Electronic Colloquium on Computational Complexity, Revision 1 of Report No. 74 (2017)

Efficient Identity Testing and Polynomial Factorization over Non-associative Free Rings

V. Arvind* Rajit Datta † Partha Mukhopadhyay ‡ S. Raja§

July 6, 2017

Abstract

In this paper we study arithmetic computations in the nonassociative, and noncommutative free polynomial ring $\mathbb{F}\{x_1, x_2, \ldots, x_n\}$. Prior to this work, nonassociative arithmetic computation was considered by Hrubes, Wigderson, and Yehudayoff [HWY10], and they showed lower bounds and proved completeness results. We consider Polynomial Identity Testing (PIT) and polynomial factorization over $\mathbb{F}\{x_1, x_2, \ldots, x_n\}$ and show the following results.

1. Given an arithmetic circuit C of size s computing a polynomial $f \in \mathbb{F}\{x_1, x_2, \ldots, x_n\}$ of degree d, we give a deterministic poly(n, s, d) algorithm to decide if f is identically zero polynomial or not. Our result is obtained by a suitable adaptation of the PIT algorithm of Raz-Shpilka [RS05] for noncommutative ABPs.
2. Given an arithmetic circuit C of size s computing a polynomial $f \in \mathbb{F}\{x_1, x_2, \ldots, x_n\}$ of degree d, we give an efficient deterministic algorithm to compute circuits for the irreducible factors of f in time poly(n, s, d) when $\mathbb{F} = \mathbb{Q}$. Over finite fields of characteristic p, our algorithm runs in time poly(n, s, d, p).

1 Introduction

Noncommutative computation, introduced in complexity theory by Hyafil [Hya77] and Nisan [Nis91], is an important subfield of algebraic complexity theory. The main algebraic structure of interest is the free noncommutative ring $\mathbb{F}\langle X\rangle$ over a field $\mathbb{F}$, where $X = \{x_1, x_2, \cdots, x_n\}$ is a set of free noncommuting variables. A central problem is Polynomial Identity Testing which may be stated as follows:

Let $f \in \mathbb{F}\langle X\rangle$ be a polynomial represented by a noncommutative arithmetic circuit C. The circuit C can either be given by a black box (using which we can evaluate C on matrices with entries from $\mathbb{F}$ or an extension field), or the circuit may be explicitly given. The algorithmic problem is to check if the polynomial computed by C is identically zero. We recall the formal definition of a noncommutative arithmetic circuit.

*Institute of Mathematical Sciences (HBNI), Chennai, India, `email: arvind@imsc.res.in`
†Chennai Mathematical Institute, Chennai, India, `email: rajit@cmi.ac.in`
‡Chennai Mathematical Institute, Chennai, India, `email: partham@cmi.ac.in`
§Chennai Mathematical Institute, Chennai, India, `email: sraja@cmi.ac.in`

ISSN 1433-8092

Definition 1. *An* arithmetic circuit C *over a field* $\mathbb{F}$ *and indeterminates* $X = \{x_1, x_2, \cdots, x_n\}$ *is a directed acyclic graph (DAG) with each node of indegree zero labeled by a variable or a scalar constant from* $\mathbb{F}$*: the indegree* 0 *nodes are the input nodes of the circuit. Each internal node of the DAG is of indegree two and is labeled by either a* $+$ *or a* $\times$ *(indicating that it is a plus gate or multiply gate, respectively). Furthermore, the two inputs to each* $\times$ *gate are designated as left and right inputs which prescribes the order of multiplication at that gate. A gate of* C *is designated as* output*. Each internal gate computes a polynomial (by adding or multiplying its input polynomials), where the polynomial computed at an input node is just its label. The* polynomial computed *by the circuit is the polynomial computed at its output gate.*

When the multiplication operation of the circuit in Definition 1 is *noncommutative*, it is called a *noncommutative arithmetic circuit* and it computes a polynomial in the free noncommutative ring $\mathbb{F}\langle X\rangle$. Since cancellation of terms is restricted by noncommutativity, intuitively it appears noncommutative polynomial identity testing would be easier than polynomial identity testing in the commutative case. This intuition is supported by fact that there is a deterministic polynomial-time white-box PIT algorithm for noncommutative ABP [RS05]. In the commutative setting a deterministic polynomial-time PIT for ABPs would be a major breakthrough.[1] However, there is little progress towards obtaining an efficient deterministic PIT for general noncommutative arithmetic circuits. For example, the problem is open even for noncommutative *skew* circuits.

If *associativity* is also dropped then it turns out that PIT becomes easy, as we show in this work. More precisely, we consider the free noncommutative and nonassociative ring of polynomials $\mathbb{F}\{X\}$, $X = \{x_1, x_2, \ldots, x_n\}$, where a polynomial is an $\mathbb{F}$-linear combination of monomials, and each monomial comes with a bracketing order of multiplication. For example, in the nonassociative ring $\mathbb{F}\{X\}$ the monomial $(x_1(x_2x_1))$ is different from monomial $((x_1x_2)x_1)$, although in the associative ring $\mathbb{F}\langle X\rangle$ they clearly coincide.

When the multiplication operation is both noncommutative and nonassociative, it is called a *nonassociative noncommutative circuit* and it computes a polynomial in the free nonassociative noncommutative ring $\mathbb{F}\{X\}$. Previously, the nonassociative arithmetic model of computation was considered by Hrubes, Wigderson, and Yehudayoff [HWY10]. They showed completeness and explicit lower bound results for this model. We show the following result about PIT.

- Let $f(x_1, x_2, \ldots, x_n) \in \mathbb{F}\{X\}$ be a degree d polynomial given by an arithmetic circuit of size s. Then in deterministic $\text{poly}(s, n, d)$ time we can test if f is an identically zero polynomial in $\mathbb{F}\{X\}$.

