A Special Case of Rational Identity Testing and the Brešar-Klep Theorem

V. Arvind
Institute of Mathematical Sciences (HBNI), Chennai, India
arvind@imsc.res.in

Abhranil Chatterjee
Institute of Mathematical Sciences (HBNI), Chennai, India
abhranilc@imsc.res.in

Rajit Datta
Chennai Mathematical Institute, India
rajit@cmi.ac.in

Partha Mukhopadhyay
Chennai Mathematical Institute, India
partham@cmi.ac.in

Abstract

We explore a special case of rational identity testing and algorithmic versions of two theorems on noncommutative polynomials, namely, Amitsur's theorem [2] and the Brešar-Klep theorem [7] when the input polynomial is given by an algebraic branching program (ABP). Let f be a degree-d n-variate noncommutative polynomial in the free ring $\mathbb{Q}\langle x_1, x_2, \ldots, x_n\rangle$ over rationals.

1. We consider the following special case of *rational identity testing*: Given a noncommutative ABP as white-box, whose edge labels are linear forms or inverses of linear forms, we show a deterministic polynomial-time algorithm to decide if the rational function computed by it is equivalent to zero in the free skew field $\mathbb{Q}\langle\langle X\rangle\rangle$. Given black-box access to the ABP, we give a deterministic quasi-polynomial time algorithm for this problem.
2. Amitsur's theorem implies that if a noncommutative polynomial f is nonzero on $k \times k$ matrices then, in fact, $f(M_1, M_2, \ldots, M_n)$ is *invertible* for some matrix tuple $(M_1, M_2, \ldots, M_n) \in (\mathbb{M}_k(\mathbb{Q}))^n$. While a randomized polynomial time algorithm to find such $(M_1, M_2, \ldots, M_n)$ given black-box access to f is simple, we obtain a deterministic $s^{O(\log d)}$ time algorithm for the problem with black-box access to f, where s is the minimum ABP size for f and d is the degree of f.
3. The Brešar-Klep Theorem states that the span of the range of any noncommutative polynomial f on $k \times k$ matrices over $\mathbb{Q}$ is one of the following: zero, scalar multiples of I_k, trace-zero matrices in $\mathbb{M}_k(\mathbb{Q})$, or all of $\mathbb{M}_k(\mathbb{Q})$. We obtain a deterministic polynomial-time algorithm to decide which case occurs, given white-box access to an ABP for f. We also give a deterministic $s^{O(\log d)}$ time algorithm given black-box access to an ABP of size s for f. Our algorithms work when $k \geq d$.

Our techniques are based on some automata theory combined with known techniques for noncommutative ABP identity testing [14, 9].

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

Keywords and phrases Rational identity testing, ABP with inverses, Brešar-Klep Theorem, Invertible image, Amitsur's theorem

Digital Object Identifier 10.4230/LIPIcs.MFCS.2020.10

Acknowledgements We thank the reviewers of MFCS 2020 for their invaluable feedback.

1 Introduction

Let $\mathrm{X} = \{x_1, x_2, \ldots, x_n\}$ be a set of n free noncommuting variables and $\mathbb{F}$ be any scalar field. The *free noncommutative ring* $\mathbb{F}\langle \mathrm{X} \rangle$ is the ring of all noncommutative polynomials in X-variables over the field $\mathbb{F}$.

Noncommutative arithmetic complexity deals with the complexity of computing noncommutative polynomials in noncommutative models of computation like circuits, formulas, and branching programs. For instance, noncommutative arithmetic circuits have addition and multiplication gates, and circuit inputs are either variables from $\mathrm{X} = \{x_1, x_2, \ldots, x_n\}$ or scalars from the field $\mathbb{F}$. Multiplication gates respect its input order since the variables are noncommuting. An important research theme is polynomial identity testing (PIT) for noncommutative models of computation. It is motivated by the hope that efficient deterministic PIT algorithms in noncommutative models of computation should be substantially easier than their commutative counterparts.

Bogdanov and Wee [6] showed a randomized polynomial-time PIT algorithm for noncommutative circuits computing a polynomial of polynomially bounded degree, based on the Amitsur-Levitzki theorem [1]. This theorem states that a nonzero polynomial $p \in \mathbb{F}\langle \mathrm{X} \rangle$ of degree $< 2k$ cannot be an identity for the ring $\mathbb{M}_k(\mathbb{F})$ of $k \times k$ matrices over $\mathbb{F}$.

For noncommmutative algebraic branching programs (ABPs) there is a deterministic polynomial-time PIT algorithm in the white-box model [14]. In the black-box model, there is a quasi-polynomial time deterministic algorithm given by a quasi-polynomial size hitting set construction [9]. In contrast, for commutative algebraic branching programs efficient deterministic PIT algorithms are known only in very restricted cases.

Rational Identity Testing

More recently, Hrubeš and Wigderson [11] initiated the study of noncommutative computation with inverses which is mathematically complicated to analyze. We define noncommutative rational formulas and noncommutative rational circuits, analogous to noncommutative circuits computing polynomials, by allowing $+$, $\times$, and unary *inversion* gates. In particular, *rational formulas* (equivalently, *rational expressions*, which we use in more mathematical contexts) as usual have a tree-like structure with every non-output gate having a fanout of 1. These models computes *noncommutative rational functions* which are elements of the *free skew-field*. They introduce the *rational identity testing* (RIT) problem [11]: Given a noncommutative *formula*, determine if it is identically zero in the free skew-field of noncommutative rational functions. By definition, a rational expression r is identically zero in the free skew-field if and only if r has a nonempty domain of definition and for each $d \in \mathbb{N}$ and substitution from $\mathbb{M}_d(\mathbb{F})$ (the matrix algebra of $d \times d$ matrices over the field $\mathbb{F}$), the expression evaluates to the zero matrix if it is defined. Using techniques based on operator scaling and invariant theory, the RIT problem for noncommutative rational formulas is shown [10, 12] to be in deterministic polynomial time in the white-box model. It is also shown to be in randomized polynomial time in the black-box model [8].

The complexity of identity testing for noncommutative rational circuits in general remains unclear. Nothing better than an exponential time upper bound is known. In particular, even for rational circuits of inversion height one (inversion height of a circuit is the maximum number of inverse gates present in any input to output path in the circuit [11]), we do not know a sub-exponential time randomized algorithm.

Recently, we considered [3] noncommutative rational circuits that allow inverse gates applied only to circuit inputs. Such circuits can be seen as computing *free group algebra expressions*: that is, $\mathbb{F}$-linear combinations of words over the free group. Free group algebra

expressions are a special case of inversion height one rational circuits. For this special case we could give a randomized polynomial-time algorithm for identity testing [1]. We show, analogous to the Bogdanov-Wee algorithm for noncommutative polynomials [6], that it suffices for the algorithm to simply evaluate the given degree d free group algebra expression on random $2d \times 2d$ matrices over $\mathbb{F}$.

In general, a noncommutative rational circuit of inversion height one can be obtained as composition of a free group algebra expression with noncommutative polynomials. Thus, the next case to consider for identity testing is to allow inverses on linear forms. However, even in this case we do not have a sub-exponential time algorithm. This naturally leads us to consider an easier case of rational identity testing for algebraic branching programs whose multi-edges are labeled by affine linear forms or inverses of affine linear forms. Clearly, such ABPs compute rational expressions of inversion height one in the free skew field. The rational expression computed by the ABP is the sum over each source-to-sink path P of the ordered product of affine linear forms or their inverses labeling P. The size of the ABP is defined as the total number nodes and multi-edges. For this model a deterministic quasi-polynomial time white-box algorithm and a randomized quasi-polynomial time black-box algorithm follows respectively from [10, 12] and [8]. In this paper, we obtain deterministic polynomial time white-box algorithm and a deterministic quasi-polynomial time algorithm for the black-box model.

