Electronic Colloquium on Computational Complexity, Report No. 84 (2024)

A Multivariate to Bivariate Reduction for Noncommutative Rank and Related Results

V. Arvind[*] Pushkar S Joglekar[†]

Abstract

We study the *noncommutative rank* problem, ncRANK, of computing the rank of matrices with linear entries in n noncommuting variables and the problem of *noncommutative Rational Identity Testing*, RIT, which is to decide if a given rational formula in n noncommuting variables is zero on its domain of definition.

Motivated by the question whether these problems have *deterministic* NC algorithms, we revisit their interrelationship from a parallel complexity point of view. We show the following results:

1. Based on Cohn's embedding theorem [11, 13] we show deterministic NC reductions from multivariate ncRANK to bivariate ncRANK and from multivariate RIT to bivariate RIT.
2. We obtain a deterministic NC-Turing reduction from bivariate RIT to bivariate ncRANK, thereby proving that a deterministic NC algorithm for bivariate ncRANK would imply that both multivariate RIT and multivariate ncRANK are in deterministic NC.

1 Introduction

There are two main algorithmic problems of interest in this paper. These are the *noncommutative Rational Identity Testing problem* (RIT) and the *noncommutative rank* (ncRANK) problem for matrices with linear entries.

The RIT problem is a generalization of multivariate polynomial identity testing to identity testing of multivariate rational expressions. When the variables are commuting, rational identity testing and polynomial identity testing are equivalent problems. On the other hand, if the variables are all noncommuting, the RIT problem needs different algorithmic techniques as rational expressions in noncommuting variables are more complicated. Mathematically, rational expressions over noncommuting variables are quite well studied. They arise in the construction of

[*]The Institute of Mathematical Sciences (HBNI), Chennai, India, `email: arvind@imsc.res.in` and Chennai Mathematical Institute, Siruseri, Kelambakkam, India

[†]Vishwakarma Institute of Technology, Pune, India, `email: joglekar.pushkar@gmail.com`

ISSN 1433-8092

the so-called free skew fields [13]. Hrubes and Wigderson [19] initiated the algorithmic study of RIT for *rational formulas* and gave a deterministic polynomial time reduction from RIT to ncRANK. Subsequently, deterministic polynomial-time algorithms were obtained independently by Ivanyos et al [21, 20] and by Garg et al [16, 17] for the RIT problem, in fact they obtain deterministic polynomial time algorithms for ncRANK, and using Hrubes-Wigderson reduction from RIT to ncRANK get a polynomial time algorithm for RIT. The Ivanyos et al algorithm is algebraic and works for fields of all characteristics. The Garg et al algorithm has an analytic flavor and is for the characteristic zero case.

The Edmonds' Problem and ncRANK The ncRANK problem is essentially the noncommutative version of the well-known Edmonds' problem: determine the rank of a matrix M whose entries are linear forms in commuting variables (see [21, 17, 4, 7] for more details). A special case of it is to determine if a square matrix M with linear entries in commuting variables is *singular*. This is also known as the symbolic determinant identity testing problem, SDIT. There is an easy randomized NC algorithm for it, based on the Polynomial Identity Lemma [2, 26, 28, 14], by randomly substituting scalar values for the variables from the field (or a suitable extension of it) and evaluating the determinant using a standard NC algorithm. However, a deterministic polynomial-time algorithm for SDIT is an outstanding open problem [4].

1.1 This paper: overview of results and proofs

With this background, the natural algorithmic questions are whether RIT for noncommutative rational formulas and ncRANK have deterministic NC algorithms. We revisit the problems from this perspective and obtain the following new results.

1. We show that multivariate RIT for formulas is deterministic NC reducible to bivariate RIT for formulas. More precisely, given a rational formula $\Phi(x_1, x_2, \ldots, x_n)$, computing an element of the skew field $\mathbb{F}\langle\!\langle X \rangle\!\rangle$, where $X = \{x_1, x_2, \ldots, x_n\}$, the deterministic NC reduction replaces each x_i by a formula $\Phi_i(x, y)$ computing a polynomial in $\mathbb{F}\langle x, y\rangle$. Then the resulting rational formula
$$\Psi(x, y) = \Phi(\Phi_1(x, y), \Phi_2(x, y), \ldots, \Phi_n(x, y))$$
has the property that
$$\Phi(x_1, x_2, \ldots, x_n) \neq 0 \text{ iff } \Psi(x, y) \neq 0.$$

2. We next show that multivariate ncRANK is deterministic NC reducible to bivariate ncRANK. More precisely, given a $d \times d$ linear matrix $A = A_0 + \sum_{i=1}^{n} A_i x_i$ in noncommuting variables $X = \{x_1, x_2, \ldots, x_n\}$, where the A_i are

matrices over the scalar field $\mathbb{F}$, we first give a deterministic NC reduction that transforms A to a $d \times d$ matrix B whose entries are bivariate polynomials in $\mathbb{F}\langle x, y\rangle$, where x and y are two noncommuting variables, where its entries $B[i, j]$ are given by *polynomial size noncommutative formulas*, with the property that $\text{ncrk}(A) = \text{ncrk}(B)$. Then we examine the Higman linearization process [19] that transforms B into a matrix B' with linear entries in x and y such that the noncommutative rank $\text{ncrk}(B)$ of B can be easily recovered from $\text{ncrk}(B')$. We show that this process can be implemented in deterministic NC (the earlier works [19, 20, 21, 17] only consider its polynomial-time computability).

Additionally, we consider the more general problem ncRANK_{poly} of computing the noncommutative rank of a matrix whose entries are noncommutative formulas computing polynomials. We show using our parallel Higman linearization algorithm that ncRANK_{poly} is also deterministic NC reducible to bivariate ncRANK.

Both the multivariate to bivariate reductions, stated above, are crucially based on a theorem of Cohn [11] (also see [13, Theorem 4.7.3]) which we will refer to as Cohn's embedding theorem and describe it later in the introduction.

3. Finally, obtaining a deterministic NC reduction from RIT to ncRANK turns out to be quite subtle. From the work of Hrubes and Wigderson [19], who initiated this line of research on RIT, we can only obtain a sequential deterministic polynomial-time reduction from RIT to ncRANK. However, for our result we require an NC reduction. If the given rational formula has logarithmic depth, then their result already implies an NC reduction.

   Now, in the same paper [19], Hrubes and Wigderson have also shown a *depth reduction* result for multivariate noncommutative rational formulas: every rational formulas of size s is equivalent to a logarithmic depth rational formula of size $\text{poly}(s)$. Their construction is based on Brent's depth reduction result for commutative arithmetic formulas. However, due to noncommutativity and the presence of inversion gates, the formula constructed in their proof needs to be different based on whether certain rational subformulas, arising in the construction procedure, are identically zero or not. To algorithmize such steps in the construction we need to use RIT as a subroutine. As RIT has a polynomial-time algorithm [21, 17], the depth-reduction in [19] also has a polynomial time algorithm.[1]

   As the third result of this paper, building on the Hrubes-Wigderson depth-reduction construction, we are able to show that, with oracle access to RIT, rational formula depth reduction can be done in deterministic NC. Using

[1]In the commutative case, Brent's result is parallelizable to yield an NC algorithm. For noncommutative formulas without inversion gates we can obtain the depth-reduced formula in NC, as we will observe later in the paper.

this we are able to obtain a deterministic NC-Turing reduction from RIT to ncRANK. Hence, if bivariate ncRANK is in deterministic NC we will obtain a deterministic NC algorithm also for RIT. We leave open the question whether depth reduction of noncommutative rational formulas is unconditionally in NC.

We now outline the proof ideas in more detail with some intuitive explanations.

2 Preliminaries

In this section we recall the essential basic definitions and fix the notation.

Let $\mathbb{F}$ be a (commutative) field[2] and $X = \{x_1, x_2, \ldots, x_n\}$ be n free noncommuting variables. The free monoid X^* is the set of all monomials in the variables X. A *noncommutative polynomial* $f(X)$ is a finite $\mathbb{F}$-linear combination of monomials in X^*, and the *free noncommutative ring* $\mathbb{F}\langle X\rangle$ consists of all noncommutative polynomials.

Noncommutative Rational Formulas An *arithmetic circuit* computing an element of $\mathbb{F}\langle X\rangle$ is a directed acyclic graph with each indegree 0 node labeled by either an input variable $x_i \in X$ or some scalar $c \in \mathbb{F}$. Each internal node g has indegree 2 and is either a $+$ gate or a $\times$ gate: it computes the sum (resp. left to right product) of its inputs. Thus, each gate of the circuit computes a polynomial in $\mathbb{F}\langle X\rangle$ and the polynomial computed by the circuit is the polynomial computed at the *output gate*. A *formula* is restricted to have fanout 1 or 0.

When we allow the formulas/circuits to have *inversion gates* we get *rational formulas and rational circuits*.

The Free Skew Field We now briefly explain the free skew field construction. The elements of the free skew field are noncommutative rational functions which are more complicated than their commutative counterparts. Rational formulas in the commutative setting can be canonically expressed as ratios of two polynomials. There is no such canonical representation for noncommutative rational formulas.

Following Hrubes-Wigderson [19], we use Amitsur's approach [1] for formally defining skew fields.[3]

It involves defining appropriate notion of equivalence of formulas (intuitively, two formulas are equivalent if they agree on their *domain of definition*). The equivalence classes under this equivalence relation are the elements of the free skew field. We give the formal definitions below.

Let $\mathbb{M}_k(\mathbb{F})$ denote the ring of $k \times k$ matrices with entries from field $\mathbb{F}$. Note that a rational formula Φ defines a partial function

$$\hat{\Phi} : \mathbb{M}_k(\mathbb{F})^n \mapsto \mathbb{M}_k(\mathbb{F})$$

[2]In this paper, $\mathbb{F}$ will either be the field of rationals or a finite field.

[3]There are other ways to defining free skew fields [1, 3, 25, 8, 9, 10, 13].

that on input $(a_1, a_2, \dots, a_n) \in \mathbb{M}_k(\mathbb{F})^n$ evaluates Φ by substituting $x_i \leftarrow a_i$ for $i \in [n]$. $\hat{\Phi}(a_1, \dots, a_n)$ is undefined if the input to some inversion gate in Φ is not invertible in $\mathbb{M}_k(\mathbb{F})$.

Definition 2.1. *Let Φ be a rational formula in variables X. For each $k \in \mathbb{N}$, let $\mathcal{D}_{k,\Phi}$ be the set of all matrix tuples $(a_1, a_2, \dots, a_n) \in \mathbb{M}_k(\mathbb{F})^n$ such that $\hat{\Phi}(a_1, a_2, \dots, a_n)$ is defined. The* domain of definition *of Φ is the union $\mathcal{D}_\Phi = \bigcup_k \mathcal{D}_{k,\Phi}$.*

Definition 2.2. [19]

- *A rational formula Φ is called* correct *if for every gate u of Φ the subformula Φ_u has a nonempty domain of definition.*
- *Correct rational formulas Φ_1, Φ_2 are said to be* equivalent *(denoted $\Phi_1 \equiv \Phi_2$) if the intersection $\mathcal{D}_{\Phi_1} \cap \mathcal{D}_{\Phi_2}$ of their domains of definitions is nonempty and they agree on all the points in the intersection.*

We note that equivalent formulas need not have the same domain of definition. For example, $\Phi_1 = z_1z_2z_3$ and $\Phi_2 = (z_1z_2z_3 \cdot (z_2z_3 - z_3z_2)^{-1}) \cdot (z_2z_3 - z_3z_2)$ are equivalent. However, the domain of definition of Φ_1 includes all matrix tuples, whereas the domain of definition of Φ_2 contains only matrix tuples (Z_1, Z_2, Z_3) such that $Det(Z_2Z_3 - Z_3Z_2) \neq 0$.

The relation $\equiv$ as defined above is an equivalence relation on rational formulas and the equivalence classes, called *rational functions*, are the elements of the skew field $\mathbb{F}\langle\!\langle X \rangle\!\rangle$ [19, 1].

Noncommutative Rank We now recall the notion of rank for matrices over the noncommutative ring $\mathbb{F}\langle X \rangle$.