Remark 1. *We note that our algorithm in the above result does not depend on the choice of the field* $\mathbb{F}$*. A recent result of Lagarde et al. [LMP16] shows an exponential lower bound, and a deterministic polynomial-time* PIT *algorithm over* $\mathbb{R}$ *for noncommutative circuits where all parse trees in the circuit are isomorphic. We also note that in [AR16] an exponential lower bound is shown for set-multilinear arithmetic circuits with the additional semantic constraint that each monomial has a unique parse tree in the circuit (but different monomials can have different parse trees).*

Next, we consider polynomial factorization in the ring $\mathbb{F}\{\text{X}\}$. Polynomial factorization is very well-studied in the commutative ring $\mathbb{F}[X]$: Given an arithmetic circuit C computing a multivariate

[1] The situation is similar even in the lower bound case where Nisan proved that noncommutative determinant or permanent polynomial would require exponential-size algebraic branching program [Nis91].

polynomial $f \in \mathbb{F}[\mathrm{X}]$ of degree d, the problem is to efficiently compute circuits for the irreducible factors of f. A celebrated result of Kaltofen [Kal] solves the problem in randomized $\mathrm{poly}(n, s, d)$ time. Whether there is a polynomial-time deterministic algorithm is an outstanding open problem. Recently, it is shown (for fields of small characteristic and characteristic zero) that the complexity of deterministic polynomial factorization problem and the PIT problem are polynomially equivalent [KSS15]. A natural question is to determine the complexity of polynomial factorization in the noncommutative ring $\mathbb{F}\langle X\rangle$. The free noncommutative ring $\mathbb{F}\langle X\rangle$ is not even a *unique factorization domain* [Coh85]. However, unique factorization holds for homogeneous polynomials in $\mathbb{F}\langle X\rangle$, and it is shown in [AJR15] that for homogeneous polynomials given by noncommutative circuits, the unique factorization into irreducible factors can be computed in randomized polynomial time (essentially, by reduction to the noncommutative PIT problem).

In this paper, we note that the ring $\mathbb{F}\{X\}$ is a *unique factorization domain*, and given a polynomial in $\mathbb{F}\{X\}$ by a circuit, we show that circuits for all its irreducible factors can be computed in deterministic polynomial time.

- Let $f(x_1, x_2, \ldots, x_n) \in \mathbb{F}\{X\}$ be a degree d polynomial given by an arithmetic circuit of size s. Then if $\mathbb{F} = \mathbb{Q}$, in deterministic $\mathrm{poly}(s, n, d)$ time we can output the circuits for the irreducible factors of f. If $\mathbb{F}$ is a finite field such that $char(\mathbb{F}) = p$, we obtain a deterministic $\mathrm{poly}(s, n, d, p)$ time algorithm for computing circuits for the irreducible factors of f.

Outline of the proofs

- **Identity Testing Result:** The main ideas for our algorithm are based on the white-box Raz-Shpilka PIT algorithm for noncommutative ABPs [RS05]. As in the Raz-Shpilka algorithm [RS05], if the circuit computes a nonzero polynomial $f \in \mathbb{F}\{X\}$, then our algorithm output a *certificate monomial* m such that coefficient of m in f is nonzero.

  We first sketch the main steps of the Raz-Shpilka algorithm. The Raz-Shpilka algorithm processes the input ABP (assumed homogeneous) layer by layer. Suppose layer i of the ABP has w nodes. The algorithm maintains a spanning set $\mathbb{B}_i$ of at most w many linearly independent w-dimensional vectors of monomial coefficients. More precisely, each vector $v_m \in \mathbb{B}_i$ is the vector of coefficients of monomial m computed at each of the w nodes in layer i. Furthermore, the coefficient vector at layer i of any monomial is in the span of $\mathbb{B}_i$. The construction of $\mathbb{B}_{i+1}$ from $\mathbb{B}_i$ can be done efficiently. Clearly the identity testing problem can be solved by checking if there is a nonzero vector in $\mathbb{B}_d$, where d is the total number of layers.

  Now we sketch our PIT algorithm for polynomials over $\mathbb{F}\{\mathrm{X}\}$ given by circuits. Let f be the input polynomial given by the circuit C.

  We encode monomials in the free nonassociative noncommutative ring $\mathbb{F}\{\mathrm{X}\}$ as monomials in the free noncommutative ring $\mathbb{F}\langle \mathrm{X}, (,) \rangle$, such that the encoding preserves the multiplication structure of $\mathbb{F}\{\mathrm{X}\}$ (Observation 1). For $1 \le j \le d$, we can efficiently find from C a homogeneous circuit C_j that computes the degree j homogeneous part of C. Thus, it suffices to test if $C_j \equiv 0$ for each j. Hence, it suffices to consider the case when $f \in \mathbb{F}\{\mathrm{X}\}$ is homogeneous and C is a homogeneous circuit computing f.

  For $j \le d$ let G_j denote the set of degree j gates of C. The algorithm maintains a set $\mathbb{B}_j$ of $|G_j|$-dimensional linearly independent vectors of monomial coefficients such that any degree j monomial's coefficient vector is in the linear span of $\mathbb{B}_j$. Clearly, $|\mathbb{B}_j| \le |G_j|$. We compute

$\mathbb{B}_{j+1}$ from the sets $\{\mathbb{B}_i : 1 \leq i \leq j\}$. For each vector in $\mathbb{B}_j$ we also keep the corresponding monomial. In the nonassociative model a degree d monomial $m = (m_1 m_2)$ is generated in a *unique way*. To check if the coefficient vector of m is in the span of $\mathbb{B}_d$ it suffices to consider vectors in the spans of $\mathbb{B}_{d_1}$ and $\mathbb{B}_{d_2}$, where $d_1 = \deg(m_1)$ and $d_2 = \deg(m_2)$. This is a crucial difference from a general noncommutative circuit and using this property we can compute $\mathbb{B}_{j+1}$.

- **Polynomial Factorization in $\mathbb{F}\{X\}$**

  For a polynomial $f \in \mathbb{F}\{\mathrm{X}\}$, let f_j denote the homogeneous degree j part of f. For a monomial m, let $c_m(f)$ denote the coefficient of m in f. We will use the PIT algorithm as subroutine for the factoring algorithm. Arvind et al. [AJR15] have shown that given a monomial m and a homogeneous noncommutative circuit C, in deterministic polynomial time circuits for the formal left and right derivatives of C with respect to m can be efficiently computed. This result is another ingredient in our algorithm.

  We sketch the easy case, when the given polynomial f of degree d has no constant term. Applying our PIT algorithm to the homogeneous circuit C_d (computing f_d) we find a nonzero monomial $m = (m_1\ m_2)$ of degree d in f_d along with its coefficient $c_m(f)$. Notice that for any nontrivial factorization $f = gh$, m_1 is a nonzero monomial in g and m_2 is a nonzero monomial in h. Suppose $|m_1| = d_1$ and $|m_2| = d_2$. Then the left derivative of C_d with respect to m_1 gives $c_{m_1}(g)\ h_{d_2}$ and the right derivative of C_d with respect to m_2 gives $c_{m_2}(h)\ g_{d_1}$. We now use the circuits for these derivatives and the nonassociative structure, to find circuits for different homogeneous parts of g and h. The details, including the general case when f has a nonzero constant term, is in Section 4.

