Evaluating matrix circuits

Daniel König and Markus Lohrey

Universität Siegen, Germany

Abstract. The circuit evaluation problem (also known as the compressed word problem) for finitely generated linear groups is studied. The best upper bound for this problem is coRP, which is shown by a reduction to polynomial identity testing (PIT). Conversely, the compressed word problem for the linear group $\mathsf{SL}_3(\mathbb{Z})$ is equivalent to PIT. In the paper, it is shown that the compressed word problem for every finitely generated nilpotent group is in $\mathsf{DET} \subseteq \mathsf{NC}^2$. Within the larger class of polycyclic groups we find examples where the compressed word problem is at least as hard as PIT for skew arithmetical circuits.

1 Introduction

The study of circuit evaluation problems has a long tradition in theoretical computer science and is tightly connected to many aspects in computational complexity theory. One of the most important circuit evaluation problems is *polynomial identity testing (PIT)*: The input is an arithmetic circuit, whose internal gates are labelled with either addition or multiplication and its input gates are labelled with variables $(x_1, x_2, \ldots)$ or constants $(-1, 0, 1)$, and it is asked whether the output gate evaluates to the zero polynomial (in this paper, we always work in the polynomial ring over the coefficient ring $\mathbb{Z}$ or $\mathbb{Z}_p$ for a prime p). Based on the Schwartz-Zippel-DeMillo-Lipton Lemma, Ibarra and Moran [10] proved that PIT over $\mathbb{Z}$ or $\mathbb{Z}_p$ belongs to the class coRP (co-randomized polynomial time). Whether PIT $\in \mathsf{P}$ is an important problem. In [11] it was shown that if there is a language in $\mathsf{DTIME}(2^{\mathcal{O}(n)})$ with circuit complexity $2^{\Omega(n)}$, then $\mathsf{P} = \mathsf{BPP}$ (and hence $\mathsf{P} = \mathsf{RP} = \mathsf{coRP}$). On the other hand, Kabanets and Impagliazzo [12] proved that if PIT belongs to P, then (i) there is a language in $\mathsf{NEXPTIME}$ that does not have polynomial size circuits, or (ii) the permanent is not computable by polynomial size arithmetic circuits. Both conclusions are major open problem in complexity theory. Hence, although it is quite plausible that PIT $\in \mathsf{P}$, it is difficult to prove.

Circuit evaluation problems can be also studied for other structures than polynomial rings, in particular non-commutative structures. For finite monoids, the circuit evaluation was studied in [6], where it was shown that for every non-solvable finite monoid the circuit evaluation problem is P-complete, whereas for every solvable finite monoid, the circuit evaluation problem belongs to the parallel complexity class $\mathsf{DET} \subseteq \mathsf{NC}^2$. Starting with [17] the circuit evaluation problem has been also studied for infinite finitely generated (f.g) monoids, in particular infinite f.g. groups. In this context, the input gates of the circuit are labelled with generators of the monoid, the internal gates compute the product of the two input gates, and it is asked whether the circuit evaluates to the identity element. In [17] and subsequent work, the circuit evaluation problem is also called the *compressed word problem (CWP)*. This is due to the fact that if one forgets the underlying monoid structure of a multiplicative circuit, the circuit simply evaluates to a word over the monoid generators labelling the input gates. This word can be of length exponential in the number of circuit gates, and the circuit can be seen as a compressed representation of this word. In this context, circuits are also known as *straight-line programs* and are intensively studied in the area of string compression [18].

Concerning the CWP, polynomial time algorithms have been developed for many important classes of groups, e.g., finite groups, f.g. nilpotent groups, f.g. free groups, graph groups (also known as right-angled Artin groups), and virtually special groups. The latter contain all Coxeter groups, one-relator groups with torsion, fully residually free groups, and fundamental groups of hyperbolic 3-manifolds; see [19]. For the important class of f.g. linear groups, i.e., f.g. groups of matrices over a field, the CWP reduces to PIT (over $\mathbb{Z}$ or $\mathbb{Z}_p$, depending on the characteristic of the field) and hence belongs to coRP [19]. Vice versa, in [19] it was shown that PIT over $\mathbb{Z}$ reduces to the CWP for the linear group $\mathsf{SL}_3(\mathbb{Z})$. This result indicates that derandomizing the CWP for a f.g. linear group will be in general very difficult.