▶ **Theorem 1.** *Given an ABP (in white-box) of size s where each edge is labeled by an affine linear form or inverse of an affine linear form over $\mathbb{Q}$, there is a deterministic* $\mathrm{poly}(s, n)$ *time algorithm to decide if the rational expression computed by it is zero in $\mathbb{Q}\langle\langle \mathrm{X} \rangle\rangle$. If such an ABP is given as a black-box then there is a deterministic $(ns)^{O(\log(ns))}$-time algorithm for it.*

Image of Noncommutative ABPs

In the second part of the paper, we focus on the image set of noncommutative polynomials. For matrix algebra $\mathbb{M}_k(\mathbb{F})$, the image set of a noncommutative polynomial (similarly, for rational function) $f \in \mathbb{F}\langle \mathrm{X} \rangle$ is defined as the set $\mathrm{Img}_k(f) = \{f(a_1, \ldots, a_n) \mid a_1, \ldots, a_n \in \mathbb{M}_k(\mathbb{F})\}$ for some k. There is a connection between image sets of rational expressions and rational identity testing. Indeed, a rational expression r of inversion height i is defined at a matrix substitution $(a_1, \ldots, a_n) \in (\mathbb{M}_k(\mathbb{F}))^n$, precisely when for each of its subexpressions r' of inversion height $i - 1$, the matrix $r'(a_1, \ldots, a_n)$ is invertible. This connection motivates the following problem: Given a noncommutative ABP, find a matrix substitution such that the output matrix is invertible. A randomized polynomial time algorithm for this problem follows from Amitsur's theorem [2] which promises that if f is nonzero on $k \times k$ matrices then f has invertible matrices in its range. We obtain a deterministic quasi-polynomial time algorithm for this problem when $k \geq d$ where d is the degree of the polynomial.

▶ **Theorem 2.** *Given black-box access to a noncommutative polynomial $f \in \mathbb{F}\langle \mathrm{X} \rangle$ of degree d, computable by an ABP A of size s, there is a deterministic quasi-polynomial time algorithm of run time $s^{O(\log d)}$ that computes a matrix tuple $(M_1, M_2, \ldots, M_n) \in (\mathbb{M}_d(\mathbb{F}))^n$ of $d \times d$ matrices such that $f(M_1, M_2, \ldots, M_n)$ is invertible. Here the ground field $\mathbb{F}$ could be any field which is sufficiently large.*

[1] The time bound is polynomial in the maximum length d of reduced words in the expression, to be precise. We refer to d as the degree of the expression.

The Brešar-Klep Theorem

We next turn to another algorithmic question related to the image of noncommutative polynomial motivated by the following interesting theorem due to Brešar and Klep [7].

▶ **Theorem 3** (Brešar-Klep Theorem[7]). *Let $f \in \mathbb{F}\langle \mathrm{X} \rangle$ be any noncommutative polynomial, where $\mathbb{F}$ is a field of zero characteristic. Then precisely one of the following is true:*

1. $\mathrm{Img}_k(f) = 0$, *which means f is an identity for $\mathbb{M}_k(\mathbb{F})$.*
2. *The span of $\mathrm{Img}_k(f)$ consists of all scalar multiples of the identity matrix I_k (i.e., f is central for $\mathbb{M}_k(\mathbb{F})$).*
3. *The span of $\mathrm{Img}_k(f)$ is all trace zero matrices over $\mathbb{M}_k(\mathbb{F})$.*
4. *The span of $\mathrm{Img}_k(f)$ is $\mathbb{M}_k(\mathbb{F})$.*

The Brešar-Klep theorem naturally raises an algorithmic question: Given a noncommutative polynomial f and the matrix algebra $\mathbb{M}_k(\mathbb{F})$, to efficiently determine which of the four cases occur.

▶ **Proposition 4.** *Let $f \in \mathbb{Q}\langle \mathrm{X} \rangle$ be a noncommutative polynomial of degree d over rationals given by an arithmetic circuit of size s. For any matrix algebra $\mathbb{M}_k(\mathbb{Q})$ we can check in randomized time $\mathrm{poly}(s, d, k)$ which of the four conditions of the Brešar-Klep theorem hold for f over $\mathbb{M}_k(\mathbb{Q})$.*

This is easily observed by substituting the noncommuting variables with generic $k \times k$ size matrices and evaluating the commuting generic variables randomly. We show the following result which yields an efficient deterministic algorithm.

▶ **Theorem 5.** *Given a noncommutative ABP A of size s computing a polynomial $f \in \mathbb{F}\langle \mathrm{X} \rangle$ of degree d, there is a deterministic $\mathrm{poly}(n, s, d)$-time algorithm to check if $\mathrm{Img}_k(f)$ is trace zero over $\mathbb{M}_k(\mathbb{F})$ for all $k \geq d$. If A is given by black-box access, there is a deterministic $(ns)^{O(\log d)}$-time algorithm to check if $\mathrm{Img}_k(f)$ is trace zero for all $k \geq d$. Here the ground field $\mathbb{F}$ could be any field which is sufficiently large.*

The above theorem easily yields a deterministic polynomial-time algorithm to check which of the four conditions of the Brešar-Klep theorem holds for matrix algebras of dimension $k \geq d$ for a noncommutative polynomial f given by an ABP.

2 Preliminaries

Notation. The *trace* of a square matrix $A \in \mathbb{M}_t(\mathbb{F})$ is the sum of all its diagonal entries. In symbols, $\mathrm{Trace}(A) = \sum_{i=1}^{t} A[i,i]$. For an $m \times n$ matrix A and $p \times q$ matrix B, over a field $\mathbb{F}$, their *tensor product* $A \otimes B$ is an $mp \times nq$ (block) matrix obtained by replacing the $(i,j)^{th}$ entry $A[i,j]$ of A by the matrix $A[i,j]B$. For a set of noncommuting variables X, the free noncommmutative ring of polynomials over a field $\mathbb{F}$ is denoted by $\mathbb{F}\langle \mathrm{X} \rangle$. The ring of *formal power series* is denoted by $\mathbb{F}\langle\langle \mathrm{X} \rangle\rangle$. For a series (or polynomial) S, the coefficient of a monomial $m \in \mathrm{X}^*$ in S is denoted by $[m]S$. Let $\mathrm{supp}(S)$ denote the *support* of the series S: $\mathrm{supp}(S) = \{m \mid [m]S \neq 0\}$.

▶ **Definition 6** (Algebraic Branching Program). *An* algebraic branching program *(ABP) is a layered directed acyclic graph. The vertex set is partitioned into layers $0, 1, \ldots, d$, with directed edges only between adjacent layers (i to $i+1$). There is a* source *vertex of in-degree* 0 *in layer* 0, *and one out-degree-*0 sink *vertex in layer d. Each edge is labeled by an affine $\mathbb{F}$-linear form. The polynomial computed by the ABP is the sum over all source-to-sink directed paths of the ordered product of affine forms labeling the path edges.*

The *size* of the ABP is defined as the total number of nodes and multi-edges and *width* is the maximum number of nodes in a layer. The ABP model is defined for computing commutative or noncommutative polynomials. ABPs of width w can also be seen as iterated matrix multiplication $\boldsymbol{u}^T M_1 M_2 \dots M_\ell \boldsymbol{v}$, where $\boldsymbol{u}, \boldsymbol{v}$ are $w \times 1$ vectors and each M_i is a $w \times w$ matrix, whose entries are affine linear forms in variables X.