Definition 2.3 (inner rank)**.** *Let M be a matrix over $\mathbb{F}\langle X \rangle$. Its* inner rank *is the least r such that M can be written as a matrix product $M = PQ$ where Q has r rows (and P has r columns).*

Definition 2.4 (full matrices)**.** *An $n \times n$ square matrix M over $\mathbb{F}\langle X \rangle$ is* full *if it cannot be decomposed as a matrix product $M = PQ$ where P is $n \times r$ and Q is $r \times n$ for $r < n$. In other words, an $n \times n$ matrix is called full precisely when its inner rank is n.*

We can also define the rank of a matrix M to be the maximum r such that M contains an $r \times r$ full submatrix. For matrices over $\mathbb{F}\langle X \rangle$ these notions of *noncommutative rank* coincide as summarized below.[4]

Proposition 2.5. [13]

Let M be an $n \times n$ matrix over the ring $\mathbb{F}\langle X \rangle$. Then

- *M is a full matrix (that is, M has inner rank n) iff it is invertible over the skew field $\mathbb{F}\langle\!\langle X \rangle\!\rangle$.*
- *More generally, M has inner rank r iff the largest full submatrix of M is $r \times r$.*

[4]For a ring R in general, a full matrix R need not be invertible (see [19] for an example).

The Algorithmic Problems of Interest At this point we formally define the problems of interest in this paper.

1. The multivariate RIT problem takes as input a rational formula Φ, computing a rational function $\hat{\Phi}$ in $\mathbb{F}\langle\!\langle X \rangle\!\rangle$, and the problem is to check if Φ is equivalent to 0? In the bivariate RIT problem Φ computes a rational function in $\mathbb{F}\langle\!\langle x, y \rangle\!\rangle$.

2. The multivariate ncRANK problem takes as input a matrix M with affine linear form entries over X and the problem is to determine its noncommutative rank $\text{ncrk}(M)$. Bivariate ncRANK is similarly defined.

3. A more general version of ncRANK is ncRANK$_{poly}$ in which the matrix entries are allowed to be polynomials in $\mathbb{F}\langle X \rangle$ computed by noncommutative formulas. A closely related problem is SINGULAR where the problem is to test if a square matrix M over $\mathbb{F}\langle X \rangle$ with entries computed by formulas is singular or not.

The complexity class NC and NC reductions The class NC consists of decision problems that can be solved in polylog(n) time with poly(n) many processors.[5] For two decision problems A and B we say A is many-one NC reducible to B if there is a reduction from A to B that is NC computable. Similarly, A is NC-Turing reducible to B if there is an oracle NC algorithm for A that has oracle access to B.

It turns out that SINGULAR and ncRANK are equivalent even under deterministic NC reductions. [6]

Cohn's Embedding Theorem We now give an outline of Cohn's embedding theorem and how it gives us the desired reduction from multivariate to bivariate RIT and also from multivariate to bivariate ncRANK. However, for multivariate to bivariate reduction for ncRANK we will require additional NC algorithms for formula depth reduction and Higman linearization.

Let $X = \{x_1, x_2, \ldots, x_n\}$ be a set of n noncommuting variables, and let x, y be a pair of noncommuting variables. We first recall the following well-known fact, observed in the early papers on noncommutative polynomial identity testing [5, 24]: for noncommutative polynomials in $\mathbb{F}\langle X \rangle$, the problem of polynomial identity testing (PIT) is easily reducible to PIT for bivariate noncommutative polynomials in $\mathbb{F}\langle x, y \rangle$. Indeed, more formally, we have the following easy to check fact.

Fact 2.6. *The map*

$$x_i \mapsto x^{i-1}y, 1 \leq i \leq n$$

[5]This model is widely accepted as the right theoretical notion for efficient parallel algorithms.

[6]As for $M \in \mathbb{F}\langle X \rangle^{m \times n}$, $\text{ncrk}(M) = r$ iff r is a size of a largest sized invertible minor of M, so to compute $\text{ncrk}(M)$, it suffices to test singularity of matrix UMV, where U, V are generic $r \times m, n \times r$ matrices respectively with entries as fresh noncommuting variables for $r \leq \min(m, n)$. See e.g. [17, Lemma A.3] for details.

extends to an injective homomorphism (i.e. a homomorphic embedding*) from the ring* $\mathbb{F}\langle X\rangle$ *to the ring* $\mathbb{F}\langle x,y\rangle$.

However, in order to obtain our multivariate to bivariate reductions, we need a mapping $\beta : X \to \mathbb{F}\langle x,y\rangle$ which has the following properties:

- For each i, there is a small noncommutative arithmetic formula that computes $\beta(x_i)$.
- β extends to an injective homomorphism[7], not just from the ring $\mathbb{F}\langle X\rangle$ to $\mathbb{F}\langle x,y\rangle$, but also to an injective homomorphism from the skew field $\mathbb{F}\langle\!\langle X\rangle\!\rangle$ to the skew field $\mathbb{F}\langle\!\langle x,y\rangle\!\rangle$. This will guarantee that for two rational formulas Φ_1, Φ_2 computing inequivalent rational functions in $\mathbb{F}\langle\!\langle X\rangle\!\rangle$ their images $\beta(\Phi_1)$ and $\beta(\Phi_2)$ also compute inequivalent rational functions in $\mathbb{F}\langle x,y\rangle$.
- Furthermore, in order to get the multivariate to bivariate reduction for ncRANK, we will additionally require of the map β that for any matrix M over $\mathbb{F}\langle X\rangle$ its image $\beta(M)$, which is a matrix over $\mathbb{F}\langle x,y\rangle$ obtained by applying β to each entry of M, has the same rank as M. Such a homomorphic embedding is called an *honest embedding* [11]. Here we note that, full matrices over $\mathbb{F}\langle X\rangle$ are invertible over $\mathbb{F}\langle\!\langle X\rangle\!\rangle$ [13]. Consequently if one can lift embedding β to one between the corresponding free skew fields, it enforces β to be an honest embedding.

The mapping $x_i \mapsto x^{i-1}y$ actually *does not* extend to an honest embedding as observed in [11]. Indeed, the rank 2 matrix $\begin{pmatrix} x_1 & x_2 \\ x_3 & x_4 \end{pmatrix}$ has image

$$\begin{pmatrix} y & xy \\ x^2y & x^3y \end{pmatrix} = \begin{pmatrix} 1 \\ x^2 \end{pmatrix} \begin{pmatrix} y & xy \end{pmatrix}$$

which is rank 1. In general, a homomorphic embedding from a ring R to a ring R' is an *honest embedding* if it maps full matrices over R to full matrices over R'. We now state Cohn's embedding theorem.

For polynomials $f, g \in \mathbb{F}\langle x,y\rangle$ let $[f,g]$ denotes the commutator polynomial $fg - gf$. Cohn's embedding map $\beta : \mathbb{F}\langle X\rangle \to \mathbb{F}\langle x,y\rangle$ is defined as follows.

- Let $\beta(x_1) = y$. For $i \geq 2$, define $\beta(x_i) = [\beta(x_{i-1}), x]$.
- We can then naturally extend β to a homomorphism from $\mathbb{F}\langle X\rangle$ to $\mathbb{F}\langle y,x\rangle$, and it is easy to check that it is injective. In fact, we can even assume $|X|$ to be countably infinite.

Theorem 2.7 (Cohn's embedding theorem)**.** [13, Theorem 7.5.19] *The embedding map* $\beta : \mathbb{F}\langle X\rangle \to \mathbb{F}\langle x,y\rangle$ *defined above extends to an embedding between the corresponding skew fields* $\beta : \mathbb{F}\langle\!\langle Z\rangle\!\rangle \to \mathbb{F}\langle\!\langle x,y\rangle\!\rangle$ *and hence is an honest embedding.*

[7]That is, a homomorphic embedding.

Cohn's construction is based on skew polynomial rings, which explains the appearance of the iterated commutators $\beta(x_i) = [\beta(x_{i-1}), x]$. We briefly sketch the underlying ideas in the appendix (see Section 6). For more details see [11, 13].

3 The Reduction from multivariate RIT to bivariate RIT

The reduction follows quite easily from Theorem 2.7. However, we present some complexity details in this section showing that it is actually a deterministic NC reduction. The following lemma is useful to describe the reduction.

Lemma 3.1. *Recall the embedding map β defined above. $\beta(z_0) = y$ and $\beta(z_{i+1}) = [\beta(z_i), x]$ are polynomials in $\mathbb{F}\langle x, y\rangle$ for each $i \geq 0$. Then, for $n \geq 1$ we have*

$$\beta(z_n) = \sum_{i=0}^{n} (-1)^i \binom{n}{i} x^i y x^{n-i}.$$

As a consequence, there is a deterministic NC *algorithm that constructs a* poly(n)*-sized formula for $\beta(z_n)$.*

Proof. We will use induction on n. The base case follows from the fact that $\beta(z_1) = yx - xy$. Inductively assume the claim is true for all n. Now, $\beta(z_{n+1}) = [\beta(z_n), x]$

$$\begin{aligned}
&= \sum_{i=0}^{n} (-1)^i \binom{n}{i} x^i y x^{n-i+1} - \sum_{j=0}^{n} (-1)^j \binom{n}{j} x^{j+1} y x^{n-j} \text{ by induction hypothesis} \\
&= yx^{n+1} + \sum_{i=1}^{n} (-1)^i \binom{n}{i} x^i y x^{n-i+1} + \sum_{i=1}^{n+1} (-1)^i \binom{n+1-1}{i-1} x^i y x^{n+1-i} \\
&= yx^{n+1} + \sum_{i=1}^{n} (-1)^i \left[\binom{n+1-1}{i} + \binom{n+1-1}{i-1} \right] x^i y x^{n-i+1} + (-1)^{n+1} x^{n+1} y \\
&= yx^{n+1} + (-1)^{n+1} x^{n+1} y + \sum_{i=1}^{n} (-1)^i \binom{n+1}{i} x^i y x^{n+1-i} \text{ from Pascal's identity} \\
&= \sum_{i=0}^{n+1} (-1)^i \binom{n+1}{i} x^i y x^{n+1-i}
\end{aligned}$$

This completes the inductive proof.

As the binomial coefficients can be computed in NC using Pascal's identity, the expression for $\beta(z_n)$ obtained above immediately implies an NC algorithm for construction of a poly(n) sized formula for $\beta(z_n)$. □

Theorem 3.2. *The multivariate RIT problem is deterministic* NC *(in fact, logspace) reducible to bivariate RIT. More precisely, given as input a rational formula Φ computing an element of $\mathbb{F}\langle\!\langle X \rangle\!\rangle$, $X = \{x_1, x_2, \ldots, x_n\}$ there is a deterministic* NC *algorithm that computes a rational formula Ψ computing an element of $\mathbb{F}\langle\!\langle x, y \rangle\!\rangle$ such that Φ is nonzero in its domain of definition iff Ψ is nonzero in its domain of definition.*

Proof. We can identify $\mathbb{F}\langle\!\langle X \rangle\!\rangle$ with $\mathbb{F}\langle\!\langle z_0, z_1, \ldots, z_{n-1} \rangle\!\rangle$. Let $\Phi_i(x, y)$ be the poly(i) size noncommutative formula computing the nested commutator $\beta(z_i)$ for each i. In the rational formula Φ, for each i we replace the input z_i to Φ by $\Phi_i(x, y)$. The new formula we obtain is

$$\Psi(x, y) = \Phi(\Phi_1(x, y), \Phi_2(x, y), \ldots, \Phi_{n-1}(x, y)).$$

By Theorem 2.7, $\Psi(x, y) \neq 0$ on its domain of definition iff $\Phi(z_0, z_1, \ldots, z_{n-1})$ is nonzero on its domain of definition. Furthermore, because β is an embedding, it is guaranteed that if Φ has a nontrivial domain of definition then Ψ also has a nontrivial domain of definition.

As computation of Ψ from Φ involves only replacing the z_i by Φ_i, the reduction is clearly logspace computable. □

4 Reduction from n-variate ncRANK$_{poly}$ to 2-variate ncRANK

In this section we give a deterministic NC reduction from n-variate ncRANK$_{poly}$ to bivariate ncRANK. The basic idea of the reduction is as follows. Given a polynomial matrix[8] $M \in \mathbb{F}\langle X \rangle^{m\times m}$ such that each entry of M is computed by formula of size at most s. We will use Cohn's embedding theorem 2.7 to get a matrix M_1 with *bivariate* polynomial entries such that each entry of M_1 is computed by a poly(n, s) size noncommutative formula and ncrk(M) = ncrk(M_1). Notice that M_1 is an instance of bivariate ncRANK$_{poly}$. Next, we need to give an NC reduction transforming M_1 to an instance of bivariate ncRANK (which will be a matrix with linear entries in x and y).