In this paper, we further investigate the tight correspondence between commutative circuits over rings and non-commutative circuits over linear groups. In Sec. 6 we study the complexity of the CWP for f.g. nilpotent groups. It is known to be in P [19]. Here, we show that for every f.g. nilpotent group the CWP belongs to the parallel complexity class $\mathsf{DET} \subseteq \mathsf{NC}^2$, which is the class of all problems that are NC^1-reducible to the computation of the determinant of an integer matrix, see [8]. To the knowledge of the authors, f.g. nilpotent groups are the only examples of infinite groups for which the CWP belongs to NC. Even for free groups, the CWP is P-complete [17]. The main step of our proof for f.g. nilpotent groups is to show that for a torsion-free f.g. nilpotent group G the CWP belongs to the logspace counting class $\mathsf{C}_=\mathsf{L}$ (and is in fact $\mathsf{C}_=\mathsf{L}$-complete if $G \neq 1$). To show this, we use the fact that a f.g. torsion-free nilpotent group embeds into the group $\mathsf{UT}_d(\mathbb{Z})$ of d-dimensional unitriangular matrices over $\mathbb{Z}$ for some d. Then, we reduce the CWP for $\mathsf{UT}_d(\mathbb{Z})$ to the $\mathsf{C}_=\mathsf{L}$-complete problem, whether two additive circuits over the naturals evaluate to the same number. Let us mention that there are several $\mathsf{C}_=\mathsf{L}$-complete problems related to linear algebra [1].

We also study the CWP for the matrix group $\mathsf{UT}_d(\mathbb{Z})$ for the case that the dimension d is not fixed, i.e., part of the input (Sec. 7). In this case, the CWP turns out to be complete for the counting class $\mathsf{C}_=\mathsf{LogCFL}$, which is the LogCFL-analogue of $\mathsf{C}_=\mathsf{L}$.

Finally, in Sec. 8 we move from nilpotent groups to polycyclic groups. These are solvable groups, where every subgroup is finitely generated. By results of Maltsev, Auslander, and Swan these are exactly the solvable subgroups of $\mathsf{GL}_d(\mathbb{Z})$ for some d. We prove that polynomial identity testing for skew arithmetical circuits reduces to the CWP for a specific 2-generator polycyclic group of Hirsch length 3. A skew arithmetical circuit is an arithmetic circuit (as defined in the first paragraph of the introduction) such that for every multiplication gate, one of its input gates is an input gate of the circuit, i.e., a variable or a constant. These circuits exactly correspond to algebraic branching programs. Even for skew arithmetical circuits, no polynomial time algorithm is currently known (although the problem belongs to coRNC).

Full proofs can be found in the long version [14].

