Electronic Colloquium on Computational Complexity, Report No. 36 (2009)

The Power of Depth 2 Circuits over Algebras

Chandan Saha* Ramprasad Saptharishi† Nitin Saxena‡

Abstract

We study the problem of polynomial identity testing (PIT) for depth 2 arithmetic circuits over matrix algebra. We show that identity testing of depth 3 ($\Sigma\Pi\Sigma$) arithmetic circuits over a field $\mathbb{F}$ is polynomial time equivalent to identity testing of depth 2 ($\Pi\Sigma$) arithmetic circuits over $\mathsf{U}_2(\mathbb{F})$, the algebra of upper-triangular 2×2 matrices with entries from $\mathbb{F}$. Such a connection is a bit surprising since we also show that, as computational models, $\Pi\Sigma$ circuits over $\mathsf{U}_2(\mathbb{F})$ are strictly 'weaker' than $\Sigma\Pi\Sigma$ circuits over $\mathbb{F}$.

The equivalence further shows that PIT of depth 3 arithmetic circuits reduces to PIT of width-2 planar commutative *Algebraic Branching Programs*(ABP). Thus, identity testing for commutative ABPs is interesting even in the case of width-2.

Further, we give a deterministic polynomial time identity testing algorithm for a $\Pi\Sigma$ circuit over any constant dimensional commutative algebra over $\mathbb{F}$. While over commutative algebras of polynomial dimension, identity testing is at least as hard as that of $\Sigma\Pi\Sigma$ circuits over $\mathbb{F}$.

1 Introduction

Polynomial identity testing (PIT) is a fundamental problem in theoretical computer science. Over the last decade this problem has drawn significant attention from many leading researchers owing to its role in designing efficient algorithms and in proving circuit lower bounds. Identity testing is the following problem:

Problem 1.1. *Given an arithmetic circuit C with input variables $x_1, \ldots, x_n$ and constants taken from a field $\mathbb{F}$, check if the polynomial computed by C is identically zero.*

Besides being a natural problem in algebraic computation, identity testing appears in important complexity theory results such as, $\mathsf{IP} = \mathsf{PSPACE}$ [Sha90] and the PCP theorem [ALM+98]. It also plays a promising role in proving super-polynomial circuit lower bound for permanent [KI03, Agr05]. Moreover, algorithms for problems like primality testing [AKS04], graph matching [Lov79] and multivariate polynomial interpolation [CDGK91] also involve identity testing.

The first randomized polynomial time algorithm for identity testing was given by Schwartz and Zippel [Sch80, Zip79]. Several other efficient randomized algorithms [CK97, LV98, AB99, KS01] came up subsequently, resulting in a significant improvement in the number of random bits used. However, despite many attempts a deterministic polynomial time algorithm has remained elusive. Nevertheless, important progress has been made both in the designing of deterministic algorithms for special circuits, and in the understanding of why a general deterministic solution could be hard

*Indian Institute of Technology, Kanpur 208016, India. Email - csaha@cse.iitk.ac.in

†Chennai Mathematical Institute, Chennai 603103, India. Supported by MSR India PhD Fellowship. Email - ramprasad@cmi.ac.in

‡Hausdorff Center for Mathematics, Bonn 53115, Germany. Email - ns@hcm.uni-bonn.de

ISSN 1433-8092

to get.

Without loss of generality, we can assume that a circuit C has alternate layers of addition and multiplication gates. A layer of addition gates is denoted by Σ and that of multiplication gates is denoted by Π. Kayal and Saxena [KS07] gave a deterministic polynomial time identity testing algorithm for depth 3 ($\Sigma\Pi\Sigma$) circuits with constant top fan-in. As such, no other general polynomial time result is known for depth 3 circuits. A justification behind the hardness of PIT even for small depth circuits was provided recently by Agrawal and Vinay [AV08]. They showed that a deterministic black box identity test for depth 4 ($\Sigma\Pi\Sigma\Pi$) circuits would imply a quasi-polynomial time deterministic PIT algorithm for any circuit computing a polynomial of *low* degree[1].

Thus we see that the non-trivial case for identity testing starts with depth 3 circuits; whereas circuits of depth 4 are *almost* the general case. It is therefore natural to ask as to what is the complexity of the PIT problem for depth 2 ($\Pi\Sigma$) circuits if we allow the *constants* of the circuit to come from an *algebra* [2] $\mathcal{R}$, which is not a field, and has dimension over $\mathbb{F}$, $\dim_{\mathbb{F}}(\mathcal{R}) > 1$. We can make the reasonable assumption that the algebra $\mathcal{R}$ is given in *basis form* i.e. we know an $\mathbb{F}$-basis $\{e_1, \ldots, e_k\}$ of $\mathcal{R}$ and we also know how $e_i e_j$ can be expressed in terms of the basis elements, for all i and j. Therefore, the problem at hand is the following,

Problem 1.2. *Given an expression,*

$$P = \prod_{i=1}^{d} (A_{i0} + A_{i1}x_1 + \ldots + A_{in}x_n)$$

where $A_{ij} \in \mathcal{R}$, an algebra over $\mathbb{F}$ given in basis form, check if P is zero.

How hard is the above problem? At first sight, this problem might look deceptively simple. For instance, if $\mathcal{R}$ is field or even a division algebra (say, the real quaternion algebra) then it is trivial to check if $P = 0$ using polynomial number of $\mathbb{F}$-operations. However, in general this is far from what might be the case.

Since elements of a finite dimensional algebra, given in basis form, can be expressed as matrices over $\mathbb{F}$ we can equivalently write the above problem as,

Problem 1.3. *Given an expression,*

$$P = \prod_{i=1}^{d} (A_{i0} + A_{i1}x_1 + \ldots + A_{in}x_n) \tag{1}$$

where $A_{ij} \in \mathsf{M}_k(\mathbb{F})$, the algebra of $k \times k$ matrices over $\mathbb{F}$, check if P is zero using $\mathsf{poly}(knd)$ number of $\mathbb{F}$-operations.

In order to avoid confusion we would use the following convention in this paper:

Convention - Whenever we say 'arithmetic circuit' or 'arithmetic formula' without any extra qualification, we would mean a circuit or a formula over a field. Otherwise, we would explicitly mention 'arithmetic circuit (or formula) over *some* algebra' to mean that the constants of the circuit are taken from 'that' algebra.

[1] A polynomial is said to have low degree if its degree is less than the size of the circuit

[2] In this paper we always mean a finite dimensional associative algebra with unity.

1.1 The depth 2 model of computation

A depth 2 circuit C over matrices, as in Equation 1, naturally defines a computational model. Assuming $\mathcal{R} = \mathsf{M}_k(\mathbb{F})$, for some k, a polynomial $P \in \mathcal{R}[x_1, \ldots, x_n]$ outputted by C can be viewed as a $k \times k$ matrix of polynomials in $\mathbb{F}[x_1, \ldots, x_n]$. We say that a polynomial $f \in \mathbb{F}[x_1, \ldots, x_n]$ is *computed* by C if one of the k^2 polynomials in P is f. Sometimes we would abuse terminology a bit and say that P *computes* f to mean the same.

In the following discussion, we would denote the algebra of upper-triangular $k \times k$ matrices by $\mathsf{U}_k(\mathbb{F})$. The algebra $\mathsf{U}_2(\mathbb{F})$ is the *smallest* noncommutative algebra with unity over $\mathbb{F}$, in the sense that $\dim_{\mathbb{F}} \mathsf{U}_2(\mathbb{F}) = 3$ whereas any algebra with unity of dimension less than 3 is commutative. We show in this paper that already $\mathsf{U}_2(\mathbb{F})$ captures an open case of identity testing.