We also consider commutative set-multilinear polynomials. Here, the (commutative) variable set is partitioned as $\mathrm{Y} = \mathrm{Y}_1 \sqcup \mathrm{Y}_2 \sqcup \cdots \sqcup \mathrm{Y}_d$ where for each $j \in [d]$, $\mathrm{Y}_j = \{y_{i,j}\}_{i=1}^n$. A polynomial f is set-multilinear if it is homogeneous degree d such that each nonzero monomial m is of the form $m = y_{i_1,1} y_{i_2,2} \dots y_{i_d,d}$.

Given a homogeneous degree d noncommutative polynomial f, its set-multilinearization $\mathrm{SM}(f)$ is the corresponding set-multilinear polynomial obtained by replacing x_i in the j^{th} position (in a monomial) by $y_{i,j}$ in every monomial. Clearly, $f \equiv 0$ if and only if $\mathrm{SM}(f) \equiv 0$.

We recall two well-known PIT results of noncommutative ABPs.

▶ **Theorem 7** (Raz-Shpilka [14]). *Given an ABP of width w and d many layers computing a polynomial $f \in \mathbb{F}\langle \mathrm{X} \rangle$, there is a deterministic* $\mathrm{poly}(w, d, n)$ *time algorithm to test whether $f \equiv 0$ or not.*

For black-box case, Forbes and Shpilka [9], have a shown an efficient construction of quasi-polynomial size hitting set for noncommutative ABPs. Consider the class of noncommutative ABPs of width w, and depth d computing polynomials in $\mathbb{F}\langle \mathrm{X} \rangle$. The result of Forbes-Shpilka provide an explicit construction (in quasi-polynomial time) of a set $\mathcal{H}_{w,d,n}$ contained in $\mathbb{M}_{d+1}(\mathbb{F})$, such that for any ABP (with parameters w and d) computing a nonzero polynomial f, there always exists $\boldsymbol{\alpha} \in \mathcal{H}_{w,d,n}$ such that $f(\boldsymbol{\alpha}) \neq 0$.

▶ **Theorem 8** (Forbes-Shpilka [9]). *For all $w, d, n \in \mathbb{N}$, if $|\mathbb{F}| \geq \mathrm{poly}(d, n, w)$, then there is a hitting set $\mathcal{H}_{w,d,n} \subset \mathbb{M}_{d+1}(\mathbb{F})$ for noncommutative ABPs of parameters w, d, n such that $|\mathcal{H}_{w,d,n}| \leq (wdn)^{O(\log d)}$ and there is a deterministic algorithm to output the set $\mathcal{H}_{w,d,n}$ in time $(wdn)^{O(\log d)}$.*

There is an extension of this construction to commutative set-multilinear polynomials computed by ABPs where layers respect the variable partition [9]. We will use this result in Section 5.

Automata Theory. We recall some automata theory. More details can be found in the Berstel-Reutenauer book [5].

Let K be a semiring and X be an alphabet [2]. A K-weighted automaton over X is a 4-tuple, $\mathcal{A} = (Q, I, E, T)$, where Q is a finite set of states, and the mappings $I, T : Q \to K$ are weight functions for entering and leaving a state respectively, and $E : Q \times \mathrm{X} \times Q \to K$ is the weight of each transition. We define $|Q|$, the number of states, to be the size of the automaton. A path is a sequence of edges : $(q_0, a_1, q_1)(q_1, a_2, q_2) \dots (q_{t-1}, a_t, q_t)$. The weight of the path is the product of the weights of the edges. The formal series $S \in K\langle\langle \mathrm{X} \rangle\rangle$ which is the (possibly infinite) sum of the weights over all the paths that are *recognized* by $\mathcal{A}$. Then, for each word $w = a_1 a_2 \cdots a_t \in \mathrm{X}^*$, the contribution of all the paths for the word w is given by $[w]S = \sum_{q_0,\dots,q_t \in Q} I(q_0) \cdot E(q_0, a_1, q_1) \cdots E(q_{t-1}, a_t, q_t) \cdot T(q_t)$.

A K-weighted automaton $\mathcal{A}$ with *ϵ-transitions* over X is defined with E modified, such that $E : Q \times \{\mathrm{X} \cup \epsilon\} \times Q \to K$. Let $A_0 \in \mathbb{M}_{|Q|}(K)$ be the transition matrix for the ϵ-transitions.

[2] We interchangeably use X as a variable set of ABPs and as alphabet symbol of weighted automata.

It is well-known that if $\sum_k A_0^k$ converges, then another automaton $\mathcal{A}'$ without ϵ-transitions computing the same series can be constructed [13]. By definition, such automaton is said to be *valid* if $\sum_k A_0^k$ converges.

The following basic result by Schützenberger [15] is key to transform zeroness testing of weighted automata to identity testing of ABPs.

▶ **Theorem 9** (Schützenberger). *Let K be a subring of a division ring and $\mathcal{A}$ be a K-weighted automaton without any ϵ-transition with s states computing a series S in $K\langle\langle \mathrm{X} \rangle\rangle$. Then S is a nonzero series if and only if there is a word $w \in \mathrm{X}^*$ of length at most $s-1$, such that $w \in \mathrm{supp}(S)$.*

3 Identity Testing of ABPs with Inverse of Linear forms

In this section we prove Theorem 1. In the generalized ABP model we allow directed multi-edges from nodes in layer i to layer $i+1$ (we allow multiple edges between the same pair of nodes), where each edge is labeled by some affine linear form or the inverse of an affine linear form. Recall that, the size s is the total number of nodes and the multi-edges present in the ABP.

A simple fact about formal power series that we use is replacing the rational expression $(1-x)^{-1}$ by the power series x^*, which is used to convert an ABP where edges are labeled by linear forms and its inverses to an automaton computing a formal series. [3] In general, an affine linear form may have zero constant term. In order to apply the above, we require a linear shift $x_j \mapsto \alpha_j - x_j$, $j \in [n]$, enabling power series expansion of the inverses of the linear forms. The following lemma (proof omitted) explains the efficient finding of such linear shifts.

▶ **Lemma 10.** *Let $\mathbb{F}$ be a field such that $|\mathbb{F}| \geq nr+1$. We can efficiently construct a subset $S \subseteq \mathbb{F}^n$ of size $nr+1$ such that for any r affine linear forms $L_1, \ldots, L_r$ over X, there is a point $\boldsymbol{\alpha} \in S$ such that for all i, $L_i(\boldsymbol{\alpha}) \neq 0$.*

The following lemma shows that rational identity testing of such ABPs is efficiently reducible to zero testing of a weighted automaton computing a formal series in $\mathbb{F}\langle\langle \mathrm{X} \rangle\rangle$.