2 Arithmetical circuits

We use the standard notion of (division-free) arithmetical circuits. Let us fix a set $X = \{x_1, x_2, \ldots\}$ of variables. An *arithmetical circuit* is a triple $C = (V, S, \mathsf{rhs})$, where (i) V is a finite set of *gates*, (ii) $S \in V$ is the *output gate*, and (iii) for every gate A, $\mathsf{rhs}(A)$ (the *right-hand side of* A) is either a variable from X, one of the constants -1, 0, 1, or an expression of the form $B + C$ (then A is an addition gate) or $B \cdot C$ (then A is a multiplication gate), where B and C are gates. Moreover, there must exist a linear order $<$ on V such that $B < A$ whenever B occurs in $\mathsf{rhs}(A)$. A gate A with $\mathsf{rhs}(A) \in X \cup \{0, -1, 1\}$ is an *input gate*. Over a fixed ring $(R, +, \cdot)$ (which will be $(\mathbb{Z}, +, \cdot)$ in most cases) we can evaluate every gate $A \in V$ to a polynomial $\mathsf{val}_{\mathcal{C}}(A)$ with coefficients from R and variables from X (val stands for "value"). Moreover let $\mathsf{val}(\mathcal{C}) = \mathsf{val}_{\mathcal{C}}(S)$ be the polynomial to which $\mathcal{C}$ evaluates. Two arithmetical circuits $\mathcal{C}_1$ and $\mathcal{C}_2$ are equivalent over the ring $(R, +, \cdot)$ if $\mathsf{val}(\mathcal{C}_1) = \mathsf{val}(\mathcal{C}_2)$.

Fix an arithmetical circuit $\mathcal{C} = (V, S, \mathsf{rhs})$. We can view $\mathcal{C}$ as a directed acyclic graph (dag), where every node is labelled with a variable or a constant or an operator $+$, $\cdot$. If $\mathsf{rhs}(A) = B \circ C$ (for $\circ$ one of the operators), then there is an edge from B to A and C to A. The *depth* $\mathsf{depth}(A)$ (resp., *multiplication depth* $\mathsf{mdepth}(A)$) of the gate A is the maximal number of gates (resp., multiplication gates) along a path from an input gate to A. So, input gates have depth one and multiplication depth zero. The *depth* (resp., *multiplication depth*) of $\mathcal{C}$ is $\mathsf{depth}(\mathcal{C}) = \mathsf{depth}(S)$ (resp., $\mathsf{mdepth}(\mathcal{C}) = \mathsf{mdepth}(S)$). The *formal degree* $\mathsf{deg}(A)$ of a gate A is 1 if A is an input gate, $\max\{\mathsf{deg}(B), \mathsf{deg}(C)\}$ if $\mathsf{rhs}(A) = B + C$, and $\mathsf{deg}(B) + \mathsf{deg}(C)$ if $\mathsf{rhs}(A) = B \cdot C$. The formal degree of $\mathcal{C}$ is $\mathsf{deg}(\mathcal{C}) = \deg(S)$. A *positive circuit* is an arithmetical circuit without input gates labelled by the constant -1. An *addition circuit* is a positive circuit without multiplication gates. A *variable-free circuit* is a circuit without variables. It evaluates to an element of the underlying ring. A *skew circuit* is an arithmetical circuit such that for every multiplication gate A with $\mathsf{rhs}(A) = B \cdot C$, one of the gates B, C is an input gate.

In the rest of the paper we will also allow more complicated expressions in right-hand sides for gates. For instance, we may have a gate with $\mathsf{rhs}(A) = (B + C) \cdot (D + E)$. When writing down such a right-hand side, we implicitly assume that there are additional gates in the circuit, with (in our example) right hand sides $B + C$ and $D + E$, respectively. The proof of the following lemma uses standard ideas.

Lemma 1. *Given an arithmetical circuit $\mathcal{C}$ one can compute in logspace positive circuits $\mathcal{C}_1, \mathcal{C}_2$ such that* $\mathsf{val}(\mathcal{C}) = \mathsf{val}(\mathcal{C}_1) - \mathsf{val}(\mathcal{C}_2)$ *for every ring. Moreover, for $i \in \{1, 2\}$ we have* $\deg(\mathcal{C}_i) \le \deg(\mathcal{C})$, $\mathsf{depth}(\mathcal{C}_i) \le 2 \cdot \mathsf{depth}(\mathcal{C})$, *and* $\mathsf{mdepth}(\mathcal{C}_i) \le \mathsf{mdepth}(\mathcal{C})$.