Ben-Or and Cleve [BC88] showed that a polynomial computed by an arithmetic formula E of depth d, and fan-in (of every gate) bounded by 2, can also be computed by a straight-line program of length at most 4^d using only 3 registers. The following fact can be readily derived from their result (see Theorem A.1): From an arithmetic formula E of depth d and fan-in bounded by 2, we can efficiently compute the expression,

$$P = \prod_{i=1}^{m} (A_{i0} + A_{i1}x_1 + \ldots + A_{in}x_n)$$

where $m \leq 4^d$ and $A_{ij} \in \mathsf{M}_3(\mathbb{F})$ such that P *computes* the polynomial that E does. Thus solving Problem 1.3 in polynomial time even for 3×3 matrices yields a polynomial time algorithm for PIT of constant depth circuits, in particular depth 4 circuits. There is another way of arguing that the choice of $\mathcal{R}$ as $\mathsf{M}_3(\mathbb{F})$ is *almost* the general case.

For an arithmetic circuit of size s, computing a low degree polynomial, we can use the depth-reduction result by Allender, Jiao, Mahajan and Vinay [AJMV98] (see also [VSBR83]) to get an equivalent bounded fan-in formula of size $s^{O(\log s)}$ and depth $O(\log^2 s)$. From this formula we can obtain a depth 2 circuit over $\mathsf{M}_3(\mathbb{F})$ of size $4^{O(\log^2 s)} = s^{O(\log s)}$ (using Ben-Or and Cleve's result) that computes the same polynomial as the formula. Thus, PIT for depth 2 circuits over 3×3 matrices is *almost* the general case since a derandomization yields a quasi-polynomial time PIT algorithm for any circuit computing a low degree polynomial. This means, in essence a depth 2 circuit over $\mathsf{M}_3(\mathbb{F})$ plays the role of a depth 4 circuit over $\mathbb{F}$ (using Agrawal and Vinay's result).

What is natural to ask is how the complexity of PIT for depth 2 circuits over $\mathsf{M}_2(\mathbb{F})$ relates to PIT for arithmetic circuits. In this paper, we provide an answer to this. We show a surprising connection between PIT of depth 2 circuits over $\mathsf{U}_2(\mathbb{F})$ and PIT of depth 3 circuits. The reason this is a bit surprising is because we also show that, a depth 2 circuit over $\mathsf{U}_2(\mathbb{F})$ is not even powerful enough to compute a simple polynomial like, $x_1x_2 + x_3x_4 + x_5x_6$!

Known related models

Polynomial identity testing and circuit lower bounds have been studied for different algebraic models. Nisan [Nis91] showed an exponential lower bound on the size of any arithmetic formula computing the determinant of a matrix in the non-commutative *free algebra* model. The result was later generalized by Chien and Sinclair [CS04] to a large class of non-commutative algebras satisfying polynomial identities, called PI-algebras. Identity testing has also been studied for the non-commutative model by Raz and Shpilka [RS04], Bogdanov and Wee [BW05], and Arvind, Mukhopadhyay and Srinivasan [AMS08]. But unlike this model where the variables do not commute, in our setting the variables always commute but the constants are taken from an algebra $\mathcal{R}$.

The motivation for studying this later model is not only because it is a natural generalization of commutative circuits over fields but also because it gives a different perspective to the complexity of the classical PIT problem in terms of the dimension of the underlying algebra $\mathcal{R}$. It seems to 'pack' the combinatorial nature of the circuit into a larger base algebra and hence opens up the possibility of using algebra structure results. The simplest nontrivial circuit in this model is a depth 2 circuit over the smallest non-commutative algebra which is $\mathcal{R} = \mathsf{U}_2(\mathbb{F})$.

1.2 Our Results

The results we give are of two types. Some are related to identity testing while the rest are related to the weakness of the depth 2 computational model over $\mathsf{U}_2(\mathbb{F})$ and $\mathsf{M}_2(\mathbb{F})$.

Identity testing

We fill in the missing information about the complexity of identity testing for depth 2 circuits over 2×2 matrices by showing the following result.

Theorem 1.4. *Identity testing for depth 2 ($\Pi\Sigma$) circuits over $\mathsf{U}_2(\mathbb{F})$ is polynomial time equivalent to identity testing for depth 3 ($\Sigma\Pi\Sigma$) circuits.*

The above result has an interesting consequence on identity testing for Algebraic Branching Program (ABP) [Nis91] (see Definition 2.2). It is known that identity testing for non-commutative ABP can be done in deterministic polynomial time (a result due to Raz and Shpilka [RS04]). But no interesting result is known for identity testing of even width-2 commutative ABP's. The following result explains why this is the case.

Corollary 1.5. *Identity testing of depth 3 circuits reduces to identity testing of width-2 planar ABPs.*

Further, we give a deterministic polynomial time identity testing algorithm for depth 2 circuits over any constant dimensional commutative algebra given in basis form. Recall that an algebra $\mathcal{R}$ is given in basis form if we know an $\mathbb{F}$-basis $\{e_1, \ldots, e_k\}$ of $\mathcal{R}$ and we also know how $e_i e_j$ can be expressed in terms of the basis elements, for all i and j. Our result can be formally stated as follows.

Theorem 1.6. *Given an expression,*

$$P = \prod_{i=1}^{d} (A_{i0} + A_{i1}x_1 + \ldots + A_{in}x_n)$$

where $A_{ij} \in \mathcal{R}$, a commutative algebra of constant dimension over $\mathbb{F}$ that is given in basis form, there is a deterministic polynomial time algorithm to test if P is zero.

In a way, this result establishes the fact that the power of depth 2 ($\Pi\Sigma$) circuits over constant dimensional algebras is primarily derived from the non-commutative nature of the algebra. However, things can be very different for commutative algebras of polynomial dimension over $\mathbb{F}$.

Theorem 1.7. *Identity testing of a depth 3 ($\Sigma\Pi\Sigma$) circuit C reduces to identity testing of a depth 2 ($\Pi\Sigma$) circuit over a commutative algebra of dimension polynomial in the size of C.*

It would be apparent from the proof of Theorem 1.4 that our argument is simple in nature. Perhaps the reason why such a connection was overlooked before is that, unlike a depth 2 circuit over $\mathsf{M}_3(\mathbb{F})$, we do not always have the privilege of *exactly* computing a polynomial over $\mathbb{F}$ using a depth 2 circuit over $\mathsf{U}_2(\mathbb{F})$. Showing this weakness of the latter computational model constitutes the other part of our results.

Weakness of the depth 2 model over $\mathsf{U}_2(\mathbb{F})$ and $\mathsf{M}_2(\mathbb{F})$

Although Theorem 1.4 shows an equivalence of depth 3 circuits and depth 2 circuits over $\mathsf{U}_2(\mathbb{F})$ with respect to PIT, the computational powers of these two models are very different. The following result shows that a depth 2 circuit over $\mathsf{U}_2(\mathbb{F})$ is computationally strictly weaker than depth 3 circuits.