In order to do this transformation in NC, we will first apply the depth-reduction algorithm of Lemma 4.2 to get matrix M_2 whose entries are poly(n, s) size log-depth formulas that compute the same polynomials as the corresponding entries of M_1. Then we apply Higman Linearization to M_2 to obtain a bivariate linear matrix M_3. For this we will use our parallel algorithm for Higman Linearization described in Theorem 4.4. From the properties of Higman linearization we can easily recover ncrk(M_2) from ncrk(M_3). In what follows, first we give a deterministic NC algorithm for the depth reduction of noncommutative formulas and Higman linearization process. We conclude the section by giving an NC reduction from multivariate to bivariate ncRANK using above NC algorithms combined with Cohn's embedding theorem 2.7.

[8]We can assume it is a square matrix without loss of generality.

4.1 Depth reduction for noncommutative formulas without divisions

In the commutative setting Brent [6] obtained a deterministic NC algorithm to transform a given *rational* formula (which may have division gates) to a log-depth rational formula. In the noncommutative setting, Hrubes and Wigderson [19] proved the *existence* of log-depth *rational* formula equivalent to any given rational formula. Their proof is based on [6]. However, it is not directly algorithmic as explained in the introduction. We will discuss it in more detail in Section 5. However, it turns out that, if the noncommutative formula doesn't have division gates then the depth reduction is quite easy and we obtain a simple deterministic NC algorithm for it that computes a log-depth noncommutative formula equivalent to the given noncommutative formula. The proof is based Brent's commutative version. We just highlight the distinctive points arising in the noncommutative version in the proof presented in the appendix.

We introducing some notation. Let Φ be a noncommutative arithmetic formula computing a polynomial in $\mathbb{F}\langle x_1, x_2, \ldots, x_n\rangle$. Let $\hat{\Phi}$ denote the polynomial computed by Φ. For a node $v \in \Phi$, let Φ_v denote the subformula of Φ rooted at node v, so $\hat{\Phi_v}$ is the polynomial computed by the subformula rooted at v. For a node $v \in \Phi$, let $\Phi_{v\leftarrow z}$ be a formula obtained from Φ by replacing the sub-formula Φ_v by single variable z. For a node $v \in \Phi$, let $wt(v) = |\Phi_v|$ denote the number of nodes in the subformula rooted at v. By size of formula Φ we refer to number of gates in Φ.

Lemma 4.1. *Given a formula Φ of size s computing a noncommutative polynomial $f \in \mathbb{F}\langle x_1, x_2, \ldots, x_n\rangle$ there is an* NC *algorithm to obtain an equivalent formula Φ' for f with depth $O(\log s)$.*

The proof of the lemma is in the appendix.

Remark 4.2. *In Lemma , we avoid using the depth-reduction approach for noncommutative rational formulas in [19]. This is because it can introduce inversion gates even if the original formula has no inversion gates. Instead, we directly adapt ideas from Brent's construction for commutative formulas to obtain the NC algorithm. We note here that Nisan's seminal work [22] also briefly mentions noncommutative formula depth reduction (but not its parallel complexity or even in an algorithmic context).*

Higman linearization which is sometimes called Higman's trick was first used by Higman in [18]. Cohn extensively used Higman Linearization in his factorization theory of free ideal rings. Given a matrix with noncommutative polynomials as its entries, Higman linearization process transforms it into a matrix with linear entries. This transformation process has several nice properties such as: it preserves fullness of the matrix (that is the input polynomial matrix is full rank iff final linear matrix is full rank), it preserves irreducibility of the matrix, etc.

We first describe a single step of the linearization process applied to a single noncommutative polynomial, which easily generalizes to matrices with polynomial entries. Given an $m \times m$ matrix M over $\mathbb{F}\langle X\rangle$ such that $M[m,m] = f + g \times h$, apply the following:

- Expand M to an $(m+1)\times(m+1)$ matrix by adding a new last row and last column with diagonal entry 1 and remaining new entries zero:

$$\left[\begin{array}{c|c} M & 0 \\ \hline 0 & 1 \end{array}\right].$$

- Then the bottom right 2×2 submatrix is transformed as follows by elementary row and column operations

$$\begin{pmatrix} f+gh & 0 \\ 0 & 1 \end{pmatrix} \to \begin{pmatrix} f+gh & g \\ 0 & 1 \end{pmatrix} \to \begin{pmatrix} f & g \\ -h & 1 \end{pmatrix}$$

Given a polynomial $f \in \mathbb{F}\langle X\rangle$ by repeated application of the above step we will finally obtain a *linear matrix* $L = A_0 + \sum_{i=1}^{n} A_i x_i$, where each $A_i, 0 \le i \le n$ is an $\ell \times \ell$ over $\mathbb{F}$, for some ℓ. The following theorem summarizes its properties.

Theorem 4.3 (Higman Linearization). [13] *Given a polynomial $f \in \mathbb{F}\langle X\rangle$, there are matrices $P, Q \in \mathbb{F}\langle X\rangle^{\ell\times\ell}$ and a linear matrix $L \in \mathbb{F}\langle X\rangle^{\ell\times\ell}$ such that*

$$P\left(\begin{array}{c|c} f & 0 \\ \hline 0 & I_{\ell-1} \end{array}\right) Q = L \tag{1}$$

with P upper triangular, Q lower triangular, and the diagonal entries of both P and Q are all 1's. (Hence, P and Q are both invertible over $\mathbb{F}\langle\!\langle X\rangle\!\rangle$, moreover entries of P^{-1} and Q^{-1} are in $\mathbb{F}\langle X\rangle$).

Instead of a single f, we can apply Higman linearization to a matrix of polynomials $M \in \mathbb{F}\langle X\rangle^{m\times m}$ to obtain a linear matrix L such that $P(M \oplus I_k)Q = L$ for invertible matrices P, Q. Garg et al. [17] gave polynomial time algorithm to carry out Higman linearization for polynomial matrix whose entries are given by noncommutative formulas. We will give an NC algorithm to implement this transformation.

Theorem 4.4. *Let $A \in \mathbb{F}\langle X\rangle^{n\times n}$ be a polynomial matrix such that each entry of A is given by a noncommutative formula of size at most s. Then there is a deterministic* NC *algorithm (with parallel time complexity* $\mathrm{poly}(\log s, \log n)$*) to compute invertible upper and lower triangular matrices $P, Q \in \mathbb{F}\langle X\rangle^{\ell\times\ell}$ with all diagonal entries* 1 *and a linear matrix $L \in \mathbb{F}\langle X\rangle^{\ell\times\ell}$ such that $P\left(\begin{array}{c|c} A & 0 \\ \hline 0 & I_k \end{array}\right) Q = L$, where $\ell = n + k$ and k is $O(n^2 \cdot s)$. All the entries of P, Q are computable by algebraic branching programs of size* $\mathrm{poly}(n, s)$*. Moreover $ncrk(A) + k = ncrk(L)$, hence the rank of A is easily computable from the rank of L.*

The proof of the above theorem is in the appendix.

Using Theorem 2.7, Lemma 4.2 and Theorem 4.4 we get a deterministic NC reduction from multivariate ncRANK to bivariate ncRANK.

Theorem 4.5. *There is a deterministic* NC *reduction from the multivariate ncRANK problem to the bivariate ncRANK problem.*

5 NC Reduction from RIT to bivariate ncRANK

In this section we give an NC-Turing reduction from RIT to bivariate ncRANK. That is, we design an NC algorithm for RIT assuming we have an oracle for bivariate ncRANK. Hrubes and Wigderson in [19] give a polynomial time reduction from RIT to ncRANK problem [19, Theorem 2.6]. Also they show that for any given rational formula Φ there is a log-depth rational formula that is equivalent to Φ [19, Proposition 4.1].

Our key contribution here is to use Cohn's embedding theorem to transform RIT problem to the *bivariate* case. Then we parallelize the Hrubes-Wigderson reduction from RIT to ncRANK. In fact, if the input rational formula is already logarithmic depth then the Hrubes-Wigderson reduction from RIT to ncRANK can be implemented in NC. In this section we design an NC algorithm for depth reduction of rational formulas (possibly with division gates) assuming NC oracle for bivariate ncRANK.[9] Indeed, the construction of an equivalent log-depth rational formula, as described in [19], does not appear to be directly parallelizable, as its description crucially requires rational formula identity testing.[10]

We show that, indeed, the depth-reduction proof in [19] can be parallelized step by step. However, there are some key points where our algorithmic proof is different. Firstly, to solve the RIT instance arising in the depth-reduction proof, we need to recursively depth-reduce the corresponding subformula and then apply Hrubes-Wigderson reduction from RIT to ncRANK on the constructed log-depth subformula. Secondly, we need to handle an important case arising in the proof (namely, the case (2) in the description of the Normal-Form procedure in the proof of the Lemma 5.3 in the Appendix) which was not significant for the existential argument in [19]. In fact to handle this case, we require an argument based on Brent's commutative formula depth reduction [6].

In the detailed proof of Lemma 5.3 given in the Appendix, we first sketch our NC algorithm for depth reduction of rational formula assuming oracle access to bivariate ncRANK. We highlight and elaborate the key steps where our proof

[9]It is an interesting problem to devise an NC algorithm for rational formula depth reduction without oracle access to singularity test.

[10]The overall proof in [19] is based on the Brent's depth-reduction of commutative rational formulas [6]. In the commutative case addressed by Brent, it turns out that the construction procedure does not require oracle access to identity testing and he obtains an NC algorithm for obtaining the depth-reduced formula.

differs from [19]. We need to reproduce some parts of their proof for completeness, for these parts we just sketch the argument and refer to [19] for the details. Using this depth reduction algorithm we give an NC Turing reduction from RIT to bivariate ncRANK, which is a main result of this section.

5.1 Depth reduction for noncommutative formulas with inversion gates

The following is a consequence of results in [19] and [15].

Theorem 5.1 ([19], [15])**.** *Let Φ be a rational formula of size s computing a* non-zero *rational function in $\mathbb{F}\langle\!\langle X \rangle\!\rangle$. If the field $\mathbb{F}$ is sufficiently large and $k > 2s$ then, at a matrix tuple $(M_1, M_2, \ldots, M_n)$ chosen uniformly at random from $M_{k\times k}(\mathbb{F})^n$, the matrix $\hat{\Phi}(M_1, \ldots, M_n)$ is nonsingular with "high" probability.*

If $\mathbb{F}$ is small then we can pick the random matrices over a suitable extension field, and by "high" probability we mean, say, $1 - 2^{-\Omega(s+n)}$.

Let Φ_1 and Φ_2 be correct rational formulas of size at most s computing rational functions in $\mathbb{F}\langle\!\langle X \rangle\!\rangle$. By Theorem 5.1 and a union bound argument, for a random matrix substitution $(M_1, M_2, \ldots, M_n)$ from $M_{k\times k}(\mathbb{F})^n$, inputs to all the inversion gates in Φ_1 and Φ_2 simultaneously evaluate to non-singular matrices with "high" probability. Hence, for k sufficiently large we have $\mathcal{D}_{k,\Phi_1} \cap \mathcal{D}_{k,\Phi_2} \neq \emptyset$. It follows that a random matrix tuple is in $\mathcal{D}_{k,\Phi_1} \cap \mathcal{D}_{k,\Phi_2}$ with high probability.

By Theorem 5.1 and the definition of correct rational formulas (Definition 2.2), it follows that if Φ_1 and Φ_2 are size s correct formulas that are *not* equivalent then for a random matrix substitution of dimension $k > 2s$ both Φ_1 and Φ_2 are defined and they disagree with high probability. As noted in Section 2, equivalent formulas need not have the same domain of definition.

Lemma 5.3 is the main technical result of this section. It describes an NC algorithm for depth-reduction of correct formulas assuming an oracle for bivariate ncRANK. The next lemma is useful for establishing equivalences of formulas arising in the proof of Lemma 5.3. Suppose Φ is a rational formula computing the rational function $\hat{\Phi}$. For a gate v in formula Φ, Φ_v denotes the subformula rooted at v. The formula $\Phi_{v\leftarrow z} \in \mathbb{F}\langle\!\langle X \cup \{z\} \rangle\!\rangle$ is obtained from Φ by replacing subformula Φ_v with fresh variable z.

