathbb{F}'$ is a field extension of $\mathbb{F}$, with $|\mathbb{F}'| \gg d$. We regard each entry of the matrix in $\mathbb{F}'^{k \times k}$ as an indeterminate, and we view the k^2 indeterminates as commuting variables. Upon substitution, p computes a $k \times k$ matrix wherein each entry is a polynomial in k^2 (commuting) variables. If p is not a PI for $\mathbb{F}'^{k \times k}$, then some entry of p computes a nonzero polynomial of degree at most d over $\mathbb{F}$. In particular, if we pick a random matrix, p will compute a nonzero matrix with high probability, and we may obtain a lower bound for this probability via commutative Schwartz-Zippel. In order to optimize the parameters, we will pick random matrices from a linear subspace L_k of $\mathbb{F}'^{k \times k}$, where some entries of the matrix are fixed as 0 and the remaining entries are chosen randomly from some subset T of $\mathbb{F}'$.

Lemma 4.1 (Noncommutative Schwartz Zippel). *Let $p \in \mathbb{F}\{x_1, \ldots, x_n\}$ be a polynomial of degree d, $\mathbb{F}'$ be an extension of $\mathbb{F}$, L_k be a linear subspace of $\mathbb{F}'^{k \times k}$ generated by an arbitrary subset of the unit matrices e_{ij}, and $T \subseteq \mathbb{F}'$. Then either p is a PI for L_k, or*

$$\Pr[p(M_1, \ldots, M_n) = 0] \leq d/|T|,$$

where $M_1, \ldots, M_n$ are chosen independently at random from $L_k(T)$.

Proof. Let $e_1, \ldots, e_{k^2}$ denote a basis of $\mathbb{F}'^{k \times k}$ over $\mathbb{F}'$, chosen in such a way that the first l vectors $e_1, \ldots, e_l$ are the unit matrices that generate L_k. Consider the polynomial $q \in \mathbb{F}\{y_{11}, \ldots, y_{nl}\}$ which is the image of p under the substitutions $x_i = y_{i1}e_1 + \cdots + y_{il}e_l$. Informally, q is a $k \times k$ matrix where each matrix entry computes a polynomial q_i ($1 \leq i \leq k^2$) in $\mathbb{F}[y_{11}, \ldots, y_{nl}]$. We can therefore (uniquely) rewrite q in the form

$$q(y_{11}, \ldots, y_{nl}) = \sum\nolimits_{j=1}^{k^2} q_j(y_{11}, \ldots, y_{nl}) e_j,$$

where $q_1, \ldots, q_{k^2}$ are *commutative* polynomials in $\mathbb{F}[y_{11}, \ldots, y_{nl}]$. Moreover, it is easy to check that if p has degree d, then each q_j has degree at most d. If all of the polynomials $q_1, \ldots, q_{k^2}$ are identically zero, then p is identically zero, so p is a PI for L_k. Otherwise, one of $q_1, \ldots, q_{k^2}$ (say q_1) is nonzero. By the commutative Schwartz-Zippel Lemma, $q_1(a_{11}, \ldots, a_{nl})$ remains nonzero with probability $1 - d/|T|$ under a random substitution $y_{ij} = a_{ij} \in T$. Choosing random matrices $M_1, \ldots, M_n \in L_k(T)$ amounts exactly to choosing random values $a_{ij} \in T$, so it follows that $p(M_1, \ldots, M_n) \neq 0$ with at least the same probability. □

Theorem 4.2. *There exists a black-box, randomized identity testing algorithm for the class $\mathcal{P}_{n,d}$ that uses $O(d \log n \log(d \log n/\epsilon))$ random bits and succeeds with probability $1 - \epsilon$.*

Proof. Let $\mathbb{F}'$ be an extension of $\mathbb{F}$ (or perhaps $\mathbb{F}$ itself) of size at least $d \log n/\epsilon$, and let T be a subset of $\mathbb{F}'$ of size $d \log n/\epsilon$. Without loss of generality, suppose n is a power of two, and let $b_{i1}b_{i2} \ldots b_{i \log n} \in \{0,1\}^{\log n}$ denote the (padded) binary representation of $i-1$, for $1 \leq i \leq n$. Let $k = d \log n/2$. Our algorithm for polynomial identity testing does the following:

1. Choose random matrices $M_0, M_1 \in S_k(T)$.
2. For $1 \leq i \leq n$, let $N_i = \prod_{j=1}^{\log n} M_{b_{ij}}$.
3. Accept if $p(N_1, \ldots, N_n)$ evaluates to 0, and reject otherwise.