Theorem 1.8. *Let $f \in F[x_1, \ldots, x_n]$ be a polynomial such that there are no two linear functions l_1 and l_2 (with $1 \notin (l_1, l_2)$, the ideal generated by l_1 and l_2) which make $f \bmod (l_1, l_2)$ also a linear function. Then f is not computable by a depth 2 circuit over $\mathsf{U}_2(\mathbb{F})$.*

It can be shown that even a simple polynomial like $x_1x_2 + x_3x_4 + x_5x_6$ satisfies the condition stated in the above theorem (see Corollary 4.1), and hence it is not computable by any depth 2 circuit over $\mathsf{U}_2(\mathbb{F})$, no matter how large! This contrast makes Theorem 1.4 surprising as it establishes an equivalence of identity testing in two models of different computational strengths.

At this point, it is natural to investigate the computational power of depth 2 circuits if we graduate from $\mathsf{U}_2(\mathbb{F})$ to $\mathsf{M}_2(\mathbb{F})$. The following result hints that even such a model is severely restrictive in nature.

A depth 2 circuit over $\mathsf{M}_2(\mathbb{F})$ gives as output a polynomial $P = \prod_{i=1}^{d} (A_{i0} + A_{i1}x_1 + \ldots + A_{in}x_n)$, with $A_{ij} \in \mathsf{M}_2(\mathbb{F})$. Let P_ℓ denote the partial product $P_\ell = \prod_{i=\ell}^{d} (A_{i0} + A_{i1}x_1 + \ldots + A_{in}x_n)$ where $\ell \leq d$.

Definition 1.9. *A polynomial $f \in \mathbb{F}[x_1, \ldots, x_n]$ is computed by a depth 2 circuit over $\mathsf{M}_2(\mathbb{F})$* under a degree restriction *of m if the degree of each of the partial products P_ℓ is bounded by m.*

Theorem 1.10. *There exists a class of polynomials of degree n that cannot be computed by a depth 2 circuit over $\mathsf{M}_2(\mathbb{F})$, under a degree restriction of n.*

The motivation for imposing a condition like degree restriction comes very naturally from depth 2 circuits over $\mathsf{M}_3(\mathbb{F})$. Given a polynomial $f = \sum_i m_i$, where m_i's are the monomials of f, it is easy to construct a depth 2 circuit over $\mathsf{M}_3(\mathbb{F})$ that literally forms these monomials and adds then one by one. This computation is degree restricted, if we extend our definition of degree restriction to $\mathsf{M}_3(\mathbb{F})$. However, the above theorem suggests that no such scheme to compute f would succeed over $\mathsf{M}_2(\mathbb{F})$.

Remark- By transferring the complexity of an arithmetic circuit from its depth to the dimension of the underlying algebras while fixing the depth to 2, our results provide some evidence that identity testing for depth 3 circuits appears to be *mathematically* more tractable than depth 4 circuits. Besides, it might be possible to exploit the properties of these underlying algebras to say something useful about identity testing. A glimpse of this indeed appears in our identity testing algorithm over commutative algebras of constant dimension over F.

1.3 Organization

The results on identity testing are given in sections 2 and 3, while those on the weakness of the depth 2 model are given in section 4. In section 2 we prove the equivalence of identity testing between depth 3 circuits and depth 2 circuits over $\mathsf{U}_2(\mathbb{F})$ (Theorem 1.4), and show how it connects to width-2 ABPs (Corollary 1.5). The deterministic polynomial time identity testing algorithm over commutative algebra of constant dimension is presented in section 3 (Theorem 1.6). In the same section it is also shown that commutative algebras of polynomial dimensions are powerful enough to capture PIT of depth 3 ($\Sigma\Pi\Sigma$) circuits (Theorem 1.7). Finally, in section 4 we show the weakness of the depth 2 model over $\mathsf{U}_2(\mathbb{F})$ and $\mathsf{M}_2(\mathbb{F})$ (Theorem 1.8 and 1.10).

2 Identity testing over $\mathsf{M}_2(\mathbb{F})$

In this section, we show that PIT of depth 2 circuits over $\mathsf{M}_2(\mathbb{F})$ is at least as hard as PIT of depth 3 circuits, and this further implies that PIT of a width-2 commutative ABP is also at least as hard as PIT of depth 3 circuits.

2.1 Equivalence with depth 3 identity testing

We will now prove Theorem 1.4. Given a depth 3 circuit we can assume, without loss of generality, that the fan-in of the multiplication gates are the same. This multiplicative fan-in will be referred to as the *degree* of the depth 3 circuit. The following lemma is the crux of our argument. For convenience, we will call a matrix with linear functions as entries, a *linear* matrix.

Lemma 2.1. *Let $f \in \mathbb{F}[x_1, \ldots, x_n]$ be a polynomial computed by a depth 3 circuit C of degree d and top level fan-in s. Given circuit C, it is possible to construct in polynomial time a depth 2 circuit over $\mathsf{U}_2(\mathbb{F})$ of size $O((d+n)s^2)$ that computes a polynomial $p = L \cdot f$, where L is a product of non-zero linear functions.*

Proof. A depth 2 circuit over $\mathsf{U}_2(\mathbb{F})$ is simply a product sequence of 2×2 upper-triangular linear matrices. We now show that there exists such a sequence of length $O((d+n)s^2)$ such that the product 2×2 matrix has $L \cdot f$ as one of its entries.

Since f is computed by a depth 3 circuit, we can write $f = \sum_{i=1}^{s} P_i$, where each summand $P_i = \prod_j l_{ij}$ is a product of linear functions. Observe that we can compute a single P_i using a product sequence of length d as:

$$\begin{bmatrix} l_{i1} & \\ & 1 \end{bmatrix} \begin{bmatrix} l_{i2} & \\ & 1 \end{bmatrix} \cdots \begin{bmatrix} l_{i(d-1)} & \\ & 1 \end{bmatrix} \begin{bmatrix} 1 & l_{id} \\ & 1 \end{bmatrix} = \begin{bmatrix} L' & P_i \\ & 1 \end{bmatrix} \tag{2}$$

where $L' = l_{i1} \cdots l_{i(d-1)}$.

Each matrix of the form $\begin{bmatrix} 1 & l \\ & 1 \end{bmatrix}$, where $l = a_0 + \sum a_i x_i$, can be further expanded as,

$$\begin{bmatrix} 1 & a_0 \\ & 1 \end{bmatrix} \begin{bmatrix} 1 & a_1 x_1 \\ & 1 \end{bmatrix} \cdots \begin{bmatrix} 1 & a_n x_n \\ & 1 \end{bmatrix} = \begin{bmatrix} 1 & l \\ & 1 \end{bmatrix}$$

These will be the only type of non-diagonal matrices that would appear in the sequence.

The proof will proceed by induction where Equation 2 serves as the induction basis. A generic intermediate matrix would look like $\begin{bmatrix} L_1 & L_2 g \\ & L_3 \end{bmatrix}$ where each L_i is a product of non-zero linear functions and g is a partial summand of P_i's. We shall inductively double the number of summands in g at each step.