▶ **Lemma 11.** *Let A be a generalized ABP of size s with each edge labeled by either an affine linear form or the inverse of an affine linear form, computing a rational expression f in $\mathbb{F}\langle\langle \mathrm{X} \rangle\rangle$. Let r be the total number of multi-edges of A. Then, there is an automaton A' without ϵ-transitions of size at most $s+r$ computing a formal series in $\mathbb{F}\langle\langle \mathrm{X} \rangle\rangle$ such that f is an identity in $\mathbb{F}\langle\langle \mathrm{X} \rangle\rangle$ if and only if A' computes a zero series in $\mathbb{F}\langle\langle \mathrm{X} \rangle\rangle$. Moreover, A' can be constructed in $\mathrm{poly}(n, s, r)$ time.*

Proof. We present the proof in two parts. We first explain the automaton construction. Then, we show that this construction is identity preserving.

Construction of the Automaton: Let $L_1, L_2, \ldots, L_r$ be all the linear forms appearing as L_i or L_i^{-1} in A. By Lemma 10, we can efficiently compute a set $S \subseteq \mathbb{F}^n$ of size $nr+1$ such that there exists a point $\boldsymbol{\alpha} \in S$ such that for each $i \in [r]$, $L_i(\alpha_1 - x_1, \ldots, \alpha_n - x_n)$ has a nonzero constant term. Fix such a tuple $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_n) \in S$. We apply the linear shift $x_j \mapsto \alpha_j - x_j$ to each edge label of A, and let g denote the rational expression in $\mathbb{F}\langle\langle \mathrm{X} \rangle\rangle$ computed by the resulting ABP.

[3] Notice that by the Kleene expression x^* is meant the formal power series $1 + x + x^2 + \cdots$ instead of the set $\{\epsilon, x, x^2 \ldots\}$. In this sense, in general, we will consider weighted automaton as evaluating to a formal power series.

As each $L_i(\alpha_1 - x_1, \ldots, \alpha_n - x_n)$ has a constant term, we may write it as $\beta_i(1 - \tilde{L}_i)$, for a homogeneous linear form $\tilde{L}_i$, where $\beta_i \neq 0$. We can convert $(1 - \tilde{L}_i)^{-1}$ to the formal power series $\tilde{L}_i^*$ to obtain $L_i^{-1} = \beta_i^{-1}\tilde{L}_i^*$. Thus, any edge labeled L_i^{-1} can be labeled by a Kleene-$*$ expression. From this observation, we now show that g can also be converted to a formal series in $\mathbb{F}\langle\langle \mathrm{X} \rangle\rangle$ computed by a small automaton.

This is a standard adaptation of Kleene's original construction. We locally substitute each $*$-expression by a small automaton. We illustrate this with an example. Consider the edge shown in Figure 1 having linear form and inverses. In Figure 2, we convert the linear forms with inverses to $*$-expressions by the linear shift $x_j \mapsto 1 - x_j$. Finally, in Figure 3, we show the transitions of an equivalent automaton by replacing the $*$-rational expressions by their corresponding automata.

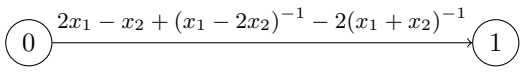

Figure 1 Edge Labels having linear form and inverses.

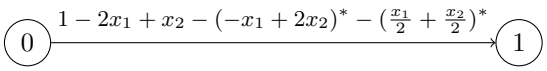

Figure 2 Edge Labels rewritten as $*$-rational expression after applying the shift $x_i \mapsto 1 - x_i$.

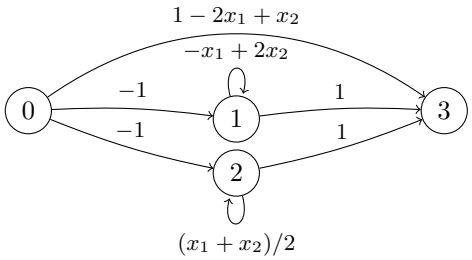

Figure 3 Edge Labels replaced by an appropriate automaton.

It is useful to consider the transition matrix M for the final automaton. In the current example this is given by the following matrix.

$$M = \begin{bmatrix} 0 & -1 & -1 & 1 - 2x_1 + x_2 \\ 0 & -x_1 + 2x_2 & 0 & 1 \\ 0 & 0 & \frac{x_1 + x_2}{2} & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}.$$

Clearly, applying the above transformation to each edge of the input ABP A produces an automaton $\tilde{A}$ of size at most $s + r$, because we introduce a new node in the automaton for each L^{-1} term. Moreover, $\tilde{A}$ can be constructed in $\mathrm{poly}(n, s, r)$ time.

▷ Claim 12. $\tilde{A}$ computes a valid formal series in $\mathbb{F}\langle\langle \mathrm{X} \rangle\rangle$.

Proof. Consider the transition matrix of the automaton A_0 corresponding to the ϵ-transitions. To show that $\tilde{A}$ computes a valid formal series in $\mathbb{F}\langle\langle \mathrm{X} \rangle\rangle$, it suffices to prove that $\sum_k A_0^k$ converges (Proposition 2 in [13]). As the automaton introduces self-loops labeled by homogeneous linear forms only, and it does not have back-edges, the matrix A_0 is strictly upper triangular (see the above example). Hence, A_0 is nilpotent and $\sum_k A_0^k$ converges. ◁

As mentioned in Section 2, by a standard construction we can compute an automaton A' without ϵ-transitions equivalent of $\tilde{A}$ [13]. The overall time to construct A' is bounded by $\text{poly}(n, s, r)$.

Identity Preserving

▷ Claim 13. $A \not\equiv 0$ in $\mathbb{F}\langle\langle \mathrm{X} \rangle\rangle$ if and only if A' does not compute a zero series in $\mathbb{F}\langle\langle \mathrm{X} \rangle\rangle$.

Proof. Let f be a nonzero rational expression in $\mathbb{F}\langle\langle \mathrm{X} \rangle\rangle$ computed by A. Then, for some $t \in \mathbb{N}$ and matrix tuple $(M_1, \ldots, M_n) \in (\mathbb{M}_t(\mathbb{F}))^n$, we have $f(M_1, \ldots, M_n) \neq 0$. Therefore, $g = f(\alpha_1 - x_1, \ldots, \alpha_n - x_n)$ is also a nonzero rational expression in $\mathbb{F}\langle\langle \mathrm{X} \rangle\rangle$ as $g(M'_1, \ldots, M'_n) \neq 0$, where $M'_j = \alpha_j I_t - M_j$ for each $j \in [n]$.

To prove that A' computes a nonzero series, it suffices to show that for some matrix substitution A' is defined and outputs a nonzero matrix on that substitution. In g, each affine linear form with inverse looks like $\beta_i(1 - \tilde{L}_i)^{-1}$ where β_i is nonzero and $\tilde{L}_i$ is a homogeneous linear form. Now, let $N_i = \tilde{L}_i(M'_1, \ldots, M'_n)$. Since g is defined and nonzero at the point $(M'_1, \ldots, M'_n)$, the matrix $(I_t - N_i)$ is invertible for each i. But it may happen that for some $j \in [r]$, the matrix $\sum_k N_j^k$ does not converge and hence A' is not defined at this matrix tuple. To avoid this problem, we can choose $\gamma \in \mathbb{Q}$ sufficiently small ensuring that $g(\gamma M'_1, \ldots, \gamma M'_n)$ is still defined and nonzero, moreover, for each $i \in [r]$, the matrix $\sum_k N_i^k$, thus obtained, also converges. The following fact is classical and a proof of it is, for example, in [16].