Let $q(Y_0, Y_1)$ be the polynomial obtained from $p(X_1, \ldots, X_n)$ after performing the substitution $X_i = \prod_{j=1}^{\log n} Y_{b_{ij}}$. We remark that q is not identically zero, as any monomial of q has a unique pre-image in p. Moreover, q has degree at most $d \log n$, so $q \in \mathcal{P}_{2,d \log n}$. By Lemma 4.1, either q is a PI for $S_k(\mathbb{F})$, or $q(M_0, M_1) \neq 0$ with probability $1 - \epsilon$. By Lemma 3.2, q cannot be a PI for $S_k(\mathbb{F})$, so the latter must hold. □

4.2 Lower bounds on query complexity

Theorem 4.3. *Let A be an associative algebra with identity of dimension k over $\mathbb{F}$. Any deterministic black-box algorithm for testing $\mathcal{P}_{n,d}$ that evaluates over A must make at least n^d/k queries. The lower bound holds for both adaptive and nonadaptive queries.*

Proof. Let $e_1, \ldots, e_k$ be a basis for A as an $\mathbb{F}$-vector space. Given a set of elements $a_{ij} \in A$, $1 \leq i \leq q, 1 \leq j \leq n$, where $q < n^d/k$, we exhibit a polynomial $p \in \mathcal{P}_{n,d}$ such that for all $1 \leq i \leq q$, $p(a_{i1}, \ldots, a_{in}) = 0$. Write

$$p(x_1, \ldots, x_n) = \sum\nolimits_{s \in [n]^d} \beta_s x_{s_1} \cdots x_{s_d},$$

where $\beta_s \in \mathbb{F}$ are "indeterminates". For each $s \in [n]^d$, $a_{is_1} \cdots a_{is_d} \in A$, so we may write $a_{is_1} \cdots a_{is_d} = \sum_{l=1}^{k} a_{isl} e_l$. Upon substitution, we obtain:

$$\begin{aligned} p(a_{i1}, \ldots, a_{in}) &= \sum\nolimits_{s \in [n]^d} \beta_s \sum\nolimits_{l=1}^{k} a_{isl} e_l \\ &= \sum\nolimits_{l=1}^{k} \Big(\sum\nolimits_{s \in [n]^d} a_{isl} \beta_s \Big) e_l. \end{aligned}$$

The equation $p(a_{i1}, \ldots, a_{in}) = 0$ is equivalent to the set of linear constraints:

$$\text{For all } 1 \leq l \leq k, \sum\nolimits_{s \in [n]^d} a_{isl} \beta_s = 0.$$

Taking the constraints over all possible $i = 1, 2, \ldots, q$, we have a system of qk linear constraints over the variables β_s. Since there are n^d distinct β_s and $n^d > qk$, there exists a nonzero solution for the system. This specifies a nonzero polynomial $p \in \mathcal{P}_{n,d}$ that vanishes on all queries. □

Corollary 4.4. *Let A be an associative algebra with identity of dimension k over $\mathbb{F}$. Any randomized black-box algorithm using R random bits and Q queries for testing $\mathcal{P}_{n,d}$ by evaluating over A satisfies $R + \log Q \geq d \log n - \log k$.*

Any efficient algorithm can conceivably only compute over algebras with $poly(n)$ dimensions, so for $d = \omega(1)$, we have a lower bound of $\Omega(d \log n)$ for $R + \log Q$. Our algorithm from Theorem 4.2 achieves $\tilde{\Theta}(d \log n)$, which is tight up to polylogarithmic factors.