Organization

In Section 2 we describe some useful properties of nonassociative and noncommutative polynomials. In Section 3 we give the PIT algorithm for $\mathbb{F}\{X\}$. In Section 4 we describe the factorization algorithm for $\mathbb{F}\{X\}$. Finally, we list some open problems in Section 5.

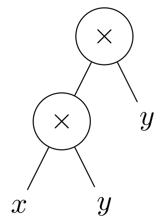
2 Preliminaries

For an arithmetic circuit C, a *parse tree* for a monomial m is a multiplicative sub-circuit of C rooted at the output gate defined by the following process starting from the output gate:

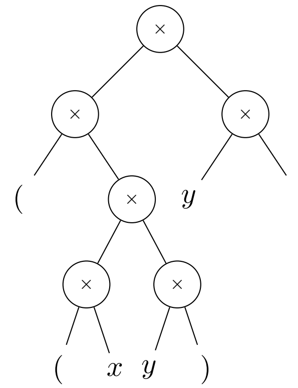
- At each $+$ gate retain exactly one of its input gates.
- At each $\times$ gate retain both its input gates.
- Retain all inputs that are reached by this process.
- The resulting subcircuit is multiplicative and computes a monomial m (with some coefficient).

For arithmetic circuits C computing polynomials in the free nonassociative noncommutative ring $\mathbb{F}\{X\}$, the same definition for the parse tree of a monomial applies. As explained in the introduction, in this case each parse tree (generating some monomial) comes with a bracketed structure for the multiplication. It is convenient to consider a polynomial in $\mathbb{F}\{x_1, \ldots, x_n\}$ as an element in the noncommutative ring $\mathbb{F}\langle x_1, \ldots, x_n, (,)\rangle$ where we introduce two auxiliary variables (and) (for left and right bracketing) to encode the parse tree structure of any monomial. We illustrate the encoding by the following example.

Consider the monomial (which is essentially a binary tree with leaves labeled by variables) in the nonassociative ring $\mathbb{F}\{x, y\}$ shown in Figure 1a. Its encoding as a bracketed string in the free noncommutative ring $\mathbb{F}\langle x, y, (,)\rangle$ is ((x y) y) and its parse tree shown in Figure 1b.



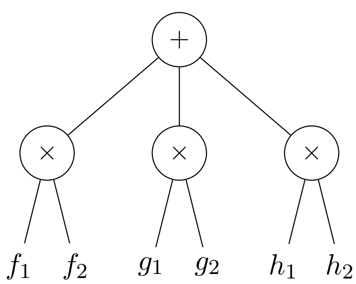
(a) A nonassociative and noncommutative monomial xyy



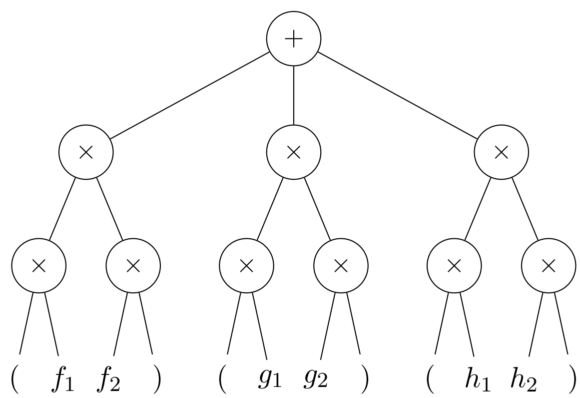
(b) Corresponding monomial $((xy)\ y) \in \mathbb{F}\langle X\rangle$.

Figure 1: nonassociative & noncommutative monomial and its corresponding noncommutative bracketed monomial

Consider an arithmetic circuit C computing a polynomial $f \in \mathbb{F}\{X\}$. The circuit C can be efficiently transformed to a circuit $\tilde{C}$ that computes the corresponding polynomial $\tilde{f} \in \mathbb{F}\langle X, (,)\rangle$ by simply introducing the bracketing structure for each multiplication gate of C in a bottom-up manner as indicated in the following example figures. Consider the circuits described in Figures 2a and 2b where f_i, g_i, h_i's are polynomials computed by subcircuits. Clearly the bracket variables preserve the parse tree structure. The following fact is immediate.



(a) C computing a nonassociative, noncommutative polynomial.



(b) $\tilde{C}$ that computes the corresponding noncommutative polynomial.

Figure 2: Nonassociative circuit and its corresponding noncommutative bracketed circuit

Observation 1. *A nonassociative noncommutative circuit C computes a nonzero polynomial $f \in \mathbb{F}\{X\}$ if and only if the corresponding noncommutative circuit $\tilde{C}$ computes a nonzero polynomial $\tilde{f} \in \mathbb{F}\langle X, (,)\rangle$.*

We recall that the free noncommutative ring $\mathbb{F}\langle \mathrm{X}\rangle$ is not a unique factorization domain (UFD) [Coh85] as shown by the following standard example : $xyx+x = x(yx+1) = (xy+1)x$. In contrast, the nonassociative free ring $\mathbb{F}\{\mathrm{X}\}$ is a UFD.

Proposition 1. *Over any field $\mathbb{F}$, the ring $\mathbb{F}\{\mathrm{X}\}$ is a unique factorization domain. More precisely, any polynomial $f \in \mathbb{F}\{X\}$ can be expressed a product $f = g_1 g_2 \cdots g_r$ of irreducible polynomials $g_i \in \mathbb{F}\{X\}$. The factorization is unique upto constant factors and reordering.*

Remark 2. *Usually, even the ordering of the irreducible factors in the factorization is unique. Exceptions arise because of the equality $(g+\alpha)(g+\beta) = (g+\beta)(g+\alpha)$ for any polynomial $g \in \mathbb{F}\{X\}$ and $\alpha, \beta \in \mathbb{F}$.*

We shall indirectly see a proof of this proposition in Section 4 where we describe the algorithm for computing all irreducible factors.