▶ **Fact 1.** *For any matrix B over $\mathbb{Q}$, the Neumann series $\sum_k B^k$ converges if the spectral norm of B is less than* 1.

▶ **Observation 1.** *Let g be a rational expression in $\mathbb{F}\langle\langle \mathrm{X} \rangle\rangle$ and suppose $g(M'_1, \ldots, M'_n) \neq 0$ for some $t \times t$ matrices M'_i. Then there are only finitely many $\gamma \in \mathbb{F}$ for which $g(\gamma M'_1, \ldots, \gamma M'_n)$ is not defined or $g(\gamma M'_1, \ldots, \gamma M'_n) = 0$.*

Proof. Let us think the parameter γ as indeterminate and note that the output matrix $g(\gamma M'_1, \ldots, \gamma M'_n)$ is a $t \times t$ matrix, where each entry is a commutative rational function of form $\frac{h_1}{h_2}$ and h_1 and h_2 are univariate polynomials in γ. The degree of each such h_1, h_2 is some finite value depending on the rational expression g. Clearly, it is not a zero matrix in $\mathbb{M}_k(\mathbb{F}(\gamma))$, as for $\gamma = 1$, it is nonzero. Hence, to ensure that $g(\gamma M'_1, \ldots, \gamma M'_n)$ is defined and nonzero, it suffices to avoid the roots of the univariates of each entry. ◀

By Observation 1, we can choose γ small enough such that, for each $i \in [r]$, spectral norm of N_i is less than 1. By Fact 1, the automaton A' is also defined and nonzero on $(\gamma M'_1, \ldots, \gamma M'_n)$. Therefore, A' computes a nonzero series.

Conversely, suppose that A' computes a nonzero series in $\mathbb{F}\langle\langle \mathrm{X} \rangle\rangle$. Consider any word w such that $[w]A' \neq 0$. Then consider the automaton $\mathcal{A}$ that accepts only the word w and let $A_1, \ldots, A_n$ be the transition matrices of the automaton $\mathcal{A}$ for the variables $x_1, \ldots, x_n$. It can be easily observed that $A'(A_1, \ldots, A_n)$ is well-defined and a nonzero matrix whose top right-most entry is $[w]A'$ [see [4] for details]. Since whenever A' converges on a point, so does g, we conclude that $g(A_1, \ldots, A_n) \neq 0$, which also implies that $f(\alpha_1 I_t - A_1, \ldots, \alpha_n I_t - A_n) \neq 0$. Hence f is nonzero in $\mathbb{F}\langle\langle \mathrm{X} \rangle\rangle$. ◁

Now the proof of the lemma follows. ◀

Proof of Theorem 1. We now present the algorithms for white-box and black-box models.

The White-Box Case Let r be the total number of linear forms or inverses of linear forms in A. Clearly, r is bounded by s. Using Lemma 11, we reduce the problem of deciding whether the ABP A is zero in $\mathbb{F}\langle\langle \mathrm{X} \rangle\rangle$ to the problem of deciding whether the automaton A' is computing a zero series or not in $\mathbb{F}\langle\langle \mathrm{X} \rangle\rangle$. From Lemma 11, A' is of size W which is at most $2s$. Now invoking Theorem 9, we conclude that $A' \not\equiv 0$ if and only if there is a word of length at most $W-1$ which has nonzero coefficient in A'. Consider the corresponding transition matrix $M_{A'}$ of A'. For each $\ell \leq W-1$, we construct the branching program $B^{(\ell)} = \boldsymbol{u}^T M_{A'}^{\ell} \boldsymbol{v}$ where $\boldsymbol{u}, \boldsymbol{v}$ are the vectors corresponding to the initial states and final states respectively. As A' does not have any ϵ-transitions, $B^{(\ell)}$ computes words in A' of length exactly ℓ. It suffices to check for each $\ell \leq W-1$, whether $B^{(\ell)}$ computes an identically zero polynomial. The identity testing algorithm is obtained by applying Theorem 7 on the ABPs $B^{(\ell)}$. The running time of the algorithm is clearly bounded by $\mathrm{poly}(n, s)$.

The Black-Box Case We now present a deterministic quasi-polynomial time black-box algorithm. Let r be the total number of linear forms or inverses of linear forms in A. Clearly, r is bounded by s. Lemma 10 yields a set S of size $nr+1$ such that for some $\boldsymbol{\alpha} \in S$, the linear shift $x_j \mapsto \alpha_j - x_j$ ensures that for every edge label, each of the r many L^{-1} in A, the linear form L has a nonzero constant term. Let us fix such $\boldsymbol{\alpha} \in S$. From the proof of Lemma 11, we conclude that there is an automaton A' of size at most $2s$ such that A is zero in $\mathbb{F}\langle \mathrm{X} \rangle$ if and only if A' computes a zero series in $\mathbb{F}\langle\langle \mathrm{X} \rangle\rangle$.

Let $A'^{(\ell)}$ denotes the series computed by A' truncated to the words of length at most ℓ. Let $W = 2s$. Now, by Theorem 9, $A' \not\equiv 0$ if and only if $A'^{(W-1)} \not\equiv 0$. We now discuss the effect of $\mathcal{H}_{W^2,W-1,n}$, hitting set from Theorem 8 on A'. It is well known from the proof of Theorem 8 that for each $(h_1, \ldots, h_n) \in \mathcal{H}_{W^2,W-1,n}$, each h_i is a $W \times W$ matrix of the following form [9]:

$$h_i = \begin{bmatrix} 0 & a_1 & 0 & \cdots & 0 \\ 0 & 0 & a_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & a_{W-1} \\ 0 & 0 & \cdots & 0 & 0 \end{bmatrix}.$$

Using the shape of the matrices h_i, it can be easily checked that for all words $w \in \mathrm{X}^*$ of length at least W, $w(h_1, \ldots, h_n) = 0$. Hence, evaluating A' at some $(h_1, \ldots, h_n) \in \mathcal{H}_{W^2,W-1,n}$ is equivalent to evaluating $A'^{(W-1)}$ at $(h_1, \ldots, h_n)$. As already discussed in the previous section for white-box case, we can construct the branching program $B^{(\ell)} = \boldsymbol{u}^T M_{A'}^{\ell} \boldsymbol{v}$ computing words of length exactly ℓ in A', for each $\ell \leq W-1$, where $M_{A'}$ is the corresponding transition matrix of automaton A' and $\boldsymbol{u}, \boldsymbol{v}$ are the vectors corresponding to the initial states and final states respectively. Hence, $A'^{(W-1)}$ can be computed by an ABP of width at most W^2. Therefore, A' computes a zero series if and only if for each $(h_1, \ldots, h_n) \in \mathcal{H}_{W^2,W-1,n}$, $A'(h_1, \ldots, h_n)$ outputs a zero matrix. Hence, by evaluating A on $(\alpha_1 I_W - h_1, \ldots, \alpha_n I_W - h_n)$ for each $\alpha \in S$ and $(h_1, \ldots, h_n) \in \mathcal{H}_{W^2,W-1,n}$, we can decide A is zero in $\mathbb{F}\langle \mathrm{X} \rangle$ or not. ◀

4 Invertible Image of Noncommutative ABPs

Let us first fix some notation for the subsequent sections. S_d denotes the set of permutations $\{\sigma : [d] \to [d]\}$. For a degree-d word $m = x_{i_1} \cdots x_{i_d} \in \mathrm{X}^*$, we define *$\sigma$-permuted word*

$m^\sigma = x_{i_{\sigma(1)}} \cdots x_{i_{\sigma(d)}}$. For a degree-$d$ homogeneous noncommutative polynomial $g \in \mathbb{F}\langle \mathrm{X} \rangle$, g^σ is defined as $g^\sigma = \sum_{m \in \mathrm{supp}(g)} [m] g \cdot m^\sigma$. For each $j \in \{0, 1, \ldots, d-1\}$, $\sigma_j \in S_d$ denotes the permutation that cyclically rotates a monomial right to left by j steps. As a permutation $\sigma_j = (j+1, j+2, \ldots, d, 1, \ldots, j)$.