At the i-th iteration let us assume that we have the matrices $\begin{bmatrix} L_1 & L_2 g \\ & L_3 \end{bmatrix}$ and $\begin{bmatrix} M_1 & M_2 h \\ & M_3 \end{bmatrix}$, each computed by a sequence of n_i linear matrices. We now want a sequence that computes a polynomial of the form $L \cdot (g + h)$. Consider the following sequence,

$$\begin{bmatrix} L_1 & L_2 g \\ & L_3 \end{bmatrix} \begin{bmatrix} A & \\ & B \end{bmatrix} \begin{bmatrix} M_1 & M_2 h \\ & M_3 \end{bmatrix} = \begin{bmatrix} AL_1M_1 & AL_1M_2h + BL_2M_3g \\ & BL_3M_3 \end{bmatrix} \tag{3}$$

where A, B are products of linear functions. By setting $A = L_2M_3$ and $B = L_1M_2$ we get the desired sequence,

$$\begin{bmatrix} L_1 & L_2 g \\ & L_3 \end{bmatrix} \begin{bmatrix} A & \\ & B \end{bmatrix} \begin{bmatrix} M_1 & M_2 h \\ & M_3 \end{bmatrix} = \begin{bmatrix} L_1L_2M_1M_3 & L_1L_2M_2M_3(g+h) \\ & L_1L_3M_2M_3 \end{bmatrix}$$

This way, we have doubled the number of summands in $g+h$. The length of the sequence computing L_2g and M_2h is n_i, hence each L_i and M_i is a product of n_i many linear functions. Therefore, both A and B are products of at most $2n_i$ linear functions and the matrix $\begin{bmatrix} A & \\ & B \end{bmatrix}$ can be written as a product of at most $2n_i$ diagonal linear matrices. The total length of the sequence given in Equation 3 is hence bounded by $4n_i$.

The number of summands in f is s and the above process needs to be repeated at most $\log s + 1$ times. The final sequence length is hence bounded by $(d+n) \cdot 4^{\log s} = (d+n)s^2$. □

Proof of Theorem 1.4. It follows from Lemma 2.1 that, given a depth 3 circuit C computing f we can efficiently construct a depth 2 circuit over $\mathsf{U}_2(\mathbb{F})$ that outputs a matrix, $\begin{bmatrix} L_1 & L \cdot f \\ & L_2 \end{bmatrix}$, where L is a product of non-zero linear functions. Multiplying this matrix by $\begin{bmatrix} 1 & 0 \\ & 0 \end{bmatrix}$ to the left and $\begin{bmatrix} 0 & 0 \\ & 1 \end{bmatrix}$ to the right yields another depth 2 circuit D that outputs $\begin{bmatrix} 0 & L \cdot f \\ & 0 \end{bmatrix}$. Thus D computes an identically zero polynomial over $\mathsf{U}_2(\mathbb{F})$ if and only if C computes an identically zero polynomial. This shows that PIT for depth 3 circuits reduces to PIT of depth 2 circuits over $\mathsf{U}_2(\mathbb{F})$.

The other direction, that is PIT for depth 2 circuits over $\mathsf{U}_2(\mathbb{F})$ reduces to PIT for depth 3 circuits, is trivial to observe. The diagonal entries of the output 2×2 matrix is just a product of linear functions whereas the off-diagonal entry is a sum of at most d' many products of linear functions, where d' is the multiplicative fan-in of the depth 2 circuit over $\mathsf{U}_2(\mathbb{F})$. □

2.2 Width-2 algebraic branching programs

Algebraic Branching Program (ABP) is a model of computation defined by Nisan [Nis91]. Formally, an ABP is defined as follows.

Definition 2.2. *(Nisan [Nis91]) An* algebraic branching program (ABP) *is a directed acyclic graph with one source and one sink. The vertices of this graph are partitioned into levels labelled* 0 *to* d,

where edges may go from level i to level $i+1$. The parameter d is called the degree of the ABP. The source is the only vertex at level 0 and the sink is the only vertex at level d. Each edge is labelled with a homogeneous linear function of $x_1, \ldots, x_n$ (i.e. a function of the form $\sum_i c_i x_i$). The width *of the ABP is the maximum number of vertices in any level, and the* size *is the total number of vertices.*

An ABP computes a function in the obvious way; sum over all paths from source to sink, the product of all linear functions by which the edges of the path are labelled.

An ABP is said to be *planar* if the underlying graph is planar.

The following argument shows how Corollary 1.5 follows easily from Theorem 1.4.

Proof of Corollary 1.5. Theorem 1.4 constructs a depth 2 circuit D that computes $P = \prod_j (A_{j0} + A_{j1}x_1 + \ldots + A_{jn}x_n)$, where each $A_{ji} \in \mathsf{U}_2(\mathbb{F})$. We can make D homogeneous by introducing an extra variable z, such that $P = \prod_j (A_{j0}z + A_{j1}x_1 + \ldots + A_{jn}x_n)$. This means, the product sequence considered in Lemma 2.1, is such that all the linear matrices have homogeneous linear functions as entries and the only non-diagonal linear matrices are of the form $\begin{bmatrix} z & cx_i \\ & z \end{bmatrix}$. It is now straightforward to construct a width-2 ABP by making the j^{th} linear matrix in the sequence act as the adjacency matrix between level j and $j+1$ of the ABP. The ABP constructed is planar since it has layers only of the following two kinds:

where l_1. l_2 are homogeneous linear functions. □

As a matter of fact, the above argument actually shows that PIT of depth 2 circuits over $\mathsf{M}_2(\mathbb{F})$ reduces to PIT of width-2 ABPs.

3 Identity testing over commutative algebras

We would now prove Theorem 1.6. The main idea behind this proof is a structure theorem for finite dimensional commutative algebras over a field. To state the theorem we need the following definition.

Definition 3.1. *A ring $\mathcal{R}$ is* local *if it has a unique maximal ideal.*

An element u in a ring $\mathcal{R}$ is said to be a *unit* if there exist an element u' such that $uu' = 1$, where 1 is the identity element of $\mathcal{R}$. An element $m \in \mathcal{R}$ is *nilpotent* if there exist a positive integer n with $m^n = 0$. In a local ring the unique maximal ideal consists of all non-units in $\mathcal{R}$.

The following theorem shows how a commutative algebra decomposes into local sub-algebras. The theorem is quite well known in the theory of commutative algebras. But since we need an effective version of this theorem, we present the proof here for the sake of completion and clarity.

Theorem 3.2. *A finite dimensional commutative algebra $\mathcal{R}$ over $\mathbb{F}$ is isomorphic to a direct product of local rings i.e.*

$$\mathcal{R} \cong \mathcal{R}_1 \oplus \ldots \oplus \mathcal{R}_\ell$$

where each $\mathcal{R}_i$ is a local ring contained in $\mathcal{R}$ and any non-unit in $\mathcal{R}_i$ is nilpotent.

Proof. If all non-units in $\mathcal{R}$ are nilpotents then $\mathcal{R}$ is a local ring and the set of nilpotents forms the unique maximal ideal. Therefore, suppose that there is a non-nilpotent zero-divisor z in $\mathcal{R}$. (Any non-unit z in a finite dimensional algebra is a zero-divisor i.e. $\exists y \in \mathcal{R}$ and $y \neq 0$ such that $yz = 0$.) We would argue that using z we can find an *idempotent* $v \notin \{0, 1\}$ in $\mathcal{R}$ i.e. $v^2 = v$.

Assume that we do have a non-trivial idempotent $v \in \mathcal{R}$. Let $\mathcal{R}v$ be the sub-algebra of $\mathcal{R}$ generated by multiplying elements of $\mathcal{R}$ with v. Since any $a = av + a(1-v)$ and $\mathcal{R}v \cap \mathcal{R}(1-v) = \{0\}$, we get $\mathcal{R} \cong \mathcal{R}v \oplus \mathcal{R}(1-v)$ as a non-trivial decomposition of $\mathcal{R}$. (Note that $\mathcal{R}$ is a direct sum of the two sub-algebras because for any $a \in \mathcal{R}v$ and $b \in \mathcal{R}(1-v)$, $a \cdot b = 0$. This is the place where we use commutativity of $\mathcal{R}$.) By repeating the splitting process on the sub-algebras we can eventually prove the theorem. We now show how to find an idempotent from the zero-divisor z.