Given a noncommutative circuit C computing a homogeneous polynomial in $\mathbb{F}\langle \mathrm{X}\rangle$ and a monomial m over X, one can talk of the left and right derivatives of C w.r.t m [AJR15]. Let $f = \sum_{m'} c_{m'}(f)m'$ for some $f \in \mathbb{F}\langle \mathrm{X}\rangle$ and A be the subset of monomials m' of f that have m as prefix. Then the left derivative of f w.r.t. m is

$$\frac{\partial^{\ell} f}{\partial m} = \sum_{m' \in A} c_{m'}(f)m'',$$

where $m' = m \cdot m''$ for $m' \in A$. Similarly we can define the right derivative $\frac{\partial^{r} f}{\partial m}$. As shown in [AJR15], if f is given by a circuit C then in deterministic polynomial time we can compute circuits for $\frac{\partial^{\ell} f}{\partial m}$ and $\frac{\partial^{r} f}{\partial m}$. We briefly discuss this in the following lemma.

Lemma 1. [AJR15] *Given a noncommutative circuit C of size s computing a homogeneous polynomial f of degree d in $\mathbb{F}\langle \mathrm{X}\rangle$ and monomial m, there is a deterministic $\mathrm{poly}(n, d, s)$ time algorithm that computes circuits $C_{m,\ell}$ and $C_{m,r}$ for the left and right derivatives $\frac{\partial^{\ell} C}{\partial m}$ and $\frac{\partial^{r} C}{\partial m}$, respectively.*

Proof. We explain only the left partial derivative case. Let m be a degree d' monomial and $f \in \mathbb{F}\langle X\rangle$ be a homogeneous degree d polynomial f computed by circuit C. In [AJR15],a small substitution deterministic finite automaton A with $d'+2$ states is constructed that recognizes all length d strings with prefix m and substitutes 1 for prefix m. The transition matrices of this automaton can be represented by $(d' + 2) \times (d' + 2)$ matrices. From the evaluation of circuit C on these transition matrices will recover the circuit for $\frac{\partial^{\ell} C}{\partial m}$ in the $(1, d' + 1)^{th}$ entry of the output matrix. □

The left and right partial derivatives of inhomogeneous polynomials are similarly defined. The same matrix substitution works for non-homogeneous polynomials as well [AJR15]. As discussed above, given a nonassociative arithmetic circuit C computing a polynomial $f \in \mathbb{F}\{\mathrm{X}\}$, we can transform C into a noncommutative circuit $\tilde{C}$ that computes a polynomial $\tilde{f} \in \mathbb{F}\langle X, (,)\rangle$. Suppose we want to compute the left partial derivative of f w.r.t. a monomial $m \in \mathbb{F}\{X\}$. Using the tree structure of m we transform it into a monomial $\tilde{m} \in \mathbb{F}\langle X, (,)\rangle$ and then we can apply Lemma 1 to $\tilde{C}$ and $\tilde{m}$ to compute the required left partial derivative. We can similarly compute the right partial derivative. We use this in Section 4.

We also note the following simple fact that the homogeneous parts of a polynomial $f \in \mathbb{F}\{X\}$ given by a circuit C can be computed efficiently. We can apply the above transformation to obtain circuit $\tilde{C}$ and use a standard lemma (see e.g., [SY10]) to compute the homogeneous parts of $\tilde{C}$.

Lemma 2. *Given a noncommutative circuit C of size s computing a noncommutative polynomial f of degree d in $\mathbb{F}\langle \mathrm{X}, (,) \rangle$, one can compute homogeneous circuits C_j (where each gate computes a homogeneous polynomial) for j^{th} homogeneous part f_j of f, where $0 \leq j \leq d$, deterministically in time* $\mathrm{poly}(n, d, s)$.

3 Identity Testing in $\mathbb{F}\{X\}$

In this section we describe our identity testing algorithm.

Theorem 1. *Let $f(x_1, x_2, \ldots, x_n) \in \mathbb{F}\{X\}$ be a degree d polynomial given by an arithmetic circuit of size s. Then in deterministic* $\mathrm{poly}(s, n, d)$ *time we can test if f is an identically zero polynomial in $\mathbb{F}\{X\}$.*

Proof. By Lemma 2 we can assume that the input is a homogeneous nonassociative circuit C computing some homogeneous degree d polynomial in $\mathbb{F}\{X\}$ (i.e. every gate in C computes a homogeneous polynomial). Also, all the $\times$ gates in C have fanin 2 and $+$ gates have unbounded fanin. We can assume the output gate is a $+$ gate. We can also assume w.l.o.g. that the $+$ and $\times$ gates alternate in each input gate to output gate path in the circuit (otherwise we introduce sum gates with fan-in 1).

The j^{th}-layer of circuit C to be the set of all $+$ gates in computing degree j homogeneous polynomials. Let s^+ be the total number of $+$ gates in C. To each monomial m we can associate a vector $v_m \in \mathbb{F}^{s^+}$ of coefficients, where v_m is indexed by the $+$ gates in C, and $v_m[g]$ is the coefficient of monomial m in the polynomial computed at the $+$ gate g. We can also write

$$v_m[g] = c_m(p_g),$$

where p_g is the polynomial computed at the sum gate g.

For the j^{th} layer of $+$ gates, we will maintain a maximal linearly independent set $\mathbb{B}_j$ of vectors v_m of monomials. These vectors correspond to degree j monomials. Although $v_m \in \mathbb{F}^{s^+}$, notice that $v_m[g] = 0$ at all $+$ gates that do not compute a degree j polynomial. Thus, $|\mathbb{B}_\daleth|$ is bounded by the number of $+$ gates in the j^{th} layer. Hence, $|\mathbb{B}_\daleth| \leq s$.

The sets $\mathbb{B}_j$ are computed inductively for increasing values of j. For the base case, the set $\mathbb{B}_1$ can be easily constructed by direct computation. Inductively, suppose the sets $\mathbb{B}_i : 1 \leq i \leq j-1$ are already constructed. We describe the construction of $\mathbb{B}_j$. Computing $\mathbb{B}_d$ and checking if there is a nonzero vector in it yields the identity testing algorithm.

We now describe the construction for the j^{th} layer assuming we have basis $\mathbb{B}_{j'}$ for every $j' < j$. Consider a $\times$ gate with its children computing homogeneous polynomials of degree d_1 and d_2 respectively. Notice that $j = d_1 + d_2$ and $0 < d_1, d_2 < j$. Consider the monomial[2] set

[2]We note that the nonassociative monomial $m_1 m_2$ is a binary tree with the root having two children: the left child is the root of the binary tree for m_1 and the right child is the root of the binary tree for m_2.

$$M = \{m_1 m_2 \mid v_{m_1} \in \mathbb{B}_{d_1} \text{ and } v_{m_2} \in \mathbb{B}_{d_2}\}.$$

We construct vectors $\{v_m \mid m \in M\}$ as follows.

$$v_{m_1 m_2}[g] = \sum_{(g_{d_1}, g_{d_2})} v_{m_1}[g_{d_1}] v_{m_2}[g_{d_2}],$$

where g is a $+$ gate in the j^{th} layer, g_{d_1} is a $+$ gate in the d_1^{th} layer, g_{d_2} is a $+$ gate in the d_2^{th} layer, and there is a $\times$ gate which is input to g and computes the product of g_{d_1} and g_{d_2}.