As preparation, we show in the following lemma (proof omitted) that each *cyclic shift* of a noncommutative homogeneous ABP of size s can be computed by an ABP of size polynomial in s.

▶ **Lemma 14.** *Let A be a homogeneous ABP of size s computing a noncommutative polynomial $g \in \mathbb{F}\langle \mathrm{X} \rangle$ of degree d. For each $j \in [d-1]$, there is a $O(s^2)$ size ABP computing g^{σ_j}.*

Proof of Theorem 2. We first explain the proof for homogenous degree d ABPs. For each $i \in [n]$ construct the following matrices

$$M_i = \begin{bmatrix} 0 & y_{i,1} & 0 & \cdots & 0 \\ 0 & 0 & y_{i,2} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & y_{i,(d-1)} \\ y_{i,d} & 0 & 0 & \cdots & 0 \end{bmatrix}_{d \times d} . \tag{1}$$

It is possible to view these matrices as the transition matrices of a labeled automaton. We observe that for a monomial $m = x_{i_1} x_{i_2} \cdots x_{i_d}$, the matrix $m(M_1, \ldots, M_n)$ is a diagonal matrix. Moreover, for any $j \in [d]$, the $(j,j)^{th}$ entry is given by $(y_{i_1,j} y_{i_2,j+1} \cdots y_{i_{d-(j-1)},d})(y_{i_{d-(j-2)},1} \cdots y_{i_d,j-1})$. Since Y is a set of commutative variables, the above is same as $\mathrm{SM}(m^{\sigma_{d-(j-1)}})$ [4].

Thus, for any homogeneous degree d polynomial f, by linearity we get that $f(M_1, \ldots, M_n)$ is also a diagonal matrix and the $(j,j)^{th}$ entry is $\mathrm{SM}(f^{\sigma_{d-(j-1)}})$.

The image of the polynomial f is invertible on a point $(M_1, \ldots, M_n)$, if and only if $\det(f(M_1, \ldots, M_n)) \neq 0$. Further if the shape of each M_i is as described in Equation 1, we have $\det(f(M_1, \ldots, M_n)) = \prod_{j=0}^{d-1} \mathrm{SM}(f^{\sigma_j})$. Note that, if the noncommutative polynomial f is nonzero then for each $\sigma_j \in S_d$, f^{σ_j} is also nonzero. Recall that, for any f, f is nonzero if and only if $\mathrm{SM}(f)$ is nonzero. Hence, given a nonzero polynomial f, $\det(f(M_1, \ldots, M_n))$ is a nonzero polynomial as every diagonal entry evaluates to a nonzero commutative polynomial.

Since f^{σ_0} has an ABP of size s, each cyclic shift f^{σ_j} has an ABP of size $O(s^2)$ by Lemma 14. Therefore, the set-multilinearization $\mathrm{SM}(f^{\sigma_0})$ has an ABP of size s, and each $\mathrm{SM}(f^{\sigma_j})$ has an ABP of size $O(s^2)$, over the same variable partition $\mathrm{Y} = \mathrm{Y}_1 \sqcup \mathrm{Y}_2 \sqcup \cdots \sqcup \mathrm{Y}_d$. It is obtained by making the input ABP set-multilinear i.e. by replacing each x_i variable in the j^{th} layer by $y_{i,j}$.

Now we briefly discuss how to use a generator of Forbes-Shpilka [9] for set-multilinear ABPs to complete the algorithm. Let $\mathcal{G} : \mathbb{F} \mapsto \mathbb{F}^{nd}$ be the hitting set generator for the set-multilinear algebraic branching programs of size $O(s^2)$ over the variable set $\mathrm{Y} = \mathrm{Y}_1 \sqcup \ldots \sqcup \mathrm{Y}_d$ with d layers promised by the result in [9]. The map $\mathcal{G} : z \mapsto (p_{1,1}(z), p_{1,2}(z), \ldots, p_{n,d}(z))$ is a polynomial map where each $p_{i,j}$ is of degree at most $D = (snd)^{O(\log d)}$ with the property that $\mathrm{SM}(f^{\sigma_j}) \circ \mathcal{G}$ is nonzero univariate if and only if $f^{\sigma_j} \not\equiv 0$. Thus, to prove that $f(M_1, \ldots, M_n)$

[4] For any homogeneous noncommutative polynomial f, recall the definition of $\mathrm{SM}(f)$ from Section 2.

is invertible, it suffices to show

$$\prod_{j=0}^{d-1}(\mathrm{SM}(f^{\sigma_j})\circ\mathcal{G}) = \left(\prod_{j=0}^{d-1}\mathrm{SM}(f^{\sigma_j})\right)\circ\mathcal{G} = \det(f(M_1,\ldots,M_n))\circ\mathcal{G} \not\equiv 0. \tag{2}$$

Now we note that $\det(f(M_1,\ldots,M_n))\circ\mathcal{G}(z)$ is a univariate of degree at most d^2D. Thus to test equation (1) for identity, it suffices to go over d^2D+1 distinct values of z. More precisely, we choose distinct field elements $\alpha_1,\ldots,\alpha_{d^2D+1}\in\mathbb{F}$ and construct the hitting set,

$$\mathcal{H} = \{(p_{1,1}(\alpha_i),\ldots,p_{n,d}(\alpha_i))|i\in[d^2D+1]\},$$

In case, the given ABP of degree d is not homogeneous, then the substitution $x_i = tM_i$ is performed where t is a commutative variable. The words (monomials) of degree i produce terms with t-degree i. Now $\det(f(tM_1,tM_2,\ldots,tM_n))$ will have a term with t-degree d^2 which is produced by the identity permutation and no other permutations can produce a term of same t-degree [5]. Thus using Forbes-Shpilka generator $\mathcal{G}$, we know that for some $z=\alpha^0$, the polynomial $\det(f(tM_1,tM_2,\ldots,tM_n))\circ\mathcal{G}|_{z=\alpha^0}$ is a nonzero univariate in t of degree d^2 and hence it suffices to try the d^2+1 distinct substitution for t such that the final output becomes nonzero on one such substitution $t=t_0$. ◀

▶ Remark 15. In the white-box case (when f is given by an ABP) we can find a matrix substitution $(M_1,M_2,\ldots,M_n)$ in deterministic polynomial time, applying the theory of matrix pencils as developed in [11, 12]. The proof will appear in the full version of the paper.