Lemma 5.2 (local surgery)**.** *Let Φ be a correct rational formula and v be a gate in Φ. Suppose Ψ is a correct rational formula equivalent to Φ_v. Let $\Psi' = (A \cdot z + B) \cdot (C \cdot z + D)^{-1}$ is a formula equivalent to $\Phi_{v\leftarrow z}$ such that A, B, C, D are correct formulas which do not depend upon z and $\hat{C} \cdot \hat{\Delta} + \hat{D} \neq 0$ for any formula Δ such that $\Phi_{v\leftarrow\Delta}$ is correct. Let Φ' denote the rational formula obtained by replacing z in Ψ' by Ψ. Then Φ' is correct and it is equivalent to Φ.*

Proof. From the definitions of Φ_v and $\Phi_{v\leftarrow z}$ it follows that $\Phi = \Phi_{v\leftarrow\Phi_v}$. As Φ is correct, from the properties of formulas C, D as stated in the lemma it follows that

$\hat{C}\hat{\Phi}_v + \hat{D} \neq 0$. Which implies $\hat{C}\hat{\Psi} + \hat{D} \neq 0$ as $\Psi \equiv \Phi_v$. This shows that the formula $\Phi' = (A \cdot \Psi + B) \cdot (C \cdot \Psi + D)^{-1}$ is correct. Now let $\tau = (M_1, \ldots, M_n)$ is a matrix tuple in $\mathcal{D}_\Phi \cap \mathcal{D}_{\Phi'}$, the intersection of domains of definition of Φ and Φ'. This implies $\tau \in \mathcal{D}_{\Phi_v}$ as $\mathcal{D}_\Phi \subseteq \mathcal{D}_{\Phi_v}$, Φ_v being subformula of Φ. Similarly, $\tau \in \mathcal{D}_\Psi$ as $\mathcal{D}_{\Phi'} \subseteq \mathcal{D}_\Psi$, Ψ being a subformula of Φ'. So $\tau \in \mathcal{D}_{\Phi_v} \cap \mathcal{D}_\Psi$. As $\Phi_v \equiv \Psi$, it follows that $\Phi_v(\tau) = \Psi(\tau)$. As $\tau \in \mathcal{D}_\Phi$. It implies that $(M_1, M_2, \ldots, M_n, \Phi_v(\tau)) = (\tau, \Phi_v(\tau)) \in \mathcal{D}_{\Phi_{v \leftarrow z}}$. Similarly, as $\tau \in \mathcal{D}_{\Phi'}$, it follows that $(\tau, \Psi(\tau)) \in \mathcal{D}_{\Psi'}$. As $\Phi_v(\tau) = \Psi(\tau)$, it implies

$$(\tau, \Phi_v(\tau)) = (\tau, \Psi(\tau)) \in \mathcal{D}_{\Phi_{v \leftarrow z}} \cap \mathcal{D}_{\Psi'}$$

As $\Phi_{v \leftarrow z} \equiv \Psi'$, this implies $\Phi_{v \leftarrow z}(\tau, \Phi_v(\tau)) = \Psi'(\tau, \Psi(\tau))$. But $\Phi_{v \leftarrow z}(\tau, \Phi_v(\tau)) = \Phi(\tau)$ and $\Psi'(\tau, \Psi(\tau)) = \Phi'(\tau)$. So we get $\Phi(\tau) = \Phi'(\tau)$ for any $\tau \in \mathcal{D}_\Phi \cap \mathcal{D}_{\Phi'}$. Thus proving $\Phi \equiv \Phi'$. ◻

Lemma 5.3. *Given a* correct *formula Φ of size s computing a rational function $f \in \mathbb{F}\langle\!\langle X \rangle\!\rangle$, for sufficiently large s and absolute constants c, b*

1. *we give an* NC *algorithm, with oracle access to bivariate ncRANK, that outputs a correct formula Φ' of depth at most $c \log_2 s$ which is equivalent to Φ.*
2. *If a variable z occurs in Φ at most once then we give an* NC *algorithm, with bivariate ncRANK as oracle, that constructs* correct *rational formulas A, B, C, D which do not depend on z with depth at most $c \log_2 s + b$ and the formula $\Phi' = (A \cdot z + B) \cdot (C \cdot z + D)^{-1}$ is equivalent to Φ. Moreover, the rational function $\hat{C}\hat{\Psi} + \hat{D} \neq 0$ for any formula Ψ such that $\Phi_{z \leftarrow \Psi}$ is correct.*

The proof of the above lemma is given in the appendix.

Hrubes-Wigderson reduction from RIT to ncRANK Now we recall the polynomial-time reduction from RIT to ncRANK from [19, Theorem 2.6]. Given a rational formula Φ their reduction outputs an invertible linear matrix M in the variables X.[11] Their reduction ensures that the top right entry of M^{-1} is $\hat{\Phi}$, the rational function computed by the formula Φ. It turns out that if Φ is already of logarithmic depth then their reduction can be implemented in NC.

Theorem 5.4 ([19]). *Let Φ be a rational formula of size s and depth $O(\log s)$ computing a rational expression in $\mathbb{F}\langle\!\langle X \rangle\!\rangle$ there is an* NC *algorithm to construct an invertible linear matrix M_Φ such that the top right entry of M_Φ^{-1} is $\hat{\Phi}$.*

Proof. We only briefly sketch the NC algorithm. Their reduction recursively constructs the matrix M_Φ, using the formula structure of Φ.

Given a formula Φ we can compute the sizes of all its subformulas in NC using a standard pointer doubling algorithm. This allows us to estimate the dimensions

[11]Notice that the entries of M^{-1} are elements of the skew field $\mathbb{F}\langle\!\langle X \rangle\!\rangle$

of matrices M_{Φ_v} for subformulas Φ_v for each gate v of Φ. We can also compute in NC the precise location for placement of the sub-matrices M_{Φ_v} inside the matrix M_Φ following their construction. Assuming that Φ is already of logarithmic depth, there are only $O(\log s)$ nested recursive calls for this recursive procedure. This ensures that the overall process can be implemented in NC. □

After constructing linear matrix M_Φ such that the top right entry of M_Φ^{-1} is $\hat{\Phi}$, define matrix M' as $M' = \begin{pmatrix} v^T & M_\Phi \\ 0 & -u \end{pmatrix}$

where u, v are $1 \times k$ vectors, such that $u = (1, 0, \ldots, 0)$ and $v = (0, 0, \ldots, 0, 1)$ where k is the dimension of the matrix M_Φ. It follows that $\hat{\Phi} \neq 0$ iff M' is invertible in the skew field $\mathbb{F}\langle\!\langle X \rangle\!\rangle$ (see e.g. [16, Proposition 3.29]). So we have the following theorem.

Theorem 5.5 ([19]). *Let Φ be a rational formula of size s and depth $O(\log s)$ computing a rational expression in $\mathbb{F}\langle\!\langle X \rangle\!\rangle$ then there is an* NC *algorithm to construct a linear matrix M such that $\hat{\Phi} \neq 0$ iff M is invertible in the skew field $\mathbb{F}\langle\!\langle X \rangle\!\rangle$.*

Now, from Lemma 5.3, Theorem 5.5, and Theorem 3.2, we obtain an NC Turing reduction from multivariate RIT to bivariate ncRANK.

Theorem 5.6. *There is a deterministic* NC *Turing reduction from RIT problem to ncRANK problem for bivariate linear matrices.*

Concluding Remarks. Motivated by the question whether RIT and ncRANK have deterministic NC algorithms, we show that multivariate RIT is NC-reducible to bivariate RIT and multivariate ncRANK is NC-reducible to bivariate ncRANK. RIT is known to be polynomial-time reducible to ncRANK, and indeed that is how the polynomial-time algorithm for RIT works, by reducing to ncRANK and solving ncRANK. We show that RIT is deterministic NC-Turing reducible to ncRANK. We prove this by showing that noncommutative rational formula depth reduction is NC-Turing reducible to ncRANK. The main open problem is to obtain deterministic NC algorithms for bivariate ncRANK and bivariate RIT. We also leave open finding an unconditional NC algorithm for depth-reduction of noncommutative rational formulas.

References

[1] S. A. Amitsur. Rational identities and applications to algebra and geometry. *Journal of Algebra*, 3:304–359, 1966.

[2] Vikraman Arvind, Pushkar S. Joglekar, Partha Mukhopadhyay, and S. Raja. Randomized polynomial-time identity testing for noncommutative circuits. *Theory Comput.*, 15:1–36, 2019.

[3] G. M. Bergman. Skew fields of noncommutative rational functions, after amitsur. *Sé Schü–Lentin–Nivat, Paris*, 1970.

[4] Markus Bläser, Gorav Jindal, and Anurag Pandey. A deterministic PTAS for the commutative rank of matrix spaces. *Theory Comput.*, 14(1):1–21, 2018.

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

[6] Richard P. Brent. The parallel evaluation of general arithmetic expressions. *J. ACM*, 21(2):201–206, 1974.

[7] Abhranil Chatterjee and Partha Mukhopadhyay. The noncommutative edmonds' problem re-visited. *CoRR*, abs/2305.09984, 2023.

[8] P. M. Cohn. The embedding of fir in skew fields. *Proceedings of the London Mathematical Society*, 23:193–213, 1971.

[9] P. M. Cohn. Universal skew fields of fractions. *Sympos. Math.*, 8:135–148, 1972.

[10] P. M. Cohn. *Free Rings and their Relations*. London Mathematical Society Monographs. Academic Press, 1985.

[11] P. M. Cohn. An embedding theorem for free associative algebras. *Mathematica Pannonica*, 1(1):49–56, 1990.

[12] P. M. Cohn. *Introduction to Ring Theory*. Springer, 2000.

[13] P. M. Cohn. *Free Ideal Rings and Localization in General Rings*. New Mathematical Monographs. Cambridge University Press, 2006.

[14] Richard A. DeMillo and Richard J. Lipton. A probabilistic remark on algebraic program testing. *Inf. Process. Lett.*, 7(4):193–195, 1978.

[15] Harm Derksen and Visu Makam. Polynomial degree bounds for matrix semi-invariants. *CoRR*, abs/1512.03393, 2015.

[16] Ankit Garg, Leonid Gurvits, Rafael Mendes de Oliveira, and Avi Wigderson. A deterministic polynomial time algorithm for non-commutative rational identity testing. *CoRR*, abs/1511.03730, 2015.

[17] Ankit Garg, Leonid Gurvits, Rafael Mendes de Oliveira, and Avi Wigderson. Operator scaling: Theory and applications. *Found. Comput. Math.*, 20(2):223–290, 2020.

[18] Graham Higman. The units of group-rings. *Proceedings of the London Mathematical Society*, s2-46(1):231–248, 1940.

[19] Pavel Hrubes and Avi Wigderson. Non-commutative arithmetic circuits with division. *Theory Comput.*, 11:357–393, 2015.

[20] Gábor Ivanyos, Youming Qiao, and K. V. Subrahmanyam. Non-commutative edmonds' problem and matrix semi-invariants. *Comput. Complex.*, 26(3):717–763, 2017.

[21] Gábor Ivanyos, Youming Qiao, and K. V. Subrahmanyam. Constructive non-commutative rank computation is in deterministic polynomial time. *Comput. Complex.*, 27(4):561–593, 2018.

[22] Noam Nisan. Lower bounds for non-commutative computation (extended abstract). In *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing, May 5-8, 1991, New Orleans, Louisiana, USA*, pages 410–418. ACM, 1991.

[23] Sanguthevar Rajasekaran and John H. Reif, editors. *Handbook of Parallel Computing - Models, Algorithms and Applications*. Chapman and Hall/CRC, 2007.

[24] Ran Raz and Amir Shpilka. Deterministic polynomial identity testing in non-commutative models. *Comput. Complex.*, 14(1):1–19, 2005.

[25] L. H. Rowen. *Polynomial identities in ring theory, Pure and Applied Mathematics*. Academic Press Inc., Harcourt Brace Jovanovich Publishers, New York, 1980.

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

[27] James C. Wyllie. *Complexity of Parallel Computation, PhD Thesis*. Cornell University, 1979.

[28] Richard Zippel. Probabilistic algorithms for sparse polynomials. In Edward W. Ng, editor, *Symbolic and Algebraic Computation, EUROSAM '79, An International Symposiumon Symbolic and Algebraic Computation, Marseille, France, June 1979, Proceedings*, volume 72 of *Lecture Notes in Computer Science*, pages 216–226. Springer, 1979.