Let $\mathbb{B}_{d_1,d_2}$ denote a maximal linearly independent subset of $\{v_m \mid m \in M\}$. Then we let $\mathbb{B}_d$ be a maximal linearly independent subset of

$$\bigcup_{d_1+d_2=d} \mathbb{B}_{d_1,d_2}.$$

Claim 1. *For every monomial m of degree j, v_m is in the span of $\mathbb{B}_j$.*

Proof of Claim. Let $m = m_1 m_2$ and the degree of m_1 is d_1 and the degree of m_2 is d_2 [3]. By *Induction Hypothesis* vectors v_{m_1} and v_{m_2} are in the span of $\mathbb{B}_{d_1}$ and $\mathbb{B}_{d_2}$ respectively. Hence, we can write

$$v_{m_1} = \sum_{i=1}^{D_1} \alpha_i v_{m_i} \quad v_{m_i} \in \mathbb{B}_{d_1} \quad \text{and} \quad v_{m_2} = \sum_{j=1}^{D_2} \beta_j v_{m'_j} \quad v_{m'_j} \in \mathbb{B}_{d_2},$$

where $|\mathbb{B}_{d_j}| = D_j$. Now, for a gate g in the j^{th} layer, By Induction Hypothesis and by construction we have

$$\begin{aligned} v_m[g] &= \sum_{(g_{d_1}, g_{d_2})} v_{m_1}[g_{d_1}] v_{m_2}[g_{d_2}] = \sum_{g_{d_1}, g_{d_2}} (\sum_{i=1}^{D_1} \alpha_i v_{m_i}[g_{d_1}])(\sum_{j=1}^{D_2} \beta_j v_{m'_j}[g_{d_2}]) \\ &= \sum_{i=1}^{D_1}\sum_{j=1}^{D_2} \alpha_i \beta_j \sum_{g_{d_1}, g_{d_2}} v_{m_i}[g_{d_1}] v_{m'_j}[g_{d_2}] = \sum_{i=1}^{D_1}\sum_{j=1}^{D_2} \alpha_i \beta_j v_{m_i m'_j}[g]. \end{aligned}$$

Thus v_m is in the span of $\mathbb{B}_{d_1,d_2}$ and hence in the span of $\mathbb{B}_j$. This proves the claim.

The PIT algorithm only has to check if $\mathbb{B}_d$ has a nonzero vector. This proves the claim.

Suppose the input nonassociative circuit C computing some degree d polynomial $f \in \mathbb{F}\{X\}$ is inhomogeneous. Then, using Lemma 2 we can first compute in polynomial time homogeneous circuits $C_j \ : \ 0 \le j \le d$, where C_j computes the degree-j homogeneous part f_j. Then we run the above algorithm on each C_j to check whether f is identically zero. This completes the proof of the theorem. □

[3] Here a crucial point is that for a nonassociative monomial of degree d, such a choice for d_1 and d_2 is *unique*. This is a place where a general noncommutative circuit behaves very differently.

4 Polynomial Factorization in $\mathbb{F}\{X\}$

In this section we describe our polynomial-time white-box factorization algorithm for polynomials in $\mathbb{F}\{X\}$. More precisely, given as input a nonassociative circuit C computing a polynomial $f \in \mathbb{F}\{X\}$, the algorithm outputs circuits for all irreducible factors of f. The algorithm uses as subroutine the PIT algorithm for polynomial in $\mathbb{F}\{X\}$ described in Section 3.

To facilitate exposition, we completely describe a deterministic polynomial-time algorithm that computes a nontrivial factorization $f = g \cdot h$ of f, by giving circuits for g and h, unless f is irreducible. We will briefly outline how this extends to finding all irreducible factors efficiently.

We start with a special case.

Lemma 3. *Let $f \in \mathbb{F}\{\mathrm{X}\}$ be a degree d polynomial given by a circuit C of size s such that the constant term in f is zero. Furthermore, suppose there is a factorization $f = g \cdot h$ such that the constant terms in g and h are also zero. Then in deterministic* $\mathrm{poly}(n, d, s)$ *time we can compute the circuits for polynomials g and h.*

Proof. We first consider the even more restricted case when C computes a homogeneous degree d polynomial $f \in \mathbb{F}\{X\}$. For the purpose of computing partial derivatives, it is convenient to transform C into the noncommutative circuit $\tilde{C}$, as explained in Section 2, which computes the fully bracketed polynomial $\tilde{f} \in \mathbb{F}\langle X, (,)\rangle$. Using Theorem 1 we compute a monomial $m = (m_1 m_2)$ where m_1 and m_2 are also fully bracketed. We can transform $\tilde{C}$ to drop the outermost opening and closing brackets. Now, using Lemma 1, we compute the resulting circuits left partial derivative w.r.t. m_1 and right partial derivative w.r.t. m_2. Call these $\tilde{f_1}$ and $\tilde{f_2}$. We can check if $\tilde{f} = (\tilde{f_1}\tilde{f_2})$: we first recover the corresponding nonassociative circuits for f_1 and f_2 from the circuits for $\tilde{f_1}$ and $\tilde{f_2}$. Then we can apply the PIT algorithm of Theorem 1 to check if $f = f_1 f_2$. Clearly, f is irreducible iff $f \neq f_1 f_2$. Continuing thus, we can fully factorize f into its irreducible factors.