5 Trace of Image of an Algebraic Branching Program

We now prove Theorem 5. By the following lemma it suffices to show it for homogeneous ABPs. Recall that the image of f over matrix algebra $\mathbb{M}_k(\mathbb{F})$ is defined as $\mathrm{Img}_k(f) = \{f(M_1,\ldots,M_n) \mid (M_1,\ldots,M_n)\in(\mathbb{M}_k(\mathbb{F}))^n\}$, and $\mathrm{Trace}(\mathrm{Img}_k(f))$ is the set of traces of the matrices in $\mathrm{Img}_k(f)$. Use $\boldsymbol{M}$ to denote the matrix tuple $(M_1,\ldots,M_n)$. We say $\mathrm{Trace}(\mathrm{Img}_k(f)) = \{0\}$ if and only if $\mathrm{Trace}(f(\boldsymbol{M})) = 0$ for each $\boldsymbol{M}\in(\mathbb{M}_k(\mathbb{F}))^n$.

▶ Lemma 16. *Let $f\in\mathbb{F}\langle X\rangle$ be a noncommutative polynomial of degree d, and f_i be its homogeneous degree-i component for each $i\in\{0,1,\ldots,d\}$. Then for all $k\in\mathbb{N}$, $\mathrm{Trace}(\mathrm{Img}_k(f)) = \{0\}$ if and only if $\mathrm{Trace}(\mathrm{Img}_k(f_i)) = \{0\}$ for each i.*

Proof. Consider the substitution $x_i\mapsto z\cdot x_i$ for a commuting variable z. We can write

$$\mathrm{Trace}(f(zM_1,\ldots,zM_n)) = \mathrm{Trace}\left(\sum_{i=1}^{d} f_i(\boldsymbol{M})z^i\right) = \sum_{i=1}^{d}\mathrm{Trace}(f_i(\boldsymbol{M}))z^i.$$

If $\mathrm{Trace}(\mathrm{Img}_k(f_i)) = \{0\}$ for each $i\in[d]$, then clearly $\mathrm{Trace}(\mathrm{Img}_k(f)) = \{0\}$. Suppose $\mathrm{Trace}(\mathrm{Img}_k(f_i))\neq\{0\}$ for some i. Then there is a matrix tuple $\boldsymbol{M}$ such that $\mathrm{Trace}(f(zM_1,\ldots,zM_n))$ is a nonzero univariate in z. Hence, there is a substitution $z=\alpha$ for which $\mathrm{Trace}(f(\alpha M_1,\ldots,\alpha M_n))\neq 0$ which shows that $\mathrm{Trace}(\mathrm{Img}_k(f))\neq\{0\}$. ◀

Proof of Theorem 5. By Lemma 16 and the fact that homogeneous components can be extracted efficiently both in the black-box and in the white-box setting, we can assume the given ABP is homogeneous of degree d. First we prove that if $\mathrm{Img}_k(f)$ is trace zero for some $k\geq d$, then the coefficients of f have the following symmetry property.

[5] Because the non-diagonal entries of the output matrix contain the terms with t degree $\leq d-1$.

▶ Lemma 17. *For a homogeneous polynomial $g \in \mathbb{F}\langle X\rangle$ of degree d, $\mathrm{Trace}(\mathrm{Img}_k(g)) = \{0\}$ at any dimension $k \geq d$, if and only if for every monomial m we have $\sum_{\sigma\in C_d}[m^\sigma]g = 0$, where $C_d = \{\sigma_0, \sigma_1, \ldots, \sigma_{d-1}\}$ is the set of all d cyclic shift permutations.*

Proof. Let M_d be the set of all monomials in g. Let M'_d denote a maximal subset of M_d constructed as follows: Group the monomials in M_d such that monomials in the same group are cyclic shifts of each other. Now define M'_d by taking one monomial from each such group. For any matrix tuple $\boldsymbol{M}$, by the cyclic property of trace, for each $\sigma \in C_d$, $\mathrm{Trace}(m^\sigma(\boldsymbol{M}))$ is same. Hence, we may write,

$$\mathrm{Trace}(g(\boldsymbol{M})) = \sum_{m\in M_d}[m]g \cdot \mathrm{Trace}(m(\boldsymbol{M})) = \sum_{m\in M'_d}\left(\sum_{\sigma\in C_d}[m^\sigma]g\right)\cdot \mathrm{Trace}(m(\boldsymbol{M})).$$

If for every monomial m we have $\sum_{\sigma\in C_d}[m^\sigma]g = 0$ then $\mathrm{Trace}(\mathrm{Img}_k(g)) = \{0\}$ for each k. For the converse direction, suppose there is a monomial m such that $\sum_{\sigma\in C_d}[m^\sigma]g \neq 0$. Let $m = x_{i_1}x_{i_2}\ldots x_{i_d}$. Construct an automaton that accepts only the cyclic shifts of m. Below, we give an illustrative example for $d = 6$.

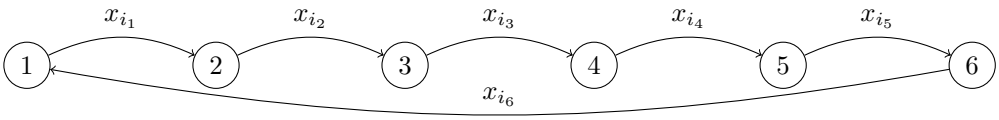

Figure 4 Example of the automata when $d = 6$.

The permutation σ_0 is the identity permutation. When the start state and final state are both $j \in [d]$, then only the word $m^{\sigma_{j-1}} = x_{i_j}x_{i_{j+1}}\cdots x_{i_d}x_{i_1}x_{i_2}\cdots x_{i_{j-1}}$ is accepted (also note that if start state and final state are different then no word of length d is accepted). The transition of the automata gives us $d \times d$ matrices $M_{x_{i_1}}, \ldots, M_{x_{i_d}}$ where $M_{x_{i_j}}(k, \ell) = 1$ if $k = j$ and $\ell = j{+}1 \pmod d$ and 0 otherwise. Substituting $x_{i_j} = M_{x_{i_j}}$ and setting $M_{x_t} = [0]$ if $x_t \notin \{x_{i_1}, x_{i_2}\ldots x_{i_d}\}$, we observe that the matrix $g(M_{x_1}, \ldots, M_{x_n})$ is a diagonal matrix and the $(j, j)^{th}$ entry is $[m^{\sigma_{j-1}}]g$, and thus $\mathrm{Trace}(g(M_{x_1}, \ldots, M_{x_n})) = \sum_{\sigma\in C_d}[m^\sigma]g \neq 0$. ◀

As a corollary of Lemma 17, we obtain the following.

▶ Corollary 18. *For all matrix substitution of dimension $k \geq d$, $\mathrm{Trace}(\mathrm{Img}_k(g)) = \{0\}$ if and only if $\sum_{\sigma\in C_d} g^\sigma \equiv 0$.*

Proof. Observe that $g^\sigma = \sum_m [m]g \cdot m^\sigma$. Hence $\sum_{\sigma\in C_d} g^\sigma = \sum_{\sigma\in C_d}\sum_m [m]g \cdot m^\sigma = \sum_m(\sum_{\sigma\in C_d}[m^\sigma]g)\cdot m$. Then the proof follows from Lemma 17. ◀

Now we would like to check if the input polynomial f computed by the given ABP has the above property. It turns out that if the input polynomial f is given as white-box, a combination of Lemma 14 and Theorem 7 can easily yield a deterministic polynomial-time algorithm.