Appendix

6 Cohn's Embedding Theorem

Let $X = \{x_1, x_2, \ldots, x_n\}$ be a set of n noncommuting variables, and let x, y be a pair of noncommuting variables. The goal of Cohn's construction [11] is to obtain an honest embedding from $\mathbb{F}\langle X\rangle \to \mathbb{F}\langle x, y\rangle$. Indeed, his construction gives an honest embedding map even for a countable set of variables $X = \{x_1, x_2, \ldots\}$.

The first point is that the free noncommutative rings $\mathbb{F}\langle X\rangle$ and $\mathbb{F}\langle x, y\rangle$ have both enough structure[12] that guarantees the following.

Lemma 6.1. *If a homomorphic embedding $\phi : \mathbb{F}\langle X\rangle \to \mathbb{F}\langle x, y\rangle$ can be extended to a homomorphic embedding $\phi : \mathbb{F}\lessdot X \gtrdot \to \mathbb{F}\lessdot x, y \gtrdot$ then, in fact, ϕ is an honest embedding.*

The reason for this is basically, that if an $n \times n$ M over $\mathbb{F}\langle X\rangle$ is a full matrix then it is invertible with an inverse M^{-1} over the skew field $\mathbb{F}\lessdot X \gtrdot$. Thus, it suffices to find a homomorphic embeddings that extends to one between the corresponding free skew fields.

Cohn solved this problem of finding such an embedding [13, Theorem 4.7.3] by an ingenious construction using skew polynomial rings.

Skew Polynomial Rings We recall the definition of skew polynomial rings and state some basic properties (details can be found in Cohn's text [12, Chapter 1.1]). Let R be an *integral domain*[13] and let $\sigma : R \to R$ be a ring *endomorphism*. Let

$$A = \{x^n a_n + x^{n-1} a_{n-1} + \ldots + a_0 \mid \text{ each } a_i \in R\}$$

be the set of all formal univariate polynomials in the indeterminate x which is assumed to not commute with elements of R. Addition of elements in A can be defined component-wise as usual. The multiplication operation is defined with a "twist" to it, using the ring endomorphism σ, which we briefly explain below.

A *σ-derivation* on R is defined as an additive homomorphism $\delta : R \to R$ such that

$$\delta(ab) = \sigma(a)\delta(b) + \delta(a)b \text{ for all } a, b \in R.$$

We define

$$ax = x\sigma(a) + \delta(a), \text{ for all } a \in R,$$

which extends to ring multiplication in A. Under these operations the set A is the *skew polynomial ring* denoted $R[x; \sigma, \delta]$. If the R-endomorphism σ is the identity map 1 and $\delta = 0$ in this definition, then we obtain the univariate polynomial ring $R[x]$ in which the variable x commutes with elements of R.

[12]Technically, both these rings are semifir [11, 13].

[13]That means R is a, possibly noncommutative, ring with unity 1 and without zero divisors.

Following Cohn's construction in his embedding theorem, we consider skew polynomial rings of the form $R[x;1,\delta]$, where the endomorphism $\sigma = 1$. We refer to δ as a derivation and we have:

$$\delta(ab) = a\delta(b) + \delta(a)b \text{ for all } a, b \in R, \text{ and} \tag{2}$$
$$ax = xa + \delta(a), \text{ for all } a \in R. \tag{3}$$

Cohn's Construction We now describe the construction, adding some details to the rather terse description in [13].

We set the integral domain R to be $\mathbb{F}\langle X\rangle$ for $X = \{x_1, x_2, \ldots,\}$. Consider the map $\delta : X \mapsto X$ defined as

$$\delta(x_i) = x_{i+1}.$$

The map δ naturally extends to a unique derivation on $\mathbb{F}\langle X\rangle$ as follows. For scalars $a \in \mathbb{F}$, we define $\delta(ax_i) = a\delta(x_i)$, by linearity. Next, define δ on all monomials in X^*. Let $\delta(x_ix_j) = \delta(x_i)x_j + x_i\delta(x_j)$. In general, for a degree-ℓ monomial $m = x_{i_1}x_{i_2}\ldots x_{i_\ell}$ we define

$$\delta(m) = \delta(x_{i_1}x_{i_2}\ldots x_{i_k})x_{i_{k+1}}x_{i_{k+2}}\ldots x_{i_\ell} + x_{i_1}x_{i_2}\ldots x_{i_k}\delta(x_{i_{k+1}}x_{i_{k+2}}\ldots x_{i_\ell}).$$

It is easy to verify that the above definition of $\delta(m)$ is independent of $k \in [\ell]$. We now extend this definition to the entire ring $\mathbb{F}\langle X\rangle$ by linearity. By an easy induction on the degree of polynomials in $\mathbb{F}\langle X\rangle$ we obtain the following.

Lemma 6.2. *The mapping δ defined above is a derivation on the ring $R = \mathbb{F}\langle X\rangle$.*

Let $H = R[x;1,\delta]$ be the skew polynomial ring defined by the derivation δ described above. By definition, δ satisfies Equations 2 and 3. Therefore, putting $a = x_i$ in Equation 3, for each $i \geq 1$ we have

$$x_{i+1} = x_ix - xx_i = [x_i, x],$$

This actually gives a homomorphic embedding from the ring $\mathbb{F}\langle X\rangle$ to the bivariate ring $\mathbb{F}\langle x,y\rangle$. To see this, we define a map $\beta : X \mapsto \mathbb{F}\langle x,y\rangle$ as follows:

- Let $\beta(x_1) = y$. For $i \geq 2$, let $\beta(x_i) = [\beta(x_{i-1}), x]$.
- We can then naturally extend β to a homomorphism from $\mathbb{F}\langle X\rangle$ to $\mathbb{F}\langle x,y\rangle$, and it is easy to check that it is injective.

Hence we have

Theorem 6.3. [13, Theorem 4.5.3] *β is a homomorphic embedding from $\mathbb{F}\langle X\rangle$ to $\mathbb{F}\langle x,y\rangle$.*

Furthermore, by the definition of δ, the elements of the skew polynomial ring $R[x; 1, \delta]$ are also polynomials in the ring $\mathbb{F}\langle x, y\rangle$. Indeed, the map β can be extended to an isomorphism from the ring $R[x; 1, \delta]$ to $\mathbb{F}\langle x, y\rangle$ as follows: for $f = x^n a_n + x^{n-1}a_{n-1} + \ldots + a_0 \in R[x; 1, \delta]$ define $\beta(f) = x^n\beta(a_n) + x^{n-1}\beta(a_{n-1}) + \ldots + x\beta(a_1) + \beta(a_0)$.

Theorem 6.4. [13, Theorem 4.5.3] *β is a homomorphic embedding from $\mathbb{F}\langle X\rangle$ to $\mathbb{F}\langle x, y\rangle$. Furthermore, β is an isomorphism from $R[x; 1, \delta]$ to $\mathbb{F}\langle x, y\rangle$.*

Using properties of the field of fractions of the skew polynomial ring $R[x; 1, \delta]$ Cohn shows that β extends to an embedding between the skew fields.

Theorem 6.5 (Cohn's embedding theorem)**.** *The embedding map $\beta : \mathbb{F}\langle X\rangle \to \mathbb{F}\langle x, y\rangle$ extends to an embedding $\beta : \mathbb{F}\langle\!\langle X\rangle\!\rangle \to \mathbb{F}\langle\!\langle x, y\rangle\!\rangle$ which implies that β is an honest embedding.*

Proof of Lemma 4.2

Proof. First we describe a recursive construction to compute a formula Φ' equivalent to Φ and inductively prove that the depth of Φ' is $c \log_2 s$ for an absolute constant c. Then we analyze the parallel time complexity of the construction and prove that it can be implemented in NC.

Let A_Φ be $s \times s$ matrix such that for gates $u, v \in \Phi$, $(u, v)^{th}$ entry of A_Φ is 1 if gate v is a descendent of gate u. Using the well-known pointer doubling strategy (see e.g. [27], [23]) we can compute matrix A_Φ in NC. So by adding elements in each row of A_Φ, we can compute $wt(u)$ (that is the number of descendants of gate $u \in \Phi$) in NC. Let v be a gate in Φ such that $\frac{s}{3} \leq wt(v) < \frac{2s}{3}$. Such a gate always exists by a standard argument. Since we can compute the number of descendants of a gate in NC, we can also find such a gate v in NC, by simply having a processor associated to each gate to check the above inequalities. Now we are ready to describe recursive construction of Φ'.

1. In NC find a gate v in Φ such that $\frac{s}{3} \leq wt(v) < \frac{2s}{3}$.
2. Let $r = v_0$ be the root of Φ and $v_1, v_2, \ldots, v_{\ell-1}$ be gates on r to v path in Φ. Let $v = v_\ell$. For $1 \leq i \leq \ell$, let u_i denote a sibling of v_i. Let S_1 be collection of all indices j such that $1 \leq j \leq \ell$, v_j is a product gate and is a right child of its parent. Similarly let S_2 be collection of all indices j such that $1 \leq j \leq \ell$, v_j is a product gate and is a left child of its parent. Define formula $\Psi_1 = \prod_{j \in S_1} \Phi_{u_j}$. The product is computed using sequence of multiplication gates, starting with Φ_{u_j} for the first u_j (one with smallest index $j \in S_1$) each multiplication gate multiplies the product so far from right by Φ_{u_j} for the next gate u_j, $j \in S_1$, along the root to v path. Similarly define formula $\Psi_2 = \prod_{j \in S_2} \Phi_{u_j}$. Let Ψ_3 be a formula obtained from Φ by replacing subformula Φ_v by zero.

3. Recursively in parallel compute log-depth formulas $\Psi'_1, \Psi'_2, \Psi'_3, \Phi'_v$ equivalent to Ψ_1, Ψ_2, Ψ_3 and Φ_v respectively.

4. Define formula Φ' as $(\Psi'_1 \cdot \Phi'_v) \cdot \Psi'_2 + \Psi_3$.

From the definitions of Ψ_1, Ψ_2 and Ψ_3 it is clear that the polynomial computed by Φ equals $(\hat{\Psi}_1 \cdot \hat{\Phi}_v) \cdot \hat{\Psi}_2 + \hat{\Psi}_3$, where $\hat{\Psi}_1, \hat{\Psi}_1, \hat{\Psi}_1, \hat{\Phi}_v$ are the polynomials computed by Ψ_1, Ψ_2, Ψ_3 and Φ_v respectively. Hence, Φ', defined in Step 4, is equivalent to Φ.

Let $d(s)$ denote the upper bound on the depth of the formula output by the above procedure if size s formula is given to it as input. We use induction on the size s to prove that $d(s) \leq c \log_2 s$. As Ψ_1, Ψ_2 are disjoint subformulas of $\Phi_{v\leftarrow z}$, clearly we have $|\Psi_1| + |\Psi_2| \leq |\Phi_{v\leftarrow z}|$. Since $|\Phi_v| \geq \frac{s}{3}$, it implies $|\Psi_1|, |\Psi_2| \leq |\Phi_{v\leftarrow z}| \leq \frac{2s}{3}$. From the definition of Ψ_3, it is clear that $|\Psi_3| \leq |\Phi_{v\leftarrow z}| \leq \frac{2s}{3}$. So the size of each formula Ψ_1, Ψ_2, Ψ_3 and Φ_v is upper bounded by $\frac{2s}{3}$. Hence, inductively, for each of the formulas $\Psi'_1, \Psi'_2, \Psi'_3, \Phi'_v$ the depth is upper bounded by $c \log_2 \frac{2s}{3}$. As Φ' is obtained from $\Psi'_1, \Psi'_2, \Psi'_3, \Phi'_v$ using two multiplications and an addition as in Step 4, it follows that the depth of $\Phi' = d(s) \leq c \log_2 \frac{2s}{3} + 3$. Choosing $c \geq \frac{3}{(\log_2 3 - 1)}$ we get $d(s) \leq c \log_2 \frac{2s}{3} + 3 \leq c \log_2 s$. This completes the induction, proving that the depth of Φ' is at most $c \log_2 s$.

Let $t(s)$ denotes parallel time complexity of the above procedure. Since Steps 1, 2, 4 can be implemented in NC they together take $(\log s)^k$ parallel time for an absolute constant k. As all the recursive calls in Step 3 are processed in parallel, we have the recurrence $t(s) \leq t(2s/3) + (\log s)^k$. Hence, $t(s) \leq (\log s)^{(k+1)}$. This shows that the above procedure can be implemented in NC, completing the proof of the theorem. □

Proof of Theorem 4.4

Proof. By Lemma 4.2 we have an NC algorithm to convert every entry of A to a log-depth formula. We will first describe parallel algorithm for Higman linearization of single polynomial f given by noncommutative log-depth formula Φ. Higman linearization of a polynomial matrix can be handled similarly.