Now we prove the actual statement. Applying Lemma 2, we compute homogeneous circuits $C_j : 1 \leq j \leq d$ for the homogeneous degree j component f_j of the polynomial f. Clearly $f_d = g_{d_1} h_{d_2}$. We run the PIT algorithm of Theorem 1 on the circuit C_d to extract a monomial m of degree d along with its coefficient $c_m(f_d)$ in f_d. Notice that the monomial m is of the form $m = (m_1\ m_2)$. If g and h are nontrivial factors of f then m_1 and m_2 are monomials in g and h respectively. Compute the circuits for the left and right derivatives with respect to m_1 and m_2.

$$\frac{\partial^\ell C_d}{\partial m_1} = c_{m_1}(g_{d_1}) \cdot h_{d_2} \quad \text{and} \quad \frac{\partial^r C_d}{\partial m_2} = c_{m_2}(h_{d_2}) \cdot g_{d_1}.$$

In general the $(i + d_2)^{th} : i \leq d - d_2$ homogeneous part of f can be expressed as

$$f_{i+d_2} = g_i h_{d_2} + \sum_{t=i+1}^{i+d_2-1} g_t h_{d_2\ -\ (t-i)}.$$

We depict the circuit C_{i+d_2} for the polynomial f_{i+d_2} in Figure 3. The top gate of the circuit is a $+$ gate. From C_{i+d_2}, we construct another circuit C'_{i+d_2} keeping only those $\times$ gates as children whose left degree is i and right degree is d_2. The resulting circuit is shown in Figure 4. The circuit C'_{i+d_2} must compute $g_i h_{d_2}$. By taking the right partial of C'_{i+d_2} with respect to m_2, we obtain the circuit for $c_{m_2}(h_{d_2})\ g_i$.

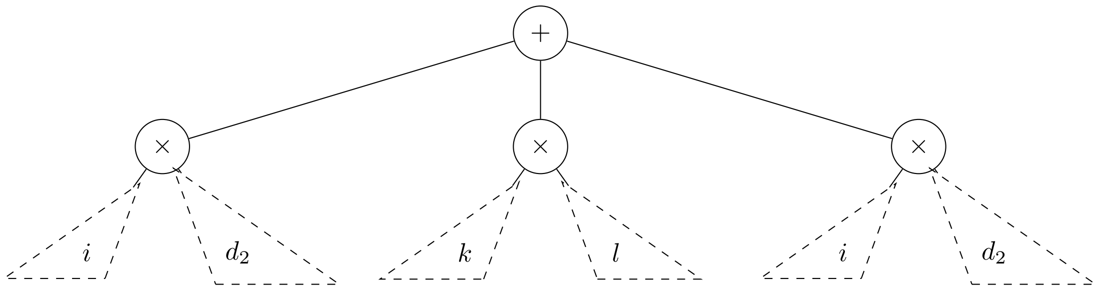


Figure 3: Circuit C_{i+d_2} for f_{i+d_2}

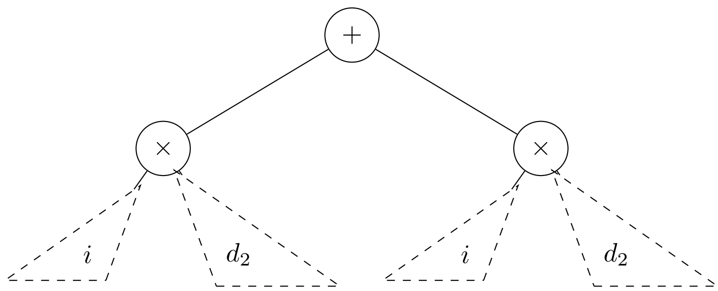


Figure 4: C'_{i+d_2} keeps only degree (i, d_2) type $\times$ gates.

We repeat the above construction for each $i \in [d_1]$ to obtain circuits for $c_{m_2}(h_{d_2})g_i$ for $1 \leq i \leq d_1$. Similarly we can get the circuits for $c_{m_1}(g_{d_1})h_i$ for each $i \in [d_2]$ using the left derivatives with respect to the monomial m_1.

By adding the above circuits we get the circuits C_g and C_h for $c_{m_2}(h_{d_2})g$ and $c_{m_1}(g_{d_1})h$ respectively. We set $C_g = \frac{c_{m_2}(h_{d_2})}{c_m(f)}g$ so that $C_g C_h = f$. Using PIT algorithm one can easily check whether g and h are nontrivial factors. In that case we further recurse on g and h to obtain their irreducible factors. $\square$

Now we consider the general case when f and its factors g, h have arbitrary constant terms. In the subsequent proofs we assume, for convenience, that $\deg(g) \geq \deg(h)$. The case when $\deg(g) < \deg(h)$ can be handled analogously. We first consider the case $\deg(g) = \deg(h)$.

Lemma 4. *For a degree d polynomial $f \in \mathbb{F}\{X\}$ given by a circuit C suppose $f = (g + \alpha)(h + \beta)$, where $g, h \in \mathbb{F}\{X\}$ such that $\deg(g) = \deg(h)$, and $\alpha, \beta \in \mathbb{F}$. Suppose $m = (m_1 m_2)$ is a nonzero degree d monomial. Then, in deterministic polynomial time we can compute circuits for the polynomials $c_{m_1}(g) \cdot h$ and $c_{m_2}(h) \cdot g$, where $c_{m_1}(g)$ and $c_{m_2}(h)$ are coefficient of m_1 and m_2 in g and h respectively.*

Proof. We can write $f = (g + \alpha)(h + \beta) = g \cdot h + \beta \cdot g + \alpha \cdot h + \alpha \cdot \beta$. Applying the PIT algorithm of Theorem 1 on f, we compute a maximum degree monomial $m = (m_1 m_2)$. Computing the left derivative of circuit C w.r.t. monomial m_1, after removing the outermost brackets, we obtain a circuit computing $c_{m_1}(g)h + \beta c_{m_1}(g) + \alpha c_{m_1}(h)$. Dropping the constant term, we obtain a circuit computing polynomial $c_{m_1}(g)h$. Similarly, computing the right derivative w.r.t m_2 yields a circuit for $c_{m_2}(h)g + \beta c_{m_2}(g) + \alpha c_{m_2}(h)$. Removing the constant term we get a circuit for $c_{m_2}(h)g$. $\square$

When $\deg(g) > \deg(h)$ we can recover $h + \beta$ entirely (upto a scalar factor) and we need to obtain the homogeneous parts of g separately.

Lemma 5. *Let $f = (g + \alpha) \cdot (h + \beta)$ be a polynomial of degree d in $\mathbb{F}\{X\}$ given by a circuit C. Suppose $\deg(g) > \deg(h)$. Then, in deterministic polynomial time we can compute the circuit C' for $c_{m_1}(g)(h + \beta)$.*

Proof. Again, applying the PIT algorithm to f we obtain a nonzero degree d monomial $m = (m_1\ m_2)$ of f. If $f = (g + \alpha)(h + \beta)$ then $f = g \cdot h + \alpha h + \beta g + \alpha\beta$. As $\deg(g) > \deg(h)$, the left partial derivative of C with respect to m_1 yields a circuit C' for $c_{m_1}(g)\ (h + \beta)$. □

Extracting the homogeneous components from the circuit C' given by Lemma 5, yields circuits for $\{c_{m_1}(g)h_i\ :\ i \in [d_2]\}$. We also get the constant term $c_{m_1}(g)\beta$. Now we obtain the homogeneous components of g as follows.