5 Minimizing ABPs

Definition 5.1 ([9]). *An* algebraic branching program (ABP) *is a leveled directed acyclic graph with one vertex of in-degree zero, which is called the source, and one vertex of out-degree zero, which is called the sink. The vertices of the graph are partitioned into levels numbered $0, \ldots, d$. Edges are labeled with a homogeneous linear function in the input variables, and may only connect vertices from level i to vertices from level $i + 1$. The source is the only vertex at level 0 and the sink is the only vertex at level d. Finally, the size of the ABP is the number of vertices in the graph.*

The polynomial that is computed by an ABP is the sum over all directed paths from the source to the sink of the product of linear functions that labeled the edges of the path. It is clear that an ABP with $d + 1$ levels computes a homogeneous polynomial of degree d.

Lemma 5.2. *There is a deterministic polynomial-time algorithm that on input k homogeneous ABPs of the same degree, computes the space of linear dependencies amongst the k ABPs.*

Proof. Suppose the k ABPs define polynomials $P_1, \ldots, P_k$ in $\mathbb{F}\{x_1, \ldots, x_n\}$. Let u, v denote new formal variables, and for each $\alpha_1, \ldots, \alpha_k \in \mathbb{F}$, consider the polynomial P in $\mathbb{F}\{x_1, \ldots, x_n, u, v\}$ computed by the ABP with source node s and sink node t and for each $i = 1, 2, \ldots, k$, an edge from s to the source node of P_i labelled u, and an edge from the sink node of P_i to t labelled $\alpha_i v$. It is easy to see that

$$\alpha_1 P_1 + \ldots + \alpha_k P_k \equiv 0 \Leftrightarrow P \equiv 0$$

Now, apply the Raz-Shpilka identity testing algorithm [10, Sec 2.2] to P, reducing P to a ABP P' of depth 2, with the guarantee that $P \equiv 0$ iff $P' \equiv 0$. In addition, P' has the following structure: the edges between the source node and level 1 nodes are labelled with linear expressions over new variables $x'_1, \ldots, x'_m$ and the edges between the level 1 nodes and the sink node are labelled with $\alpha_1 v, \ldots, \alpha_k v$ respectively. Now, we can expand the polynomial computed by P' and by solving the linear system over the variables $x'_1 v, \ldots, x'_m v$ compute the linear space comprising all $\alpha_1, \ldots, \alpha_k$ such that P' (and thus P) computes the zero polynomial. □

Corollary 5.3. *There is a deterministic polynomial-time algorithm that on input k arithmetic formulas, computes the space of linear dependencies amongst the k formulas.*

Proof. Let d denote the maximal depth of the input formula. Our algorithm will first transform each of the given formula to at most $d + 1$ ABPs and for each $i = 0, 1, \ldots, d$, compute the linear dependencies for the k ABPs computing the homogeneous part of degree i for each input formula. The final output is the intersection of the $d+1$ linear spaces. □

Let f be a homogeneous polynomial of degree d on n variables $x_1, \ldots, x_n$. A *k-term* is an ordered sequence of k variables amongst $x_1, \ldots, x_n$ (allowing repetitions). For each $0 \leq k \leq d$, we define a matrix $M_k(f) \in \mathbb{F}^{n^k \times n^{d-k}}$ as follows: there is a row for each k-term τ and each $(d-k)$-term σ, and the (τ, σ)-th entry of $M_k(f)$ is the coefficient of monomial $\tau\sigma$ in f.

Theorem 5.4 ([9]). *The smallest ABP computing f has size $\sum_{k=0}^{d} \operatorname{rank}(M_k(f))$.*

Theorem 5.5. *There is a deterministic polynomial-time algorithm that on input an ABP outputs a smallest equivalent ABP.*

Proof. For $k = 0, 1, \ldots, d$, where d is the degree of the polynomial computed by the input ABP B, let $v_1, \ldots, v_t$ be the vertices of the ABP in the k'th level. We define the matrices L_k and R_k as follows: L_k will have a row for each k-term and a column for each $0 \leq i \leq t$ and R_k will have a row for each $0 \leq i \leq t$ and a column for each $(d-k)$-term. For a k-term τ and i, (τ, i)-th entry of L_k is the coefficient of τ in the polynomial computed by the restricted ABP with sink v_i. For a $(d-k)$-term σ and i, the (i, σ)-th entry of $R_k[i, \sigma]$ is the coefficient of σ in the polynomial computed by the restricted ABP with source v_i. It is easy to verify that $M_k(f) = L_k R_k$ and thus $\operatorname{rank}(M_k(f)) = \min\{\operatorname{rank}(L_k), \operatorname{rank}(R_k)\}$. In fact, the lower bound in Theorem 5.4 follows from this and the observation that $\operatorname{rank}(L_k), \operatorname{rank}(R_k) \leq t$.