Claim 6.6. *Given a noncommutative formula Φ of size s and depth $O(\log s)$ computing a polynomial f in $\mathbb{F}\langle X\rangle$. We can compute Higman linearization of f in deterministic* NC. *More precisely, we can compute a linear matrix $L_\Phi \in \mathbb{F}\langle X\rangle^{(s+1)\times(s+1)}$, invertible upper and lower triangular matrices $P,Q \in \mathbb{F}\langle X\rangle^{(s+1)\times(s+1)}$ with all diagonal entries 1 such that $P(f \oplus I_s)Q = L_\Phi$. All the entries of P, Q are computable by algebraic branching programs of size* poly(s). *Moreover, $f \not\equiv 0$ iff L_Φ is a full noncommutative rank matrix.*

Proof. The basic idea is to compute the Higman linearization recursively in parallel. Since the formula size is known and for every + or × gate in Φ we need to add a new row and column for the Higman process, we know exactly the dimension of the final matrix L and the precise location for placement of the submatrices inside L, corresponding to each subformula of Φ. To obtain Higman linearization of Φ we recursively in parallel compute Higman linearization of the subformulas rooted at the left and the right child of the root. Based on whether at root of Φ we have a $+$-gate or a $\times$-gate, we appropriately compose Higman linearizations of the sub-formulas rooted at the left and the right child of the root. As depth of Φ is logarithmic, there are only $O(\log s)$ nested recursive calls, this ensures that the overall process can be implemented in NC.

We now give the details of the inductive proof.

Let r be the root of Φ, and g and h be the polynomials computed at the left and right child of r, respectively. If r is a $\times$-gate, the matrix obtained after the first step of the Higman Linearization is

$$\begin{pmatrix} 1 & g \\ 0 & 1 \end{pmatrix}\begin{pmatrix} g\cdot h & 0 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} 1 & 0 \\ -h & 1 \end{pmatrix} = \begin{pmatrix} 0 & g \\ -h & 1 \end{pmatrix}$$

If r is a $+$-gate, we do not explicitly deal with it, as we are eventually interested only in *linearizing* the matrix. Nevertheless, for the $+$-gate case we carry out following step.[14]

$$\begin{pmatrix} 1 & g \\ 0 & 1 \end{pmatrix}\begin{pmatrix} g+h & 0 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} 1 & 0 \\ -1 & 1 \end{pmatrix} = \begin{pmatrix} h & g \\ -1 & 1 \end{pmatrix}$$

It reduces computing Higman linearization of $g + h$ to computing it for g and for h.

We can in NC compute the size of sub-tree rooted at any gate of Φ. Let s_1 and s_2 be sizes of the left and right sub-trees of the root r. Let $w = -h$ if r is a $\times$-gate and is equal to h when r is a $+$-gate. Suppose we compute Higman linearization of g and w recursively in parallel. More precisely, we obtain invertible upper and lower triangular matrices $P_g, Q_g \in \mathbb{F}\langle X\rangle^{(s_1+1)\times(s_1+1)}$ respectively, with all diagonal entries 1 and linear matrix $L_g \in \mathbb{F}\langle X\rangle^{(s_1+1)\times(s_1+1)}$ such that $P_g\,(g \oplus I_{s_1})\,Q_g = L_g$. The entries of P_g, Q_g are given by ABPs of size poly(s_1). Similarly, we obtain invertible upper and lower triangular matrices $P_w, Q_w \in \mathbb{F}\langle X\rangle^{(s_2+1)\times(s_2+1)}$ respectively, with all diagonal entries 1 and linear matrix $L_w \in \mathbb{F}\langle X\rangle^{(s_2+1)\times(s_2+1)}$ such that $P_w\,(w \oplus I_{s_2})\,Q_w = L_w$. The entries of P_w, Q_w are given by ABPs of size poly(s_2). Let $R_g \in \mathbb{F}\langle X\rangle^{1\times s_1}$ be a row matrix obtained by dropping $(1,1)^{th}$ entry from the first row of P_g. Similarly, column matrix C_g is obtained by dropping$(1,1)^{th}$ entry from the first column of Q_g. Analogously, define row and column matrices R_w and C_w. Let P'_g, Q'_g denote bottom right $s_1 \times s_1$ blocks of matrices P_g and Q_g respectively. Similarly, let P'_w, Q'_w denote bottom right $s_2 \times s_2$ blocks of matrices P_w and Q_w respectively. So we have,

$$L_g = \begin{pmatrix} 1 & R_g \\ 0 & P'_g \end{pmatrix}\begin{pmatrix} g & 0 \\ 0 & I_{s_1} \end{pmatrix}\begin{pmatrix} 1 & 0 \\ C_g & Q'_g \end{pmatrix} = \begin{pmatrix} g + R_gC_g & R_gQ'_g \\ P'_gC_g & P'_gQ'_g \end{pmatrix}$$

[14]This facilitates the presentation and calculation of parallel placement of linear submatrices corresponding to the recursive calls inside the final Higman linearized matrix for f.

Similarly, we have

$$L_w = \begin{pmatrix} 1 & R_w \\ 0 & P'_w \end{pmatrix} \begin{pmatrix} w & 0 \\ 0 & I_{s_2} \end{pmatrix} \begin{pmatrix} 1 & 0 \\ C_w & Q'_w \end{pmatrix} = \begin{pmatrix} w + R_w C_w & R_w Q'_w \\ P'_w C_w & P'_w Q'_w \end{pmatrix}$$

Now we are ready to define matrices P, Q, L using matrices P_g, Q_g, P_w, Q_w so that $P(f \oplus I_s)Q = L$.

- **r is $\times$-gate**

  After the first step of Higman linearization on f the matrix obtained is $\begin{pmatrix} 0 & g \\ w & 1 \end{pmatrix}$.

  When we obtain Higman linearization for g, w recursively, these polynomials sit at $(1,1)^{th}$ entry of $g \oplus I_{s_1}$ and $w \oplus I_{s_2}$ respectively. Whereas in the matrix above, the polynomials g, w sit at $(1,2)^{th}, (2,1)^{th}$ entries respectively. Now we define block matrices $\tilde{P_g}, \tilde{P_w}$ and $\tilde{Q_g}, \tilde{Q_w}$ for performing row and column operations, respectively, on the appropriate rows and columns of the matrix, taking into account the location of g and h. Whenever we are carrying out linearization for g it keeps the block corresponding to linearization of h intact and vice-versa. Define $\tilde{P_g}, \tilde{P_w}$ as

$$\tilde{P_g} = \left(\begin{array}{cc|cc} 1 & 0 & R_g & 0 \\ 0 & 1 & 0 & 0 \\ \hline \multicolumn{2}{c|}{\multirow{2}{*}{0}} & P'_g & 0 \\ & & 0 & I_{s_2} \end{array}\right), \tilde{P_w} = \left(\begin{array}{cc|cc} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & R_w \\ \hline \multicolumn{2}{c|}{\multirow{2}{*}{0}} & I_{s_1} & 0 \\ & & 0 & P'_w \end{array}\right)$$

  Similarly, define $\tilde{Q_g}, \tilde{Q_w}$ as

$$\tilde{Q_g} = \left(\begin{array}{cc|cc} 1 & 0 & \multicolumn{2}{c}{\multirow{2}{*}{0}} \\ 0 & 1 & & \\ \hline 0 & C_g & Q'_g & 0 \\ 0 & 0 & 0 & I_{s_2} \end{array}\right), \tilde{Q_w} = \left(\begin{array}{cc|cc} 1 & 0 & \multicolumn{2}{c}{\multirow{2}{*}{0}} \\ 0 & 1 & & \\ \hline 0 & 0 & I_{s_1} & 0 \\ C_w & 0 & 0 & Q'_w \end{array}\right)$$

  $\tilde{P_g}, \tilde{Q_g}$ carry out linearization of g, $\tilde{P_w}, \tilde{Q_w}$ carry out linearization of w. Define $P_1 = \begin{pmatrix} 1 & g \\ 0 & 1 \end{pmatrix} \oplus I_{s_1+s_2}$ and $Q_1 = \begin{pmatrix} 1 & 0 \\ w & 1 \end{pmatrix} \oplus I_{s_1+s_2}$, which would carry out the first step of linearization.

  Finally, define $P = \tilde{P_w}\tilde{P_g}P_1$. We can see that $P = \left(\begin{array}{cc|cc} 1 & g & R_g & 0 \\ 0 & 1 & 0 & R_w \\ \hline \multicolumn{2}{c|}{\multirow{2}{*}{0}} & P'_g & 0 \\ & & 0 & P'_w \end{array}\right)$.

Similarly, define $Q = \tilde{Q_w}\tilde{Q_g}Q_1 = \left(\begin{array}{cc|cc} 1 & 0 & \multicolumn{2}{c}{\multirow{2}{*}{0}} \\ w & 1 & & \\ \hline 0 & C_g & Q'_g & 0 \\ C_w & 0 & 0 & Q'_w \end{array}\right)$

As P'_g and P'_w are bottom right blocks of the upper triangular matrices P_g and P_w, it follows that P'_g, P'_w, and hence also P is invertible, and also upper triangular with all diagonal entries 1. Similarly, Q is invertible and lower triangular with all diagonal entries 1. As P, Q are realized as a matrix product as defined above and entries of P_g, Q_g and P_w, Q_w are given by ABPs of size $\text{poly}(s_1)$, $\text{poly}(s_2)$ respectively, we can construct ABPs of $\text{poly}(s)$ size for entries of P and Q in NC.

Now, define $L_\Phi = P(f \oplus I_s)Q = \left(\begin{array}{cc|cc} 0 & g + R_gC_g & R_gQ'_g & 0 \\ w + R_wC_w & 1 & 0 & R_wQ'_w \\ \hline 0 & P'_gC_g & P'_gQ'_g & 0 \\ P'_wC_w & 0 & 0 & P'_wQ'_w \end{array}\right)$

Clearly, L_Φ is linear as L_g and L_w are linear. Note that we do not need to compute L_Φ as a product $P(f \oplus I_s)Q$. If we explicitly know linear entries of L_g, L_w (which we do know recursively) we can explicitly compute linear entries of L_Φ.

- **r is +-gate**

  This case is handled similarly. In case of +-gate, the matrix obtained after first step is $\begin{pmatrix} w & g \\ -1 & 1 \end{pmatrix}$. So to ensure the Higman linearized matrices of g and w are correctly placed inside the Higman linearized matrix for f we define P, Q as

$$P = \left(\begin{array}{cc|cc} 1 & g & R_g & R_w \\ 0 & 1 & 0 & 0 \\ \hline \multicolumn{2}{c|}{\multirow{2}{*}{0}} & P'_g & 0 \\ & & 0 & P'_w \end{array}\right), Q = \left(\begin{array}{cc|cc} 1 & 0 & \multicolumn{2}{c}{\multirow{2}{*}{0}} \\ -1 & 1 & & \\ \hline 0 & C_g & Q'_g & 0 \\ C_w & 0 & 0 & Q'_w \end{array}\right)$$

and define L_Φ as

$$L_\Phi = P(f \oplus I_s)Q = \left(\begin{array}{cc|cc} w + R_wC_w & g + R_gC_g & R_gQ'_g & R_wQ'_w \\ -1 & 1 & 0 & 0 \\ \hline 0 & P'_gC_g & P'_gQ'_g & 0 \\ P'_wC_w & 0 & 0 & P'_wQ'_w \end{array}\right)$$

Clearly, any entry of L_Φ is a scalar or some entry of L_g or L_w, so L_Φ is linear. Again, entries of L_Φ can be computed explicitly, given the matrices L_g and L_w explicitly. This completes the Case 2.

As P, Q are invertible, it implies $\text{ncrk}(f \oplus I_s) = \text{ncrk}(L)$, which implies L is full iff $f \not\equiv 0$. This proves the claim.