Lemma 6. *Suppose circuit C computes f, where $f = (g + \alpha)\ (h + \beta)$ of degree d, $\alpha, \beta \in \mathbb{F}$, $\deg(g) = d_1$ and $\deg(h) = d_2$ such that $d_1 > d_2$.*

- *Let m be a nonzero degree d monomial of f such that $m = (m_1\ m_2)$. Then circuits for $\{c_{m_2}(h)g_i : i \in [d_1 - d_2 + 1, d_1]\}$ can be computed in deterministic polynomial time.*
- *The $(d_2 + i)^{th}$ homogeneous part of f is given by $f_{d_2+i} = \sum_{j=0}^{d_2-1} g_{d_2+i-j}\ h_j + g_i\ h_{d_2}$ for $1 \le i \le d_1 - d_2$. From the circuit C_{d_2+i} of f_{d_2+i}, we can efficiently compute circuits for $\{c_{m_2}(h_{d_2})g_i : 1 \le i \le d_1 - d_2\}$.*

Proof. For the first part, fix any $i \in [d_1 - d_2 + 1, d_1]$, and compute the homogeneous $(i + d_2)^{th}$ part f_{i+d_2} of f by a circuit C_{i+d_2}. Similar to Lemma 3, we focus on the sub-circuits of C_{i+d_2} formed by $\times$ gate of the degree type (i, d_2). Since i is at least $d_1 - d_2 + 1$, such gates can compute the multiplication of a degree i polynomial with a degree d_2 polynomial. Then, by taking the right partial derivative with respect to m_2 we recover the circuits for $c_{m_2}(h_{d_2})\ g_i$ for any $i \in [d_1 - d_2 + 1, d_1]$.

Next, the goal is to recover the circuits for g_i (upto a scalar multiple), where $1 \le i \le d_1 - d_2$, and also recover the constant terms α and β. When $i \le d_1 - d_2$ a product gate of type (i, d_2) can entirely come from g which requires a different handling.

We explain only the case when $i = d_1 - d_2$ (the others are similar). For $i = d_1 - d_2$, we have $f_{d_1} = \beta g_{d_1} + \sum_{j=1}^{d_2-1} g_{d_1-j}\ h_j + g_{d_1-d_2}\ h_{d_2}$. By Lemma 5, we can compute a circuit C' for $c_{m_1}(g)(h + \beta)$. Extracting the constant term yields $c_{m_1}(g)\beta$. From Lemma 6 we have a circuit C'' for $c_{m_2}(h)g_{d_1}$. Multiplying these circuits, we obtain a circuit C^* for $c_{m_2}(h)c_{m_1}(g)\beta g_{d_1}$. Since $c_{m_2}(h)c_{m_1}(g) = c_m(f)$, dividing C^* by $c_m(f)$ yields a circuit for βg_{d_1}. Note that, by the first part of this lemma, we already have circuits for every term g_{d_1-j} appearing in the above sum. Subtracting $\beta g_{d_1} + \sum_{j=1}^{d_2-1} g_{d_1-j}\ h_j$ from the circuit C_{d_1} for f_{d_1}, yields a circuit for polynomial $g_{d_1-d_2}h_{d_2}$. Computing the right derivative of the resulting circuit w.r.t m_2 (Lemma 1) yields a circuit for $c_{m_2}(h)g_{d_1-d_2}$.

For general $i \le d_1 - d_2$, when we need to compute g_i, again we will have already computed circuits for all $g_j, j > i$. A suitable right derivative computation will yield a circuit for $c_{m_2}(h)g_i$. □

Lemmas 3, 4, 5, and 6 yield an efficient algorithm for computing circuits for the two factors $c_{m_2}(h)(\sum_{i=1}^{d_1} g_i)$ and $c_{m_1}(g)(\sum_{i=1}^{d_2} h_i)$ when $\deg(g) \ge \deg(h)$. The case when $\deg(g) < \deg(h)$ is similarly handled using left partial derivatives in the above lemmas.

Now we explain how to compute the constant terms of the individual factors. We discuss the case when $\alpha \neq 0$. The other case is similar.

First we recall that given a monomial m and a noncommutative circuit C, the coefficient of m in C can be computed in deterministic polynomial time [AMS10]. We know that $f_0 = \alpha \cdot \beta$. We compute the coefficient of the monomial m_1 in the circuits for polynomials $c_{m_2}(h)c_{m_1}(g)gh$, $c_{m_2}(h)g$, and $c_{m_1}(g)h$. Let these coefficients be a, b and c, respectively. Moreover, we know that $c_{m_2}(h)c_{m_1}(g)$ is the coefficient of monomial $m = (m_1\ m_2)$ in f. Let the coefficient of m_1 in f be γ. Let $\gamma_1 = c_{m_1}(g)$ and $\gamma_2 = c_{m_2}(h)$ and $\delta = c_{m_1}(g)c_{m_2}(h)$.

Now equating the coefficient of m_1 from both side of the equation $f = (g + \alpha)(h + \beta)$ and substituting $\beta = \frac{f_0}{\alpha}$, we get

$$\gamma = \frac{a}{\gamma_1\gamma_2} + \frac{\alpha c}{\gamma_1} + \frac{f_0 b}{\alpha\gamma_2} = \frac{a}{\delta} + \frac{\alpha c}{\gamma_1} + \frac{f_0 b}{\alpha\gamma_2}.$$

Letting $\xi = \alpha\gamma_2$, this gives a quadratic equation in the unknown ξ.

$$c\xi^2 + (a - \gamma\delta)\xi + f_0 b\delta = 0.$$

By solving the above quadratic equation we get two solutions A_1 and A_2 for $\xi = \alpha\gamma_2$. Notice that $\beta\gamma_1 = \frac{\delta f_0}{\xi}$. As we have circuits for $c_{m_2}(h)g = \gamma_2 g$ and for $c_{m_1}(g)h = \gamma_1 h$, we obtain circuits for $\gamma_2(g+\alpha)$ and $\gamma_1(h+\beta)$ (two solutions, corresponding to A_1 and A_2). To pick the right solution, we can run the PIT algorithm to check if $\gamma_1\gamma_2 f$ equals the product of these two circuits that purportedly compute $\gamma_2(g + \alpha)$ and $\gamma_1(h + \beta)$.