An element $a \in \mathcal{R}$ can be expressed equivalently as a matrix in $\mathsf{M}_k(\mathbb{F})$, where $k = \dim_{\mathbb{F}}(\mathcal{R})$, by treating a as the linear transformation on $\mathcal{R}$ that takes $b \in \mathcal{R}$ to $a \cdot b$. Therefore, z is a zero-divisor if and only if z as a matrix is singular. Consider the Jordan normal form of z. Since it is merely a change of basis we would assume, without loss of generality, that z is already in Jordan normal form. (We won't compute the Jordan normal form in our algorithm, it is used only for the sake of argument.) Let,

$$z = \begin{bmatrix} A & 0 \\ 0 & N \end{bmatrix}$$

where A, N are block diagonal matrices and A is non-singular and N is nilpotent. Therefore there exits a positive integer $t < k$ such that,

$$w = z^t = \begin{bmatrix} B & 0 \\ 0 & 0 \end{bmatrix}$$

where $B = A^t$ is non-singular. The claim is, there is an identity element in the sub-algebra $\mathcal{R}w$ which can be taken to be the idempotent that splits $\mathcal{R}$. To see this first observe that the minimum polynomial of w over $\mathbb{F}$ is $m(x) = x \cdot m'(x)$, where $m'(x)$ is the minimum polynomial of B. Also if $m(x) = \sum_{i=1}^{k} \alpha_i x^i$ then $\alpha_1 \neq 0$ as it is the constant term of $m'(x)$ and B is non-singular. Therefore, there exists an $a \in \mathcal{R}$ such that $w \cdot (aw - 1) = 0$. We can take $v = aw$ as the identity element in the sub-algebra $\mathcal{R}w$. This $v \notin \{0, 1\}$ is the required idempotent in $\mathcal{R}$. □

We are now ready to prove Theorem 1.6.

Theorem 1.6 (restated.) *Given an expression,*

$$P = \prod_{i=1}^{d} (A_{i0} + A_{i1}x_1 + \ldots + A_{in}x_n)$$

where $A_{ij} \in \mathcal{R}$, a commutative algebra of constant dimension over $\mathbb{F}$ that is given in basis form, there is a deterministic polynomial time algorithm to test if P is zero.

Proof. Suppose, the elements $e_1, \ldots, e_k$ form a basis of $\mathcal{R}$ over $\mathbb{F}$. Since any element in $\mathcal{R}$ can be equivalently expressed as a $k \times k$ matrix over $\mathbb{F}$ (by treating it as a linear transformation), we will assume that $A_{ij} \in \mathsf{M}_k(\mathbb{F})$, for all i and j. Further, since $\mathcal{R}$ is given in basis form, we can find these matrix representations of A_{ij}'s efficiently.

If every A_{ij} is non-singular, then surely $P \neq 0$. (This can be argued by fixing an ordering $x_1 \succ x_2 \succ \ldots \succ x_n$ among the variables. The coefficient of the leading monomial of P, with respect to this ordering, is a product of invertible matrices and hence $P \neq 0$.) Therefore, assume that $\exists A_{ij} = z$ such that z is a zero-divisor i.e. singular. From the proof of Theorem 3.2 it follows that there exists a $t < k$ such that the sub-algebra $\mathcal{R}w$, where $w = z^t$, contains an identity element v which is an idempotent. To find the right w we can simply go through all $1 \leq t < k$. We now argue that for the correct choice of w, v can be found by solving a system of linear equations over $\mathbb{F}$. Let $b_1, \ldots, b_{k'}$ be a basis of $\mathcal{R}w$, which we can find easily from the elements $e_1w, \ldots, e_kw$. In order to solve for v write it as,

$$v = \nu_1 b_1 + \ldots + \nu_{k'} b_{k'}$$

where $\nu_j \in \mathbb{F}$ are unknowns. Since v is an identity in $\mathcal{R}w$ we have the following equations,

$$(\nu_1 b_1 + \ldots + \nu_{k'} b_{k'}) \cdot b_i = b_i \quad \text{for } 1 \leq i \leq k'.$$

Expressing each b_i in terms of $e_1, \ldots, e_k$, we get a set of linear equations in ν_j's. Thus for the right choice of w (i.e. for the right choice of t) there is a solution for v. On the other hand, a solution for v for any w gives us an idempotent, which is all that we need.

Since $\mathcal{R} \cong \mathcal{R}v \oplus \mathcal{R}(1-v)$ we can now split the identity testing problem into two similar problems, i.e. P is zero if and only if,

$$\begin{aligned} Pv &= \prod_{i=1}^{d} (A_{i0}v + A_{i1}v \cdot x_1 + \ldots + A_{in}v \cdot x_n) \quad \text{and} \\ P(1-v) &= \prod_{i=1}^{d} (A_{i0}(1-v) + A_{i1}(1-v) \cdot x_1 + \ldots + A_{in}(1-v) \cdot x_n) \end{aligned}$$

are both zero. What we just did with $P \in \mathcal{R}$ we can repeat for $Pv \in \mathcal{R}v$ and $P(1-v) \in \mathcal{R}(1-v)$. By decomposing the algebra each time an A_{ij} is a non-nilpotent zero-divisor, we have reduced the problem to the following easier problem of checking if

$$P = \prod_{i=1}^{d} (A_{i0} + A_{i1}x_1 + \ldots + A_{in}x_n)$$

is zero, where the coefficients A_{ij}'s are either nilpotent or invertible matrices.

Let $T_i = (A_{i0} + A_{i1}x_1 + \ldots + A_{in}x_n)$ be a term such that the coefficient of x_j in T_i, i.e. A_{ij} is invertible. And suppose Q be the product of all terms other than T_i. Then $P = T_i \cdot Q$ (since $\mathcal{R}$ is commutative). Fix an ordering among the variables so that x_j gets the highest priority. The leading coefficient of P, under this ordering, is A_{ij} times the leading coefficient of Q. Since A_{ij} is invertible this implies that $P = 0$ if and only if $Q = 0$. (If A_{i0} is invertible, we can arrive at the same conclusion by arguing with the coefficients of the least monomials of P and Q under some ordering.) In other words, $P = 0$ if and only if the product of all those terms for which all the coefficients are nilpotent matrices is zero. But this is easy to check since the dimension of the algebra, k is a constant. (In fact, this is the only step where we use that k is a constant.) If number of such terms is greater than k then P is automatically zero (this follows easily from the

fact that the commuting nilpotent matrices can be simultaneously triangularized with zeroes in the diagonal). Otherwise, simply multiply those terms and check if it is zero. This takes $O(n^k)$ operations over $\mathbb{F}$. □

It is clear from the above discussion that identity testing of depth 2 ($\Pi\Sigma$) circuits over commutative algebras reduces in polynomial time to that over local rings. As long as the dimensions of these local rings are constant we are through. But what happens for nonconstant dimensions? The following result justifies the hardness of this problem.