Using the algorithm from Lemma 5.2, we may compute a basis for the polynomials computed by the t restricted ABPs with sink v_i, for $i = 1, \ldots, t$, along with a representation of the remaining polynomials as a linear combination of the basis functions; we call v_i a basis vertex if the polynomial computed by the restricted ABP with sink v_i is part of the basis. We eliminate each vertex v_i that is not a basis vertex, and each edge connecting v_i to a vertex u in the $k+1$'st level is replaced by edges connecting u to the appropriate linear combination of the basis vertices in the k'th level. Now, we have a new ABP B' computing the same polynomial as B, except B' has exactly $\operatorname{rank}(L_k)$ vertices in level k. By repeating the same procedure in a bottom-up manner, we obtain a new ABP B'' computing the same polynomial as B and where there are at most $\min\{\operatorname{rank}(L_k), \operatorname{rank}(R_k)\}$ vertices in the k'th level for any $0 \leq k \leq d$. It follows from Theorem 5.4 that B'' is a smallest ABP computing the same polynomial as B. □

6 Identity Testing for Noncommutative Formulas

In this section we apply our randomized algorithm for "black box" identity testing to the case when the input is represented by an arithmetic formula. Like in the case of commutative identity testing, we obtain an algorithm in the complexity class coRNC. We also remark on a possible partial derandomization of this algorithm, if certain lower bounds on the class of matrix identities can be proved.

6.1 Parallel evaluation of arithmetic formulas

We begin with a variant of a theorem by Brent, Kuck, and Maruyama [3] regarding parallel evaluation of arithmetic formulas. It is a straightforward generalization of their result for the noncommutative scenario. For completeness, we include a sketch of the proof.

We assume that we have a representation of $\mathbb{F}$ that allows addition in time $O(\log |\mathbb{F}|)$ and multiplication in time $O(\log^2 |\mathbb{F}|)$.

Lemma 6.1. *There is a parallel randomized algorithm that, given a noncommutative formula $p(x_1, \ldots, x_n)$ of size s, and matrices $M_1, \ldots, M_n \in \mathbb{F}^{k \times k}$, computes the matrix $p(M_1, \ldots, M_n)$ in parallel in time $O(\log(s) \log^2(k|\mathbb{F}|))$ using $\operatorname{poly}(sk)$ processors.*

Proof sketch. Let T be the parse tree of formula p. The internal nodes of p are labeled by gates, and the leaves are labeled by the variables $x_1, \ldots, x_n$. Without loss of generality, assume that all leaves are labeled by distinct variables. (This does not affect the size of the formula.) By a simple lemma of Brent [4], if $n \geq 3$, there is a node v of T such that the subtree T_v of T rooted at v has size at least $2n/3$, but both T_{vl} and T_{vr} have size at most $2n/3$, where vl, vr are the children of v. Let T_y be the tree obtained from T by contracting all descendants of v and labelling the new leaf at v with a new formal variable y.

To evaluate p, we recursively evaluate in parallel all of T_{vl}, T_{vy}, and T_y over the input matrices, treating y as a formal variable. The formal polynomial represented by T_y has the form $AyB + C$, where A, B, and C are polynomials in $x_1, \ldots, x_n$ only. We show that all of A, B, and C can be represented by trees T_A, T_B and T_C of total size at most $2n/3$. Let $\theta_1, \ldots, \theta_k$ denote the sequence of operations on the path from y to the root of T_y. Let C_i denote the subtree of θ_i that does not contain y. It is not difficult to check that T_A, T_B and T_C can be constructed as follows:

1. The tree T_A is the product of all C_i such that $\theta_i = *$ and C_i is a *left* child of θ_i.
2. The tree T_B is the product of all C_i such that $\theta_i = *$ and C_i is a *right* child of θ_i.
3. The tree T_C is obtained from T_y by setting $y = 0$.