Over $\mathbb{Q}$ we can just solve the quadratic equation in deterministic polynomial time using standard method. If $\mathbb{F} = \mathbb{F}_q$ for $q = p^r$, we can factorize the quadratic equation in deterministic time $\text{poly}(p, r)$ [vzGS92]. Using randomness, one can solve this problem in time $\text{poly}(\log p, r)$ using Berlekamp's factoring algorithm [Ber71]. This also completes the proof of the following.

Theorem 2. *Let $f \in \mathbb{F}\{X\}$ be a degree d polynomial given by a circuit of size s. If $\mathbb{F} = \mathbb{Q}$, in deterministic $\text{poly}(s, n, d)$ time we can compute a nontrivial factorization of f or reports f is irreducible. If $\mathbb{F}$ is a finite field such that char$(\mathbb{F}) = p$, we obtain a deterministic $\text{poly}(s, n, d, p)$ time algorithm that computes a nontrivial factorization of f or reports f is irreducible.*

Finally, we state the main result of this paper.

Theorem 3. *Let $f \in \mathbb{F}\{X\}$ be a degree d polynomial given by a circuit of size s. Then if $\mathbb{F} = \mathbb{Q}$, in deterministic $\text{poly}(s, n, d)$ time we can output the circuits for the irreducible factors of f. If $\mathbb{F}$ is a finite field such that char$(\mathbb{F}) = p$, we obtain a deterministic $\text{poly}(s, n, d, p)$ time algorithm for computing circuits for the irreducible factors of f.*

Remark 3. *We could apply Theorem 2 repeatedly to find all irreducible factors of the input $f \in \mathbb{F}\{X\}$. However, the problem with that approach is that the circuits for g and h we computed in the proof of Theorem 2, where $f = gh$ is the factorization, is larger than the input circuit C for f by a polynomial factor. Thus, repeated application would incur a superpolynomial blow-up in circuit size. We can avoid that by computing the required partial derivative of g as a suitable partial derivative of the circuit C directly. This will keep the circuits polynomially bounded. This idea is from [AJR15] where it is used for homogeneous noncommutative polynomial factorization. Combined with Theorem 2 this gives the polynomial-time algorithm of Theorem 3.*

5 Conclusion

Motivated by the nonassociative circuit lower bound result shown in [HWY10], we study PIT and polynomial factorization in the free nonassociative noncommutative ring $\mathbb{F}\{X\}$ and obtain efficient white-box algorithms for the problems.

Hrubes, Wigderson, and Yehudayoff [HWY10] have also shown exponential circuit-size lower bounds for nonassociative, commutative circuits. It would be interesting to obtain an efficient polynomial identity testing algorithm for that circuit model too. Even a randomized polynomial-time algorithm is not known.

Obtaining an efficient *black-box* PIT in the ring $\mathbb{F}\{X\}$ is also an interesting problem. Of course, for such an algorithm the black-box can be evaluated on a suitable nonassociative algebra. To the best of our knowledge, there seems to be no algorithmically useful analogue of the Amitsur-Levitzki theorem [AL50].

References

[AJR15] Vikraman Arvind, Pushkar S. Joglekar, and Gaurav Rattan, *On the complexity of noncommutative polynomial factorization*, Mathematical Foundations of Computer Science 2015 - 40th International Symposium, MFCS 2015, Milan, Italy, August 24-28, 2015, Proceedings, Part II, 2015, pp. 38–49.

[AL50] Avraham Shimshon Amitsur and Jacob Levitzki, *Minimal identities for algebras*, Proceedings of the American Mathematical Society (1950), no. 4, 449–463.

[AMS10] Vikraman Arvind, Partha Mukhopadhyay, and Srikanth Srinivasan, *New results on noncommutative and commutative polynomial identity testing*, Computational Complexity **19** (2010), no. 4, 521–558.

[AR16] Vikraman Arvind and S. Raja, *Some lower bound results for set-multilinear arithmetic computations*, Chicago J. Theor. Comput. Sci. **2016 (6)** (2016).

[Ber71] E. R. Berlekamp, *Factoring polynomials over large finite fields**, Proceedings of the Second ACM Symposium on Symbolic and Algebraic Manipulation (New York, NY, USA), SYMSAC '71, ACM, 1971, pp. 223–.

[Coh85] P. M. Cohn, *Free rings and their relations*, London Mathematical Society Monograph No. 19, 1985.

[HWY10] Pavel Hrubes, Avi Wigderson, and Amir Yehudayoff, *Relationless completeness and separations*, Proceedings of the 25th Annual IEEE Conference on Computational Complexity, CCC 2010, Cambridge, Massachusetts, June 9-12, 2010, 2010, pp. 280–290.

[Hya77] Laurent Hyafil, *The power of commutativity*, 18th Annual Symposium on Foundations of Computer Science (FOCS), Providence, Rhode Island, USA, 31 October - 1 November 1977, 1977, pp. 171–174.

[Kal] Erich Kaltofen, *Factorization of polynomials given by straight-line programs*, Randomness in Computation **vol. 5 of Advances in Computing Research, pages 375–412. 1989**.

[KSS15] Swastik Kopparty, Shubhangi Saraf, and Amir Shpilka, *Equivalence of polynomial identity testing and polynomial factorization*, Computational Complexity **24** (2015), no. 2, 295–331.

[LMP16] Guillaume Lagarde, Guillaume Malod, and Sylvain Perifel, *Non-commutative computations: lower bounds and polynomial identity testing*, Electronic Colloquium on Computational Complexity (ECCC) **23** (2016), 94.

[Nis91] Noam Nisan, *Lower bounds for non-commutative computation (extended abstract)*, STOC, 1991, pp. 410–418.

[RS05] Ran Raz and Amir Shpilka, *Deterministic polynomial identity testing in non-commutative models*, Computational Complexity **14** (2005), no. 1, 1–19.

[SY10] Amir Shpilka and Amir Yehudayoff, *Arithmetic circuits: A survey of recent results and open questions*, Foundations and Trends in Theoretical Computer Science **5** (2010), no. 3-4, 207–388.

[vzGS92] Joachim von zur Gathen and Victor Shoup, *Computing frobenius maps and factoring polynomials*, Computational Complexity **2** (1992), 187–224.

ECCC ISSN 1433-8092

https://eccc.weizmann.ac.il