Polynomial identity testing (PIT) for a ring R is the following computational problem: Given an arithmetical circuit $\mathcal{C}$ (with variables $x_1, \ldots, x_n$), does $\mathsf{val}(\mathcal{C}) = 0$ hold, i.e., does $\mathcal{C}$ evaluate to the zero-polynomial in $R[x_1, \ldots, x_n]$? It is an outstanding open problem in complexity theory, whether PIT for $\mathbb{Z}$ can be solved in polynomial time.

3 Complexity classes

The counting class $\#\mathsf{L}$ consists of all functions $f : \Sigma^* \to \mathbb{N}$ for which there is a logspace bounded nondeterministic Turing machine M such that for every $w \in \Sigma^*$,

$f(w)$ is the number of accepting computation paths of M on input x. The class $\mathsf{C_=L}$ contains all languages A for which there are two functions $f_1, f_2 \in \#\mathsf{L}$ such that for every $w \in \Sigma^*$, $w \in A$ if and only if $f_1(w) = f_2(w)$. The class $\mathsf{C_=L}$ is closed under logspace many-one reductions. The canonical $\mathsf{C_=L}$-complete problem is the following: The input consists of two dags G_1 and G_2 and vertices s_1, t_1 (in G_1) and s_2, t_2 (in G_2), and it is asked whether the number of paths from s_1 to t_1 in G_1 is equal to the number of paths from s_2 to t_2 in G_2. A reformulation of this problem is: Given two variable-free addition circuits $\mathcal{C}_1$ and $\mathcal{C}_2$, does $\mathsf{val}(\mathcal{C}_1) = \mathsf{val}(\mathcal{C}_2)$ hold?

We use standard definitions concerning circuit complexity, see e.g. [26]. In particular we will consider the class TC^0 of all problems that can be solved by a polynomial size circuit family of constant depth that uses NOT-gates and unbounded fan-in AND-gates, OR-gates, and majority-gates. For $\mathsf{DLOGTIME}$-uniform TC^0 it is required in addition that for binary coded gate numbers u and v, one can (i) compute the type of gate u in time $O(|u|)$ and (ii) check in time $O(|u|+|v|)$ whether u is an input gate for v. Note that the circuit for inputs of length n has at most $p(n)$ gates for a polynomial $p(n)$. Hence, the binary codings u and v have length $O(\log n)$, i.e., the above computations can be done in $\mathsf{DTIME}(\log n)$. This is the reason for using the term "$\mathsf{DLOGTIME}$-uniform". If majority gates are not allowed, we obtain the class ($\mathsf{DLOGTIME}$-uniform) AC^0. The class ($\mathsf{DLOGTIME}$-uniform) NC^1 is defined by ($\mathsf{DLOGTIME}$-uniform) polynomial size circuit families of logarithmic depth that use NOT-gates and fan-in-2 AND-gates and OR-gates. A language A is AC^0-reducible to languages $B_1, \ldots, B_k$ if A can be solved with a $\mathsf{DLOGTIME}$-uniform polynomial size circuit family of constant depth that uses NOT-gates and unbounded fan-in AND-gates, OR-gates, and B_i-gates ($1 \le i \le k$). Here, a B_i-gate (it is also called an oracle gate) receives an ordered tuple of inputs $x_1, x_2, \ldots, x_n$ and outputs 1 if and only if $x_1 x_2 \cdots x_n \in B_i$. Sometimes, also the term "uniform constant depth reducibility" is used for this type of reductions. In the same way, the weaker NC^1-reducibility can be defined. Here, one counts the depth of a B_i-gate with inputs $x_1, x_2, \ldots, x_n$ as $\log n$. The class DET contains all problems that are NC^1-reducible to the computation of the determinant of an integer matrix, see [8]. It is known that $\mathsf{C_=L} \subseteq \mathsf{DET} \subseteq \mathsf{NC}^2$, see e.g. [4, Sec.4].