After all of T_A, T_B, T_C and T_v have been evaluated, we are left with performing two matrix multiplications and one matrix addition. These can be done in parallel in time $O(\log^2(k|\mathbb{F}|))$. The processor and time complexity follow easily. □

Given a formula of size s, we can compute a field extension $\mathbb{F}'$ of $\mathbb{F}$ containing at least s elements in zero-error probabilistic time $\operatorname{poly}(|\mathbb{F}|, \log s)$ [13]. This observation,

together with Theorem 4.2 and Lemma 6.1 gives the following:

Theorem 6.2. *Fix a finite field $\mathbb{F}$. Noncommutative arithmetic formula identity testing over $\mathbb{F}$ is in* coRNC.

6.2 Towards a derandomization

For a fixed field $\mathbb{F}$, let $k(s)$ denote the smallest k such that no noncommutative ABP of size s is a PI for $\mathbb{F}^{k\times k}$. We observe that given a bound on $k(s)$, Lemma 4.1 gives the following black-box randomized algorithm for identity testing of ABPs. Given a polynomial $p \in \mathbb{F}[Y_0, Y_1]$ represented by an ABP of size s, choose random matrices $M_0, M_1 \in T^{k(s)\times k(s)}$, where T is a subset of $\mathbb{F}$ (or of a field extension of $\mathbb{F}$) of size at least s/ϵ, evaluate $p(M_0, M_1)$, and accept if p evaluates to the zero matrix. This test requires $(k(s))^2 \log(s/\epsilon)$ random bits. (Note that this can be extended to n variate polynomials by the argument used in the proof of Theorem 4.2; the randomness becomes $(k(s \log n))^2 \log(s \log n/\epsilon)$ random bits.)

The Amistur-Levitzki Theorem yields the bound $k(s) \leq s/2$, and does not give any improvement over our results from Section 4.1. However, it appears that all known examples of PI for $k \times k$ matrices have ABP size that is exponential in k. Although we are unable to prove a general upper bound for $k(s)$, we illustrate this for two well known classes of identities.

The best known identity for $\mathbb{F}^{k\times k}$ is the *standard identity* $s_{2k}(x_1, \ldots, x_{2k}) = 0$, where

$$s_n(x_1, \ldots, x_n) = \sum\nolimits_{\sigma\in \mathrm{Per}(n)} \mathrm{sign}(\sigma) x_{\sigma(1)} \ldots x_{\sigma(n)}.$$

The standard identity is a special case of a more general class of identities called normal identities. A polynomial $p(x_1, \ldots, x_t, y_1, \ldots, y_n)$ is *t-normal* if it is multilinear and if for all $1 \leq i < j \leq t$, p vanishes under the substitution $x_i = x_j$. The standard polynomial s_n is n-normal. The following lemma is an easy application of Theorem 5.4, using standard properties of normal polynomials ([11], Section 1.2).

Lemma 6.3. *Every t-normal polynomial has ABP size at least 2^t.*

Another well known identity for $\mathbb{F}^{n\times n}$ is the *identity of algebraicity*

$$a_n(x, y_1, \ldots, y_{n+1}) = \sum\nolimits_{\sigma\in \mathrm{Per}(n)} y_1 x^{\sigma(1)} y_2 \ldots y_n x^{\sigma(n)} y_{n+1}.$$

Lemma 6.4. *The polynomial a_n has ABP size $\Omega(2^n/n^{5/2})$.*

Proof sketch. We show that there exists a k such that $\mathrm{rank}(M_k(a_n)) = \Omega(2^n/n^{5/2})$. Let $\mathcal{S}_t$ be the collection of all subsets of $[n]$ of size $\lfloor n/2 \rfloor$ whose entries sum up to t. By the pigeonhole principle, there must exist a value of t such that $|\mathcal{S}_t| > \binom{n}{\lfloor n/2 \rfloor}/n^2 = \Omega(2^n/n^{5/2})$. Fix this t. Let $k = t + \lfloor n/2 \rfloor$.