□

In the general case, we need to Higman linearize an $n \times n$ polynomial matrix A. We will in parallel compute Higman linearization matrices for each entry of A using the NC algorithm described in the above claim. Let $M_{i,j}$ be $(s_{i,j}+1) \times (s_{i,j}+1)$ linear matrix corresponding to $(i,j)^{th}$ entry for $1 \leq i,j \leq n$. That is $M_{i,j} = P_{i,j}(A_{i,j} \oplus I_{s_{i,j}})Q_{i,j}$ where $P_{i,j}, Q_{i,j}$ are invertible upper and lower triangular matrices respectively. We have polynomial sized ABPs for entries of $P_{i,j}$ and $Q_{i,j}$.

Let $M'_{i,j}$ denote $s_{i,j} \times s_{i,j}$ bottom right block of $M_{i,j}$. Let $R_{i,j}$ denote $1 \times s_{i,j}$ row matrix obtained by dropping first entry of first row of $M_{i,j}$. Similarly, let $C_{i,j}$ be $s_{i,j} \times 1$ column matrix obtained by dropping first entry of the first column of $M_{i,j}$. Then, the matrix L is a 2×2 block matrix such that

1. The top left block of L is $n \times n$ and for $1 \leq i,j \leq n$, $(i,j)^{th}$ entry of the block is $(1,1)^{th}$ entry of $M_{i,j}$.

2. The top right block of L is the matrix

$$\begin{pmatrix} R_{1,1}\ R_{1,2}\ \ldots R_{1,n} & & & \\ & R_{2,1}\ R_{2,2}\ \ldots R_{2,n} & & \\ & & \ddots & \\ & & & R_{n,1}\ R_{n,2}\ \ldots R_{n,n} \end{pmatrix}$$

3. The bottom left block of L is the matrix $[B_1 B_2 \ldots B_n]^T$ where for $1 \leq i \leq n$, B_i is the matrix

$$\begin{pmatrix} C_{i,1} & & & \\ & C_{i,2} & & \\ & & \ddots & \\ & & & C_{i,n} \end{pmatrix}$$

4. The bottom right block of L is a matrix $\begin{pmatrix} D_1 & & & \\ & D_2 & & \\ & & \ddots & \\ & & & D_n \end{pmatrix}$ where D_i is a matrix $\begin{pmatrix} M'_{i,1} & & & \\ & M'_{i,2} & & \\ & & \ddots & \\ & & & M'_{i,n} \end{pmatrix}$

All unspecified entries in the above matrices are zero. As done in the proof of Claim 6.6, we can easily define invertible upper and lower triangular matrices P and Q such that $P(A \oplus I_k)Q = L$. Also we can obtain ABPs computing entries of P, Q from ABPs for the entries of the matrices $P_{i,j}, Q_{i,j}$, $1 \leq i, j \leq n$. Since P, Q are invertible, it implies that ncrk(L) is equal to ncrk$(A \oplus I_k)$ which is equal to ncrk$(A) + k$. Clearly, the dimension of the bottom right block of L is $O(n^2 \cdot s)$ which implies similar bound on k. This completes the proof of the theorem. □

Proof of Lemma 5.3

Proof. We give a recursive construction for both the parts and prove the correctness by induction on s, the size of the formula Φ.

Depth-reduce(Φ)

Input: A correct formula Φ of size s computing a rational function in $\mathbb{F}\langle\!\langle X \rangle\!\rangle$.

Output: A correct formula Φ' of depth at most $c \log_2 s$ such that $\Phi \equiv \Phi'$.

1. Find a gate $v \in \Phi$ such that $\frac{s}{3} < wt(v) \leq \frac{2s}{3}$.
2. In Parallel construct formulas Ψ, Δ such that Ψ = Depth-Reduce(Φ_v) and Δ = Normal-Form($\Phi_{v \leftarrow z}, z$).
3. Obtain formula Φ' from Δ by replacing z in Δ by Ψ.
4. Output Φ'.

Normal-Form(Φ, z)

Input: A correct formula Φ of size s computing a rational function in $\mathbb{F}\langle\!\langle X \rangle\!\rangle$ and a variable $z \in \Phi$ which appears at most once in Φ

Output: A correct formula Φ' which is of the form

$$\Phi' = (Az + B)(Cz + D)^{-1}$$

where A, B, C and D are correct rational formulas which do not depend on z with depth at most $c \log_2 s + b$. Moreover, the rational function $\hat{C}\hat{\Psi} + \hat{D} \neq 0$ for any formula Ψ such that $\Phi_{z \leftarrow \Psi}$ is correct.

Let $v_1, v_2, \ldots, v_\ell = z$ be gates on the path from root r of Φ to the leaf gate z, such that v_j is not an inverse gate. So parent of each v_j has two children, and let u_i denote the sibling of v_i. Use pointer doubling based parallel algorithm (as mentioned in the proof of Lemma 4.2) to compute $wt(u_i)$ and $wt(v_i)$ for all $i \in [\ell]$. We call gate v_i for $i \in [\ell]$ as a *balanced* gate if $wt(\Phi_{v_i}), wt(\Phi_{v_i \leftarrow z'}) \leq \frac{5s}{6}$. Now we consider two cases.

1. There exist a balanced gate v_i:

   (a) Let $v = v_i$. In parallel compute formulas Ψ_1, Ψ_2 such that

$$\begin{aligned}\Psi_1 &= (A_1 z' + B_1)(C_1 z' + D_1)^{-1} = \text{Normal-Form}(\Phi_{v \leftarrow z'}, z')\\ \Psi_2 &= (A_2 z + B_2)(C_2 z + D_2)^{-1} = \text{Normal-Form}(\Phi_v, z)\end{aligned}$$

(b) Define formulas A, B, C, D as $A_1 \cdot A_2 + B_1 \cdot C_2$, $A_1 \cdot B_2 + B_1 \cdot D_2$, $C_1 \cdot A_2 + D_1 \cdot C_2$ and $C_1 \cdot B_2 + D_1 \cdot D_2$ respectively.

(c) Let $\Phi' = (A \cdot z + B) \cdot (C \cdot z + D)^{-1}$

(d) Output Φ' and halt

2. There does not exist a balanced gate:
In this case, we can prove that there is a unique $i \in [\ell]$ such that $wt(u_i) > \frac{s}{6}$.

(a) Let v be parent of the gate v_i.

(b) In Parallel find formulae $\Psi_1, \Psi_2, \Psi_3, \Psi_4$ such that

$$\begin{aligned}
\Psi_1 &= (A_1 z' + B_1)(C_1 z' + D_1)^{-1} = \text{Normal-Form}(\Phi_{v \leftarrow z'}, z') \\
\Psi_2 &= (A_2 z + B_2)(C_2 z + D_2)^{-1} = \text{Normal-Form}(\Phi_{v_i}, z) \\
\Psi_3 &= \text{Depth-Reduce}(\Phi_{u_i}) \\
\Psi_4 &= (Az + B)(Cz + D)^{-1} = \text{Normal-Form}(\Phi'', z)
\end{aligned}$$

where Φ'' is obtained by replacing sub-tree rooted at u_i by 0 in Φ.

(c) Using the algorithm of Theorem 5.5 check if $\hat{\Psi}_3 \equiv 0$ in NC with oracle access to bivariate ncRANK. If $\hat{\Psi}_3 \equiv 0$ then output Ψ_4 and halt.

(d) If $\hat{\Psi}_3 \not\equiv 0$ then let $\Phi' = (A \cdot z + B) \cdot (C \cdot z + D)^{-1}$, where

$$A = \begin{cases} A_1 A_2 + B_1 C_2 + A_1 \hat{\Psi}_3 C_2 & \text{if } v_i \text{ is a } +\text{-gate} \\ A_1 \hat{\Psi}_3 A_2 + B_1 C_2 & \text{if } v_i \text{ is a } \times\text{-gate and is a right child of } v \\ A_1 A_2 + B_1 \hat{\Psi}_3^{-1} C_2 & \text{if } v_i \text{ is a } \times\text{-gate and is a left child of } v \end{cases}$$

$$B = \begin{cases} A_1 B_2 + B_1 D_2 + A_1 \hat{\Psi}_3 D_2 & \text{if } v_i \text{ is a } +\text{-gate} \\ A_1 \hat{\Psi}_3 B_2 + B_1 D_2 & \text{if } v_i \text{ is a } \times\text{-gate and is a right child of } v \\ A_1 B_2 + B_1 \hat{\Psi}_3^{-1} D_2 & \text{if } v_i \text{ is a } \times\text{-gate and is a left child of } v \end{cases}$$

$$C = \begin{cases} C_1 A_2 + D_1 C_2 + C_1 \hat{\Psi}_3 C_2 & \text{if } v_i \text{ is a } +\text{-gate} \\ C_1 \hat{\Psi}_3 A_2 + D_1 C_2 & \text{if } v_i \text{ is a } \times\text{-gate and is a right child of } v \\ C_1 A_2 + D_1 \hat{\Psi}_3^{-1} C_2 & \text{if } v_i \text{ is a } \times\text{-gate and is a left child of } v \end{cases}$$

$$D = \begin{cases} C_1 B_2 + D_1 D_2 + C_1 \hat{\Psi}_3 D_2 & \text{if } v_i \text{ is a } +\text{-gate} \\ C_1 \hat{\Psi}_3 B_2 + D_1 D_2 & \text{if } v_i \text{ is a } \times\text{-gate and is a right child of } v \\ C_1 B_2 + D_1 \hat{\Psi}_3^{-1} D_2 & \text{if } v_i \text{ is a } \times\text{-gate and is a left child of } v \end{cases}$$

(e) Output Φ' and halt.

Case 2 above is not explicitly dealt with in [19] as their focus is on the *existence* of a log-depth formula equivalent to Φ. In contrast to that, in our case we want to *algorithmically construct* log-depth formula Φ' equivalent to Φ. This makes the details of Case 2 crucial as the construction of Φ' in Case 2 depends on whether $\hat{\Phi_{u_i}} \equiv 0$ or not. To solve this RIT instance we need to recursively compute log-depth formula Ψ_3 equivalent to Φ_{u_i} and then invoke algorithm of theorem 5.5 to carry out RIT test in NC with oracle access to bivariate ncRANK, as in Step (c).

We first prove the correctness of both the algorithms described above using induction on s, then we analyze the parallel complexity of both the algorithms. We will choose appropriate constants c, b during the proof.

Correctness of the algorithm Depth-Reduce We know that there exists a gate $v \in \Phi$ such that $\frac{s}{3} < wt(v) \leq \frac{2s}{3}$. As in proof of Lemma 4.2 we can find such a gate v as required by Step 1 of the Depth-Reduce algorithm. Clearly, the formulas Φ_v and $\Phi_{v \leftarrow z}$ are of size at most $2s/3$. Using inductive assumption, we can construct a correct formula Ψ such that depth of Ψ is at most $c \log_2 \frac{2s}{3}$ and $\Psi \equiv \Phi_v$. Again using inductive assumption we can construct correct formulas A, B, C, D (which do not depend on z) of depth at most $c \log_2 \frac{2s}{3} + b$ such that the formula $\Delta = (A \cdot z + B) \cdot (C \cdot z + D)^{-1}$ is equivalent to $\Phi_{v \leftarrow z}$. Since Φ is equal to the formula obtained from $\Phi_{v \leftarrow z}$ by replacing z by Φ_v and Φ is correct so from inductive assumption it follows that $\hat{C}\hat{\Phi_v} + \hat{D} \neq 0$.

As $\Psi \equiv \Phi_v$, $\Delta \equiv \Phi_{v \leftarrow z}$ and $\hat{C}\hat{\Phi_v} + \hat{D} \neq 0$ from Lemma 5.2 it follows that Φ' is correct and $\Phi' \equiv \Phi$. Since $\Phi' = (A \cdot \Psi + B) \cdot (C \cdot \Psi + D)^{-1}$ we get that depth of Φ' is at most $c \log_2 \frac{2s}{3} + b + 4$. By choosing constant $c \geq \frac{b+4}{(\log_2 3 - 1)}$, we get that the depth of Φ' is at most $c \log_2 s$. Completing the inductive argument for the correctness proof of the algorithm Depth-Reduce.