Theorem 1.7 (restated.) *Given a depth 3 ($\Sigma\Pi\Sigma$) circuit C of degree d and top level fan-in s, it is possible to construct in polynomial time a depth 2 ($\Pi\Sigma$) circuit $\tilde{C}$ over a local ring of dimension $s(d-1)+2$ over $\mathbb{F}$ such that $\tilde{C}$ computes a zero polynomial if and only if C does so.*

Proof. The proof is relatively straightforward. Consider a depth 3 ($\Sigma\Pi\Sigma$) circuit computing a polynomial $f = \sum_{i=1}^{s} \prod_{j=1}^{d} l_{ij}$, where l_{ij}'s are linear functions. Consider the ring $\mathcal{R} = \mathbb{F}[y_1, \ldots, y_s]/\mathcal{I}$, where $\mathcal{I}$ is an ideal generated by the elements $\{y_i y_j\}_{1 \le i < j \le s}$ and $\{y_1^d - y_i^d\}_{1 < i \le s}$. Observe that $\mathcal{R}$ is a local ring, as $y_i^{d+1} = 0$ for all $1 \le i \le s$. Also the elements $\{1, y_1, \ldots, y_1^d, y_2, \ldots, y_2^{d-1}, \ldots, y_s, \ldots, y_s^{d-1}\}$ form an $\mathbb{F}$-basis of $\mathcal{R}$. Now notice that the polynomial,

$$\begin{aligned} P &= \prod_{j=1}^{d} (l_{j1} y_1 + \ldots + l_{js} y_s) \\ &= f \cdot y_1^d \end{aligned}$$

is zero if and only if f is zero. Polynomial P can indeed be computed by a depth 2 ($\Pi\Sigma$) circuit over $\mathcal{R}$. □

4 Weakness of the depth 2 model

In Lemma 2.1, we saw that the depth 2 circuit over $\mathsf{U}_2(\mathbb{F})$ computes $L \cdot f$ instead of f. Is it possible to drop the factor L and simply compute f? In this section, we show that in *many* cases it is impossible to find a depth 2 circuit over $\mathsf{U}_2(\mathbb{F})$ that computes f.

4.1 Depth 2 model over $\mathsf{U}_2(\mathbb{F})$

We will now prove Theorem 1.8. In the following discussion we use the notation (l_1, l_2) to mean the ideal generated by two linear functions l_1 and l_2. Further, we say that l_1 is *independent* of l_2 if $1 \notin (l_1, l_2)$.

Theorem 1.8 (restated.) *Let $f \in F[x_1, \ldots, x_n]$ be a polynomial such that there are no two linear functions l_1 and l_2 (with $1 \notin (l_1, l_2)$) which make $f \bmod (l_1, l_2)$ also a linear function. Then f is not computable by a depth 2 circuit over $\mathsf{U}_2(\mathbb{F})$.*

Proof. Assume on the contrary that f can be computed by a depth 2 circuit over $\mathsf{U}_2(\mathbb{F})$. In other words, there is a product sequence $M_1 \cdots M_t$ of 2×2 upper-triangular linear matrices such that f appears as the top-right entry of the final product. Let $M_i = \begin{bmatrix} l_{i1} & l_{i2} \\ & l_{i3} \end{bmatrix}$, then

$$f = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} l_{11} & l_{12} \\ & l_{13} \end{bmatrix} \begin{bmatrix} l_{21} & l_{22} \\ & l_{23} \end{bmatrix} \cdots \begin{bmatrix} l_{t1} & l_{t2} \\ & l_{t3} \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{4}$$

Case 1: Not all the l_{i1}'s are constants.

Let k be the least index such that l_{k1} is not a constant and $l_{i1} = c_i$ for all $i < k$. To simplify Equation 4, let

$$\begin{bmatrix} B \\ L \end{bmatrix} = M_{k+1} \cdots M_t \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$
$$\begin{bmatrix} d_i & D_i \end{bmatrix} = \begin{bmatrix} 1 & 0 \end{bmatrix} \cdot M_1 \cdots M_{i-1}$$

Observe that L is just a product of linear functions, and for all $1 \leq i < k$, we have the following relations.

$$d_{i+1} = \prod_{j=1}^{i} c_j$$
$$D_{i+1} = d_i l_{i2} + l_{i3} D_i$$

Hence, Equation 4 simplifies as

$$\begin{aligned} f &= \begin{bmatrix} d_k & D_k \end{bmatrix} \begin{bmatrix} l_{k1} & l_{k2} \\ & l_{k3} \end{bmatrix} \begin{bmatrix} B \\ L \end{bmatrix} \\ &= d_k l_{k1} B + (d_k l_{k2} + l_{k3} D_k) L \end{aligned}$$

Suppose there is some factor l of L with $1 \notin (l_{k1}, l)$. Then $f = 0 \bmod (l_{k1}, l)$, which is not possible. Hence, L must be a constant modulo l_{k1}. For appropriate constants α, β, we have

$$f = \alpha l_{k2} + \beta l_{k3} D_k \pmod{l_{k1}} \tag{5}$$

We argue that the above equation cannot be true by inducting on k. If l_{k3} was independent of l_{k1}, then $f = \alpha l_{k2} \bmod (l_{k1}, l_{k3})$ which is not possible. Therefore, l_{k3} must be a constant modulo l_{k1}. We then have the following (reusing α and β to denote appropriate constants):

$$\begin{aligned} f &= \alpha l_{k2} + \beta D_k \pmod{l_{k1}} \\ &= \alpha l_{k2} + \beta \left(d_{k-1} l_{(k-1)2} + l_{(k-1)3} D_{k-1} \right) \pmod{l_{k1}} \\ \implies f &= \left(\alpha l_{k2} + \beta d_{k-1} l_{(k-1)2} \right) + \beta l_{(k-1)3} D_{k-1} \pmod{l_{k1}} \end{aligned}$$

The last equation can be rewritten in the form of Equation 5 with $\beta l_{k3} D_k$ replaced by $\beta l_{(k-1)3} D_{k-1}$. Notice that the expression $\left(\alpha l_{k2} + \beta d_{k-1} l_{(k-1)2} \right)$ is linear just like αl_{k2}. Hence by using the argument iteratively we eventually get a contradiction at D_1.

Case 2: All the l_{i1}'s are constants.

In this case, Equation 4 can be rewritten as

$$\begin{aligned} f &= \begin{bmatrix} d_t & D_t \end{bmatrix} \begin{bmatrix} c_t & l_{t2} \\ & l_{t3} \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} \\ &= d_t l_{t2} + l_{t3} D_t \end{aligned}$$

The last equation is again of the form in Equation 5 (without the mod term) and hence the same argument can be repeated here as well to give the desired contradiction. □

The following corollary provides some explicit examples of functions that cannot be computed.

Corollary 4.1. *A depth 2 circuit over $\mathsf{U}_2(\mathbb{F})$ cannot compute the polynomial $x_1x_2 + x_3x_4 + x_5x_6$. Other examples include well known functions like det_n and perm_n, the determinant and permanent polynomials, for $n \geq 3$.*

Proof. It suffices to show that $f = x_1x_2 + x_3x_4 + x_5x_6$ satisfy the requirement in Theorem 1.8.

To obtain a contradiction, let us assume that there does exist two linear functions l_1 and l_2 (with $1 \notin (l_1, l_2)$) such that f mod (l_1, l_2) is linear. We can evaluate f mod (l_1, l_2) by substituting a pair of the variables in f by linear functions in the rest of the variables (as dictated by the equations $l_1 = l_2 = 0$). By the symmetry of f, we can assume that the pair is either $\{x_1, x_2\}$ or $\{x_1, x_3\}$.

If $x_1 = l_1'$ and $x_3 = l_2'$ are the substitutions, then $l_1'x_2 + l_2'x_4$ can never contribute a term to cancel off x_5x_6 and hence f mod (l_1, l_2) cannot be linear.