There is a submatrix N_k of $M_k(a_n)$ that is decomposable into $|\mathcal{S}_t| \times |\mathcal{S}_t|$ blocks, such that its diagonal blocks are nonzero but all the other blocks are zero. It must then hold that $\mathrm{rank}(M_k(a_n)) \geq \mathrm{rank}(N_k) \geq |\mathcal{S}_t|$. The rows of N_k are indexed by terms of the form $y_1 x^{s_1} y_2 x^{s_2} \ldots y_{\lfloor k/2 \rfloor} x^{s_{\lfloor k/2 \rfloor}}$, where $\{s_1, \ldots, s_{\lfloor k/2 \rfloor}\} \in \mathcal{S}_t$. The columns are indexed by terms of the form $y_{\lfloor k/2 \rfloor} x^{s_{\lfloor k/2 \rfloor + 1}} \ldots x^{s_n} y_{n+1}$, where $[n] - \{s_{\lfloor k/2 \rfloor + 1}, \ldots, s_n\} \in \mathcal{S}_t$. □

Finally, we note that there exist noncommutative algebras admitting identities whose formula size is polynomial in the dimension; for example the algebra of $k \times k$ upper triangular matrices over a field $\mathbb{F}$ satisfies the identity

$$(x_1x_2 - x_2x_1)(x_3x_4 - x_4x_3) \ldots (x_{2k-1}x_{2k} - x_{2k}x_{2k-1}).$$

7 Acknowledgments

We thank Steve Chien, Ran Raz, Luca Trevisan, and Salil Vadhan for helpful conversations.

References

[1] S. A. Amistur and J. Levitzki. Minimal identities for algebras. In *Proceedings of the of the American Mathematical Society*, volume 1, pages 449–463, 1950.

[2] A. Bogdanov and H. Wee. A stateful implementation of a random function supporting parity queries over hypercubes. In *Proceedings of the 8th International Workshop on Randomization and Computation (RANDOM)*, pages 298–309, 2004.

[3] R. Brent, D. Kuck, and K. Maruyama. The parallel evaluation of arithmetic expressions without division. *IEEE Transactions on Computers*, C-22:532–534, 1973.

[4] R. P. Brent. The parallel evaluation of general arithmetic expressions. *Journal of the ACM*, 21(2):201–206, 1974.

[5] S. Chien and A. Sinclair. Algebras with polynomial identities and computing the determinant. In *Proceedings of the 45th IEEE Symposium on Foundations of Computer Science*, 2004.

[6] V. Drensky and E. Formanek. *Polynomial Identity Rings*. Advanced Courses in Mathematics – CRM Barcelona. Birkhauser, 2004.

[7] V. Kabanets and R. Impagliazzo. Derandomizing polynomial identity tests means proving circuit lower bounds. In *Proceedings of the 35th ACM Symposium on Theory of Computing*, pages 355–364, 2003.

[8] K. Maruyama. The parallel evaluation of matrix expressions. Technical Report RC 4380, IBM Research Center, Yorktown Heights, New York, May 1973.

[9] N. Nisan. Lower bounds for non-commutative computation. In *Proceedings of the 23rd ACM Symposium on Theory of Computing*, pages 410–418, 1991.

[10] R. Raz and A. Shpilka. Deterministic polynomial identity testing in non commutative models. In *Proceedings of the 17th Conference on Computational Complexity*, pages 215–222, 2004.

[11] L. H. Rowen. *Polynomial Identities in Ring Theory*. Academic Press, 1980.

[12] J. T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *Journal of the ACM*, 27(4):701–717, 1980.

[13] M. Sudan. Lecture notes on algebra. http://theory.lcs.mit.edu/~madhu/FT01/scribe/algebra.ps, September 2001.

[14] R. E. Zippel. Probabilistic algorithms for sparse polynomials. In *Proceedings of EUROSAM 79*, pages 216–226, 1979.