Correctness of the algorithm Normal-Form In case (1) we know that there exists a balanced gate $v = v_i$. We have $wt(\Phi_v), wt(\Phi_{v \leftarrow z'}) \leq \frac{5s}{6}$. So by inductive assumption we know that the formulas A_j, B_j, C_j, D_j for $j \in \{1, 2\}$ are correct, and their depths are at most $c \log_2 \frac{5s}{6} + b$. Now using compositionality of the z-normal forms as in Proposition 4.1 of [19] it follows that the formula $\Phi' = (A \cdot z + B) \cdot (C \cdot z + D)^{-1}$ is equivalent to Φ where A, B, C, D are $A_1 \cdot A_2 + B_1 \cdot C_2$, $A_1 \cdot B_2 + B_1 \cdot D_2$, $C_1 \cdot A_2 + D_1 \cdot C_2$ and $C_1 \cdot B_2 + D_1 \cdot D_2$ respectively. Also, clearly the depth of A, B, C, D is at most $c \log_2 \frac{5s}{6} + b + 2$. By choosing $c \geq \frac{2}{(\log_2 6 - \log_2 5)}$ it follows that the depths of formulas A, B, C, D are at most $c \log_2 s + b$. To complete the inductive proof we need to prove that $\hat{C}\hat{\pi} + \hat{D} \neq 0$ for any formula π such that $\Phi_{z \leftarrow \pi}$ is correct. Let π be such that $\Phi_{z \leftarrow \pi}$ is correct. For simplicity lets denote formulas Φ_v and $\Phi_{v \leftarrow z'}$ by α and β respectively. Since $\Phi_{z \leftarrow \pi}$ is correct, it implies $\alpha_{z \leftarrow \pi}$ is correct as α is a subformula of Φ. So by inductive assumption $\hat{C_2}\hat{\pi} + \hat{D_2} \neq 0$. Since $\Phi_{z \leftarrow \pi}$ is correct, β being a subformula of Φ it also implies $\beta_{z' \leftarrow \gamma}$ is correct where $\gamma = (A_2 \cdot \pi + B_2) \cdot (C_2 \cdot \pi + D_2)^{-1}$. By inductive assumption we get

$$\begin{aligned}
\hat{C}_1\hat{\gamma} + D_1 &\neq 0 \text{ which implies} \\
\hat{C}_1[(\hat{A}_2 \cdot \hat{\pi} + \hat{B}_2) \cdot (\hat{C}_2 \cdot \hat{\pi} + \hat{D}_2)^{-1}] + D_1 &\neq 0 \text{ which implies} \\
\hat{C}_1(\hat{A}_2 \cdot \hat{\pi} + \hat{B}_2) + \hat{D}_1(\hat{C}_2 \cdot \hat{\pi} + \hat{D}_2) &\neq 0 \text{ as } \hat{C}_2\hat{\pi} + \hat{D}_2 \neq 0 \\
\text{So } (\hat{C}_1\hat{A}_2 + \hat{D}_1\hat{C}_2)\hat{\pi} + (\hat{C}_1\hat{B}_2 + \hat{D}_1\hat{D}_2) &\neq 0 \text{ which implies} \\
\hat{C}\hat{\pi} + \hat{D} &\neq 0
\end{aligned}$$

This completes the proof for case 1.

Next we argue that case 1 and case 2 together cover all the possibilities. To see this, we will argue that if there does not exist a unique u_i with $wt(u_i) \geq \frac{s}{6}$ then there must exist a balanced gate. There are two possibilities: either for every gate u_i, $wt(u_i) \leq \frac{s}{6}$ or there are two or more gates u_i's with $wt(u_i) > \frac{s}{6}$ If for every $i \in [\ell]$, $wt(u_i) \leq \frac{s}{6}$, we find smallest j such that $\sum_{i=1}^{j} wt(u_i) > \frac{s}{6}$. Clearly $\sum_{i=1}^{j} wt(u_i) \leq \frac{2s}{6}$, which implies $wt(\Phi_v), wt(\Phi_{v\leftarrow z'}) \leq \frac{5s}{6}$ for $v = v_{i+1}$. So v is balanced. If there are two or more u_i's such that $wt(u_i) > \frac{s}{6}$ then v be parent of gate u_i such that i is a largest index with $wt(u_i) > \frac{s}{6}$. Clearly v is a balanced gate. This proves that case 1, 2 together cover all possibilities.

Assume that there is a unique $i \in [\ell]$ such that $wt(u_i) > \frac{s}{6}$. We first apply Depth-Reduce on formula Φ_{u_i} and get a log-depth formula Ψ_3 equivalent to Φ_{u_i}, we carry out this depth reduction to efficiently test if $\Phi_{u_i} \equiv 0$?. We will give details on this later when we figure out the parallel time complexity of the algorithm. Now when $\Phi_{u_i} \equiv \Psi_3 \equiv 0$, clearly formula $\Phi \equiv \Phi''$ where Φ'' is a formula obtained from Φ by replacing sub-formula rooted at u_i by 0. As $wt(u_i) > \frac{s}{6}$, we have $|\Phi''| \leq \frac{5s}{6}$. So by inductive assumption, the correct sub-formulas A, B, C, D of Ψ_4 obtained by recursive call Normal-Form(Φ'', z) have depth at most $c \log_2 \frac{5s}{6} + b \leq c \log_2 s + b$ and $\Psi_4 \equiv \Phi'' \equiv \Phi$. So it follows, $\Phi_{z\leftarrow\pi} \equiv \Phi''_{z\leftarrow\pi}$. Consequently $\Phi_{z\leftarrow\pi}$ is correct iff $\Phi''_{z\leftarrow\pi}$ is correct. So from inductive hypothesis it follows that $\hat{C}\hat{\pi} + \hat{D} \neq 0$ for any formula π such that $\Phi_{z\leftarrow\pi}$ is correct. This proves the correctness of Normal-form procedure when $\Phi_{u_i} \equiv 0$.

Now let $\Phi_{u_i} \not\equiv 0$. Let v be the parent of u_i. Below we discuss the case when v is $\times$-gate and u_i is a right child of v.

We have $\Psi_2 \equiv \Phi_{v_i} \equiv (A_2 \cdot z + B_2) \cdot (C_2 \cdot z + D_2)^{-1}$. Let $h_1 = A_2 \cdot z + B_2$ and $h_2 = C_2 \cdot z + D_2)$. So $\Phi_{v_i} \equiv h_1 \cdot h_2^{-1}$. Now as v is $\times$-gate and v_i, u_i are left and right children of v respectively. So we get $\Phi_v \equiv h_1 h_2^{-1} \Phi_{u_i} \equiv h_1 h_2^{-1} \Psi_3$. We have $\Psi_1 \equiv \Phi_{v\leftarrow z'} = (A_1 z' + B_1)(C_1 z' + D_1)^{-1}$. So we get

$$\begin{aligned}
\Phi &\equiv (A_1 \cdot (h_1 h_2^{-1} \Psi_3) + B_1) \cdot (C_1 \cdot (h_1 h_2^{-1} \Psi_3) + D_1)^{-1} \\
&\equiv (A_1 \cdot h_1 + B_1 \cdot \Psi_3^{-1} h_2) \cdot h_2^{-1} \Psi_3 \cdot [(C_1 \cdot h_1 + D_1 \cdot \Psi_3^{-1} h_2) \cdot h_2^{-1} \Psi_3]^{-1} \\
&\equiv (A_1 \cdot h_1 + B_1 \cdot \Psi_3^{-1} h_2) \cdot h_2^{-1} \Psi_3 \cdot \Psi_3^{-1} h_2 \ \cdot (C_1 \cdot h_1 + D_1 \cdot \Psi_3^{-1} h_2)^{-1} \\
&\equiv (A_1 \cdot h_1 + B_1 \cdot \Psi_3^{-1} h_2) \cdot (C_1 \cdot h_1 + D_1 \cdot \Psi_3^{-1} h_2)^{-1}
\end{aligned}$$

By substituting values of h_1, h_2 and simplifying we get that $\Phi \equiv (A\cdot z+B)\cdot(C\cdot z+D)^{-1}$ where A, B, C, D are $A_1 \cdot A_2 + B_1 \cdot \Psi_3^{-1} \cdot C_2$, $A_1 \cdot B_2 + B_1 \cdot \Psi_3^{-1} \cdot D_2$, $C_1 \cdot A_2 + D_1 \cdot \Psi_3^{-1} \cdot C_2$ and $C_1 \cdot B_2 + D_1 \cdot \Psi_3^{-1} \cdot D_2$ respectively as defined in Step 2(d).

As $wt(u_i) > \frac{s}{6}$, clearly $|\Phi_{v_i}|, |\Phi_{v\leftarrow z'}| \leq \frac{5s}{6}$. Since

$$\begin{aligned} \Psi_1 &= (A_1 z' + B_1)(C_1 z' + D_1)^{-1} = \text{Normal-Form}(\Phi_{v\leftarrow z'}, z') \\ \Psi_2 &= (A_2 z + B_2)(C_2 z + D_2)^{-1} = \text{Normal-Form}(\Phi_{v_i}, z) \end{aligned}$$

So by inductive assumptions the sub-formulas A_1, B_1, C_1, D_1 of Ψ_1 and the sub-formulas A_2, B_2, C_2, D_2 of Ψ_2 are correct and have depths at most $c \log_2 \frac{5s}{6} + b$. As Ψ_3 = Depth-Reduce(Φ_{u_i}) and $|\Phi_{u_i}| < s$ by inductive assumption we get that the depth of Ψ_3 is at most $c \log_2 s$. Clearly $c \log_2 \frac{5s}{6} + b \leq c \log_2 s$ for $c \geq \frac{b}{\log_2 6 - \log_2 5)}$. So from expressions for A, B, C, D it follows that the depth of A, B, C, D is at most depth of Ψ_3 plus 4. Which implies that the depth of A, B, C, D is at most $c \log_2 s + 4 \leq c \log_2 s + b$ if the constant $b \geq 4$. This gives us the desired bound on the depth of A, B, C, D. To summarize if we choose constant $b \geq 4$ and choose constant c such that it satisfies all the lower bounds required in different steps of the above proof, we will get the desired bound on the depth of A, B, C, D. We can show that $\hat{C}\hat{\pi} + \hat{D} \neq 0$ for any formula π such that $\Phi_{z\leftarrow\pi}$ is correct. The proof is similar to one for Case (1), we additionally need to take into account $\times$-gate at v while composing z-Normal forms Ψ_1 and Ψ_2. We skip the details.

When v is a v is a $\times$-gate and u_i is a left child of v, the composition of z-normal forms is easy as we do not need an oracle access for RIT as in the case discussed above when u_i is the right child. In the commutative case we can by default assume that u_i is the left child. Precisely for this reason Brent's construction [6] is independent of whether $\Phi_{u_i} \equiv 0$ or not. We skip the details of cases when v is a $+$-gate or v is $\times$-gate and u_i is the left child which can be handled similar to case (1). This proves the correctness of the procedure Normal-Form.

Next we analyze the parallel time complexity of both the procedures. Let $t_1(s), t_2(s)$ denote the parallel time complexities of the procedures Depth-Reduce and Normal-Form respectively. The step (1) of the Depth reduce procedure can be implemented in NC so it has parallel time complexity $(\log_2 s)^k$ for some absolute constant k. As both the formulas Φ_v and $\Phi_{v\leftarrow z}$ have sizes at most $2s/3$, the parallel time complexity of step (2) is at most $\max(t_1(2s/3), t_2(2s/3))$. So we get the following recurrence for $t_1(s)$.

$$t_1(s) \leq (\log_2 s)^a + \max\left(t_1\left(\frac{2s}{3}\right) \ , \ t_2\left(\frac{2s}{3}\right)\right)$$

where a is an absolute constant. Now we obtain recurrence for $t_2(s)$. In Case (1) of Normal-Form when there exist a balanced gate v_i, we can find such a gate in NC. Both the formulas Φ_v and $\Phi_{v\leftarrow z'}$ have the sizes at most $5s/6$ so step 1(a) takes parallel time $\max(t_1(5s/6), t_2(5s/6))$. In case (2) the formulas $\Phi_{v\leftarrow z'}$, Φ_{v_i} and Φ'' all have sizes at most $5s/6$ and formula Φ_{u_i} has size at most $s-1$. So collectively we get the following recurrence for $t_2(s)$

$$\begin{aligned} t_2(s) &\leq (\log_2 s)^b + \max\left(t_1\left(\frac{5s}{6}\right) \, , \, t_2\left(\frac{5s}{6}\right) \, , \, t_1(s-1)\right) \\ &\leq (\log_2 s)^b + \max\left(t_2\left(\frac{5s}{6}\right) \, , \, t_1(s)\right) \end{aligned}$$

for an absolute constant b. The upper bound $t_1(s), t_2(s) \leq (\log_2 s)^c$ for sufficiently large constant c follows from an easy induction. Hence both the procedures can be implemented in deterministic NC.

□