Otherwise, let $x_1 = l_1'$ and $x_2 = l_2'$ be the substitutions. If f mod $(l_1, l_2) = l_1'l_2' + x_3x_4 + x_5x_6$ is linear, there cannot be a common x_i with non-zero coefficient in both l_1' and l_2'. Without loss of generality, assume that l_1' involves x_3 and x_5 and l_2' involves x_4 and x_6. But then the product $l_1'l_2'$ would involve terms like x_3x_6 that cannot be cancelled, contradicting linearity again. □

4.2 Depth 2 model over $\mathsf{M}_2(\mathbb{F})$

In this section we show that the power of depth 2 circuits is very restrictive even if we take the underlying algebra to be $\mathsf{M}_2(\mathbb{F})$ instead of $\mathsf{U}_2(\mathbb{F})$. In the following discussion, we will refer to a homogeneous linear function as a *linear form*.

Definition 4.2. *A polynomial f of degree n is said to be r-*robust *if f does not belong to any ideal generated by r linear forms.*

For instance, it can be checked that det_n and perm_n, the symbolic determinant and permanent of an $n \times n$ matrix, are $(n-1)$-robust polynomials. For any polynomial f, we will denote the d^{th} homogeneous part of f by $[f]_d$. And let $(h_1, \cdots, h_k)$ denote the ideal generated by $h_1, \cdots, h_k$. For the following theorem recall the definition of *degree restriction* (Definition 1.9) given in Section 1.

Theorem 4.3. *A polynomial f of degree n, such that $[f]_n$ is 5-robust, cannot be computed by a depth 2 circuit over $\mathsf{M}_2(\mathbb{F})$ under a degree restriction of n.*

We prove this with the help of the following lemma, which basically applies Gaussian column operations to simplify matrices.

Lemma 4.4. *Let f_1 be a polynomial of degree n such that $[f_1]_n$ is 4-robust. Suppose there is a linear matrix M and polynomials f_2, g_1, g_2 of degree at most n satisfying*

$$\begin{bmatrix} f_1 \\ f_2 \end{bmatrix} = M \begin{bmatrix} g_1 \\ g_2 \end{bmatrix}$$

Then, there is an appropriate invertible column operation A such that

$$M \cdot A = \begin{bmatrix} 1 & h_2 \\ c_3 & h_4 + c_4 \end{bmatrix}$$

where c_3, c_4 are constants and h_2, h_4 are linear forms.

We will defer the proof of this lemma to the end of this section, and shall use it to prove Theorem 4.3.

Proof of Theorem 4.3. Assume, on the contrary, that we do have such a sequence of matrices computing f. Since only one entry is of interest to us, we shall assume that the first matrix is a row vector and the last matrix is a column vector. Let the sequence of minimum length computing f be the following:

$$f = \bar{v} \cdot M_1 M_2 \cdots M_d \cdot \bar{w}$$

Using Lemma 4.4 we shall repeatedly transform the above sequence by replacing $M_i M_{i+1}$ by $(M_i A)(A^{-1} M_{i+1})$ for an appropriate invertible column transformation A. Since A would consist of just constant entries, $M_i A$ and $A^{-1} M_{i+1}$ continue to be linear matrices.

To begin, let $\bar{v} = [l_1, l_2]$ for two linear functions l_1 and l_2. And let $[f_1, f_2]^T = M_1 \cdots M_d \bar{w}$. Then we have,

$$\begin{bmatrix} f \\ 0 \end{bmatrix} = \begin{bmatrix} l_1 & l_2 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \end{bmatrix}$$

Hence, by Lemma 4.4, we can assume $\bar{v} = [1, h]$ and hence $f = f_1 + h f_2$. By the minimality of the sequence, $h \neq 0$. This forces f_1 to be 4-robust and the degree restriction makes $[f_2]_n = 0$.

Let $[g_1, g_2]^T = M_2 \cdots M_d \bar{w}$. The goal is to translate the properties that $[f_1]_n$ is 4-robust and $[f_2]_n = 0$ to the polynomials g_1 and g_2. Translating these properties would show each M_i is of the form described in Lemma 4.4. Thus, inducting on the length of the sequence, we would arrive at the required contradiction. In general, we have an equation of the form

$$\begin{bmatrix} f_1 \\ f_2 \end{bmatrix} = M_i \begin{bmatrix} g_1 \\ g_2 \end{bmatrix}$$

Since $[f_1]_n$ is 4-robust, using Lemma 4.4 again, we can assume that

$$\begin{bmatrix} f_1 \\ f_2 \end{bmatrix} = \begin{bmatrix} 1 & h_2 \\ c_3 & c_4 + h_4 \end{bmatrix} \begin{bmatrix} g_1 \\ g_2 \end{bmatrix} \tag{6}$$

by reusing the variables g_1, g_2 and others. Observe that in the above equation if $h_4 = 0$ then $M_{i-1} M_i$ still continues to be a linear matrix (since, by induction, M_{i-1} is of the form as dictated by Lemma 4.4) and that would contradict the minimality of the sequence. Therefore $h_4 \neq 0$.

Claim: $c_3 = 0$ (by comparing the n^{th} homogeneous parts of f_1 and g_1, as explained below).
Proof: As $h_4 \neq 0$, the degree restriction forces $\deg g_2 < n$. And since $\deg f_2 < n$, we have the relation $c_3 [g_1]_n = -h_4 [g_2]_{n-1}$. If $c_3 \neq 0$, we have $[g_1]_n \in (h_4)$, contradicting robustness of $[f_1]_n$ as then $[f_1]_n = [g_1]_n + h_2 [g_2]_{n-1} \in (h_2, h_4)$. □

Therefore Equation 6 gives,

$$\begin{bmatrix} f_1 \\ f_2 \end{bmatrix} = \begin{bmatrix} 1 & h_2 \\ 0 & c_4 + h_4 \end{bmatrix} \begin{bmatrix} g_1 \\ g_2 \end{bmatrix}$$

with $h_4 \neq 0$. Also, since $[f_2]_{n+1} = [f_2]_n = 0$ this implies that $[g_2]_n = [g_2]_{n-1} = 0$. Hence, $[g_1]_n = [f_1]_n$ is 4-robust. This argument can be extended now to g_1 and g_2. Notice that the degree of g_1 remains n. However, since there are only finitely many matrices in the sequence, there must come a point when this degree drops below n. At this point we get a contradiction as $[g_1]_n = 0$ (reusing symbol) which contradicts robustness. □

We only need to finish the proof of Lemma 4.4.

Proof of Lemma 4.4. Suppose we have an equation of the form

$$\begin{bmatrix} f_1 \\ f_2 \end{bmatrix} = \begin{bmatrix} h_1 + c_1 & h_2 + c_2 \\ h_3 + c_3 & h_4 + c_4 \end{bmatrix} \begin{bmatrix} g_1 \\ g_2 \end{bmatrix} \tag{7}$$

On comparing degree $n+1$ terms, we have

$$\begin{aligned} h_1[g_1]_n + h_2[g_2]_n &= 0 \\ h_3[g_1]_n + h_4[g_2]_n &= 0 \end{aligned}$$

If h_3 and h_4 (a similar reasoning holds for h_1 and h_2) were not proportional (i.e. not multiple of each other), then the above equation would imply $[g_1]_n, [g_2]_n \in (h_3, h_4)$. Then,

$$[f_1]_n = h_1[g_1]_{n-1} + h_2[g_2]_{n-1} + c_1[g_1]_n + c_2[g_2]_n \in (h_1, h_2, h_3, h_4)$$

contradicting the robustness of $[f_1]_n$. Thus, h_3 and h_4 (as well as h_1 and h_2) are proportional, in the same ratio as $[-g_2]_n$ and $[g_1]_n$. Using an appropriate column operation, Equation 7 simplifies to

$$\begin{bmatrix} f_1 \\ f_2 \end{bmatrix} = \begin{bmatrix} c_1 & h_2 + c_2 \\ c_3 & h_4 + c_4 \end{bmatrix} \begin{bmatrix} g_1 \\ g_2 \end{bmatrix}$$

If $c_1 = 0$, then together with $[g_2]_n = 0$ we get $[f_1]_n = h_2[g_2]_{n-1}$ contradicting robustness. Therefore $c_1 \neq 0$ and another column transformation would get it to the form claimed. □

5 Concluding remarks

We give a new perspective to identity testing of depth 3 arithmetic circuits by showing an equivalence to identity testing of depth 2 circuits over $\mathsf{U}_2(\mathbb{F})$. The reduction implies that identity testing of a width-2 algebraic branching program is at least as hard as identity testing of depth 3 circuits.