The White-Box Case

By Lemma 14 we see that, for each $\sigma \in C_d$, f^σ can be computed by an algebraic branching program of size $O(s^2)$ and hence $\hat{f} = \sum_{\sigma\in C_d} f^\sigma$ can be computed by an algebraic branching program of size $\mathrm{poly}(s, d)$. Given the algebraic branching program A the algorithm computes the algebraic branching program $\hat{A} = \sum_{\sigma\in C_d} f^\sigma$ using Lemma 14 and runs the algorithm of

Raz and Shpilka [14] on the ABP $\hat{A}$ and outputs *trace zero* if $\hat{A} \equiv 0$. The correctness of the algorithm follows from Lemma 17, and the run time of the algorithm is $\text{poly}(n, s, d)$.

The Black-Box Case

The main idea is to obtain black-box access to $\text{SM}(\hat{f})$ where $\hat{f} = \sum_{\sigma \in C_d} f^{\sigma}$ (following notation of Corollary 18). Thereafter, one can use the standard hitting set [9] for set-multilinear ABPs over the variable partition $\text{Y} = \text{Y}_1 \sqcup \text{Y}_2 \sqcup \cdots \sqcup \text{Y}_d$. Now, for each $i \in [n]$, we construct the $d \times d$ matrix M_i as shown in the proof of Theorem 2 (If $k > d$, we adjust each M_i by padding zeros).

▶ Lemma 19. $\text{Trace}(f(M_1, \ldots, M_n)) = \text{SM}(\hat{f})$.

The proof of the lemma follows quite easily. it will be given in the full version of the paper.

Using Lemma 14, the ABP size of $\hat{f}$ is at most $\text{poly}(s, d)$. Also we conclude that $\text{SM}(\hat{f})$ has a set-multilinear ABP of depth d in the variable partition $\text{Y}_1 \sqcup \text{Y}_2 \sqcup \cdots \sqcup \text{Y}_d$ of size at most $\text{poly}(s, d)$. Now the algorithm substitutes $y_{i,j}$ from the hitting set of the set-multilinear ABPs of size s over the variable partition $\text{Y}_1 \sqcup \text{Y}_2 \sqcup \cdots \sqcup \text{Y}_d$ with d many layers [9] and evaluates the polynomial on the matrices M_i and checks whether the trace of the output matrix is always zero or not. The correctness follows from Corollary 18 and the run time follows from Theorem 8 when applied to the set-multilinear case. ◀

▶ Corollary 20. *Let f be a degree-d noncommutative polynomial in $\mathbb{F}\langle \text{X} \rangle$ computed by a size s ABP. For $k \geq d$, when f is given by an ABP (the white-box case) we can check in deterministic polynomial time which of the four cases of the Brešar-Klep theorem holds for f. For $k \geq d$, when f is given only by black-box access, we can check all the possibilities in deterministic quasi-polynomial time.*

Proof. If $k \geq d$, by the Amitsur-Levitzki theorem a nonzero f is not an identity for $\mathbb{M}_k(\mathbb{F})$. To rule out the second case notice that if f is a *central polynomial* for $\mathbb{M}_k(\mathbb{F})$ then $g = zf - fz$ is an identity for $\mathbb{M}_k(\mathbb{F})$ where z is a new noncommutative variable. This is also not possible by Amitsur-Levitzki theorem as degree of $zf - fz$ is $d + 1$ and as a nonzero polynomial it cannot vanish on $\mathbb{M}_k(\mathbb{F})$ as $k \geq (d + 1)/2 + 1$. If $\text{Img}_k(f)$ is trace zero over $\mathbb{M}_k(\mathbb{F})$, then the span of the image of f can not be $\mathbb{M}_k(\mathbb{F})$ which can be checked efficiently by Theorem 5. Otherwise, if $\text{Img}_k(f)$ is not trace zero over $\mathbb{M}_k(\mathbb{F})$, its span must be the entire algebra $\mathbb{M}_k(\mathbb{F})$ as promised by the Brešar-Klep theorem. ◀

References

1. A. S. Amitsur and J. Levitzki. Minimal identities for algebras. *Proceedings of the American Mathematical Society*, 1(4):449–463, 1950. URL: http://www.jstor.org/stable/2032312.
2. S.A Amitsur. Rational identities and applications to algebra and geometry. *Journal of Algebra*, 3(3):304–359, 1966. doi:10.1016/0021-8693(66)90004-4.
3. Vikraman Arvind, Abhranil Chatterjee, Rajit Datta, and Partha Mukhopadhyay. Efficient black-box identity testing for free group algebras. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2019, September 20-22, 2019, Massachusetts Institute of Technology, Cambridge, MA, USA*, pages 57:1–57:16, 2019. doi:10.4230/LIPIcs.APPROX-RANDOM.2019.57.
4. Vikraman Arvind, Partha Mukhopadhyay, and Srikanth Srinivasan. New results on noncommutative and commutative polynomial identity testing. *Computational Complexity*, 19(4):521–558, 2010. doi:10.1007/s00037-010-0299-8.

5 J. Berstel and C. Reutenauer. *Noncommutative Rational Series with Applications*. Encyclopedia of Mathematics and its Applications. Cambridge University Press, 2011. URL: https://books.google.co.in/books?id=LL8Nhn72I_8C.

6 Andrej Bogdanov and Hoeteck Wee. More on noncommutative polynomial identity testing. In *20th Annual IEEE Conference on Computational Complexity (CCC 2005), 11-15 June 2005, San Jose, CA, USA*, pages 92–99, 2005. doi:10.1109/CCC.2005.13.

7 Matej Brešar and Igor Klep. Values of noncommutative polynomials, lie skew-ideals and the tracial nullstellensatz. *Mathematical Research Letters*, 2008.

8 Harm Derksen and Visu Makam. Polynomial degree bounds for matrix semi-invariants. *Advances in Mathematics*, 310:44–63, 2017. doi:10.1016/j.aim.2017.01.018.

9 Michael A. Forbes and Amir Shpilka. Quasipolynomial-time identity testing of non-commutative and read-once oblivious algebraic branching programs. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, 26-29 October, 2013, Berkeley, CA, USA*, pages 243–252, 2013. doi:10.1109/FOCS.2013.34.

10 Ankit Garg, Leonid Gurvits, Rafael Mendes de Oliveira, and Avi Wigderson. A deterministic polynomial time algorithm for non-commutative rational identity testing. *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 109–117, 2016.

11 Pavel Hrubeš and Avi Wigderson. Non-commutative arithmetic circuits with division. *Theory of Computing*, 11(14):357–393, 2015. doi:10.4086/toc.2015.v011a014.

12 Gábor Ivanyos, Youming Qiao, and K. V. Subrahmanyam. Constructive non-commutative rank computation is in deterministic polynomial time. *computational complexity*, 27(4):561–593, December 2018. doi:10.1007/s00037-018-0165-7.

13 Sylvain Lombardy and Jacques Sakarovitch. The removal of weighted ϵ-transitions. In Nelma Moreira and Rogério Reis, editors, *Implementation and Application of Automata*, pages 345–352, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

14 Ran Raz and Amir Shpilka. Deterministic polynomial identity testing in non-commutative models. *Computational Complexity*, 14(1):1–19, 2005. doi:10.1007/s00037-005-0188-8.

15 M.P. Schützenberger. On the definition of a family of automata. *Information and Control*, 4(2):245–270, 1961. doi:10.1016/S0019-9958(61)80020-X.

16 Dirk Werner. *Funktionalanalysis (in German)*. Springer Verlag, 2005.