We also give a deterministic polynomial time identity testing algorithm for depth 2 circuits over any constant dimensional commutative algebra. Our algorithm crucially exploits an interesting structural result involving local rings. This naturally poses the following question - Can we use more algebraic insight on non-commutative algebras to solve the general problem? The solution for the commutative case does not seem to give any interesting insight into the non-commutative case. But we have a very specific non-commutative case at hand. The question is - Is it possible to use properties very specific to the ring of 2×2 matrices to solve identity testing for depth 3 circuits?

Acknowledgement

This work was started when the first author visited Hausdorff Center for Mathematics, Bonn. We thank Marek Karpinski for the generous hospitality and several discussions. We also thank Manindra Agrawal for several insightful discussions on this work. And finally thanks to V Vinay for many useful comments on the first draft of this paper.

References

[AB99] Manindra Agrawal and Somenath Biswas. Primality and Identity Testing via Chinese Remaindering. In *FOCS*, pages 202–209, 1999.

[Agr05] Manindra Agrawal. Proving Lower Bounds Via Pseudo-random Generators. In *FSTTCS*, pages 92–105, 2005.

[AJMV98] Eric Allender, Jia Jiao, Meena Mahajan, and V. Vinay. Non-Commutative Arithmetic Circuits: Depth Reduction and Size Lower Bounds. *Theor. Comput. Sci.*, 209(1-2):47–86, 1998.

[AKS04] Manindra Agrawal, Neeraj Kayal, and Nitin Saxena. PRIMES is in P. *Ann. of Math*, 160(2):781–793, 2004.

[ALM+98] Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof Verification and the Hardness of Approximation Problems. *Journal of the ACM*, 45(3):501–555, 1998.

[AMS08] Vikraman Arvind, Partha Mukhopadhyay, and Srikanth Srinivasan. New results on noncommutative and commutative polynomial identity testing. In *IEEE Conference on Computational Complexity*, pages 268–279, 2008.

[AV08] Manindra Agrawal and V Vinay. Arithmetic circuits: A chasm at depth four. In *FOCS*, pages 67–75, 2008.

[BC88] Michael Ben-Or and Richard Cleve. Computing Algebraic Formulas Using a Constant Number of Registers. In *STOC*, pages 254–257, 1988.

[BW05] Andrej Bogdanov and Hoeteck Wee. More on noncommutative polynomial identity testing. In *IEEE Conference on Computational Complexity*, pages 92–99, 2005.

[CDGK91] Michael Clausen, Andreas W. M. Dress, Johannes Grabmeier, and Marek Karpinski. On Zero-Testing and Interpolation of k-Sparse Multivariate Polynomials Over Finite Fields. *Theor. Comput. Sci.*, 84(2):151–164, 1991.

[CK97] Zhi-Zhong Chen and Ming-Yang Kao. Reducing Randomness via Irrational Numbers. In *STOC*, pages 200–209, 1997.

[CS04] Steve Chien and Alistair Sinclair. Algebras with polynomial identities and computing the determinant. In *FOCS*, pages 352–361, 2004.

[KI03] Valentine Kabanets and Russell Impagliazzo. Derandomizing polynomial identity tests means proving circuit lower bounds. In *STOC*, pages 355–364, 2003.

[KS01] Adam Klivans and Daniel A. Spielman. Randomness efficient identity testing of multivariate polynomials. In *STOC*, pages 216–223, 2001.

[KS07] Neeraj Kayal and Nitin Saxena. Polynomial Identity Testing for Depth 3 Circuits. *Computational Complexity*, 16(2), 2007.

[Lov79] László Lovász. On determinants, matchings, and random algorithms. In *FCT*, pages 565–574, 1979.

[LV98] Daniel Lewin and Salil P. Vadhan. Checking Polynomial Identities over any Field: Towards a Derandomization? In *STOC*, pages 438–447, 1998.

[Nis91] Noam Nisan. Lower bounds for non-commutative computation. In *STOC*, pages 410–418, 1991.

[RS04] Ran Raz and Amir Shpilka. Deterministic Polynomial Identity Testing in Non-Commutative Models. In *IEEE Conference on Computational Complexity*, pages 215–222, 2004.

[Sch80] Jacob T. Schwartz. Fast Probabilistic Algorithms for Verification of Polynomial Identities. *J. ACM*, 27(4):701–717, 1980.

[Sha90] Adi Shamir. IP=PSPACE. In *FOCS*, pages 11–15, 1990.

[VSBR83] Leslie G. Valiant, Sven Skyum, S. Berkowitz, and Charles Rackoff. Fast parallel computation of polynomials using few processors. *SIAM J. Comput.*, 12(4):641–644, 1983.

[Zip79] Richard Zippel. Probabilistic algorithms for sparse polynomials. *EUROSAM*, pages 216–226, 1979.

A Appendix

For the sake of completeness, we provide a proof of the result by Ben-Or and Cleve [BC88].

Theorem A.1. *[BC88] Let E be an arithmetic formula of depth d with fan-in (of every gate) bounded by 2. Then, there exists a sequence of 3×3 matrices, whose entries are either variables or constants, of length at most 4^d such that one of the entries of their product is E.*

Proof. The proof is by induction on the structure of E. The base case when $E = c \cdot x_i$ is computed as,

$$\begin{bmatrix} 1 & & \\ & 1 & \\ c \cdot x_i & & 1 \end{bmatrix}$$

Suppose $E = f_1 + f_2$ and that we have inductively constructed sequences computing f_1 and f_2. Then the following equation gives a sequence for E.

$$\begin{bmatrix} 1 & & \\ & 1 & \\ f_1 & & 1 \end{bmatrix} \begin{bmatrix} 1 & & \\ & 1 & \\ f_2 & & 1 \end{bmatrix} = \begin{bmatrix} 1 & & \\ & 1 & \\ f_1 + f_2 & & 1 \end{bmatrix}$$

If $E = f_1 \cdot f_2$, then the following sequence computes E

$$\begin{bmatrix} 1 & & \\ -f_2 & 1 & \\ & & 1 \end{bmatrix} \begin{bmatrix} 1 & & \\ & 1 & \\ & f_1 & 1 \end{bmatrix} \begin{bmatrix} 1 & & \\ f_2 & 1 & \\ & & 1 \end{bmatrix} \begin{bmatrix} 1 & & \\ & 1 & \\ & -f_1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & & \\ & 1 & \\ f_1 f_2 & & 1 \end{bmatrix}$$

Applying the above two equations inductively, it is clear that E can be computed by a sequence of length at most 4^d. □

ECCC ISSN 1433-8092

http://eccc.hpi-web.de/