An *NAuxPDA* is a nondeterministic Turing machine with an additional pushdown store. The class $\mathsf{LogCFL} \subseteq \mathsf{NC}^2$ is the class of all languages that can be accepted by a polynomial time bounded NAuxPDA whose work tape is logarithmically bounded (but the pushdown store is unbounded). If we assign to the input the number of accepting computation paths of such an NAuxPDA, we obtain the counting class $\#\mathsf{LogCFL}$. In [25] it is shown that a function $f : \{0,1\}^* \to \mathbb{N}$ belongs to $\#\mathsf{LogCFL}$ if and only if there exists a logspace-uniform family $(\mathcal{C}_n)_{n \ge 1}$ of positive arithmetic circuits such that $\mathcal{C}_n$ computes the mapping f restricted to $\{0,1\}^n$ and there is a polynomial $p(n)$ such that the formal degree of $\mathcal{C}_n$ is bounded by $p(n)$. The class $\mathsf{C_=LogCFL}$ contains all languages A for which there are two functions $f_1, f_2 \in \#\mathsf{LogCFL}$ such that for every $w \in \Sigma^*$, $w \in A$ if and only if $f_1(w) = f_2(w)$. We need the following lemma, whose proof is based on folklore ideas:

Lemma 2. *There is an NAuxPDA $\mathcal{P}$ that gets as input a positive variable-free arithmetic circuit $\mathcal{C}$ and such that the number of accepting computations of $\mathcal{P}$ on input $\mathcal{C}$ is* $\mathsf{val}(\mathcal{C})$. *Moreover, the running time is bounded polynomially in* $\mathsf{depth}(\mathcal{C}) \cdot \mathsf{deg}(\mathcal{C})$.

4 Matrices and groups

In this paper we are concerned with certain subclasses of *linear groups*. A group is linear if it is isomorphic to a subgroup of $\mathsf{GL}_d(F)$ (the group of all invertible $(d \times d)$-matrices over the field F) for some field F.

A (n-step) solvable group G is a group G, which has a a subnormal series $G = G_n \rhd G_{n-1} \rhd G_{n-2} \rhd \cdots \rhd G_1 \rhd G_0 = 1$ (i.e., G_i is a normal subgroup of G_{i+1} for all $0 \leq i \leq n-1$) such that every quotient G_{i+1}/G_i is abelian ($0 \leq i \leq n-1$). If every quotient G_{i+1}/G_i is cyclic, then G is called *polycyclic*. The number of $0 \leq i \leq n-1$ such that $G_{i+1}/G_i \cong \mathbb{Z}$ is called the *Hirsch length* of G; it does not depend on the chosen subnormal series. If $G_{i+1}/G_i \cong \mathbb{Z}$ for all $0 \leq i \leq n-1$ then G is called *strongly polycyclic*. A group is polycyclic if and only if it is solvable and every subgroup is finitely generated. Polycyclic groups are linear. Auslander and Swan [5,24] proved that the polycyclic groups are exactly the solvable groups of integer matrices.

For a group G its *lower central series* is the series $G = G_1 \rhd G_2 \rhd G_3 \rhd \cdots$ of subgroups, where $G_{i+1} = [G_i, G]$, which is the subgroup generated by all commutators $[g, h]$ with $g \in G_i$ and $h \in G$. Indeed, G_{i+1} is a normal subgroup of G_i. The group G is *nilpotent*, if its lower central series terminates after finitely many steps in the trivial group 1. Every f.g. nilpotent group is polycyclic.

Let G be a f.g. group and let G be finitely generated as a group by Σ. Then, as a monoid G is finitely generated by $\Sigma \cup \Sigma^{-1}$ (where $\Sigma^{-1} = \{a^{-1} \mid a \in \Sigma\}$ is a disjoint copy of Σ and a^{-1} stands for the inverse of the generator $a \in \Sigma$). Recall that the *word problem* for G is the following computational problem: Given a string $w \in (\Sigma \cup \Sigma^{-1})^*$, does w evaluate to the identity of G. Kharlampovich proved that there exist finitely presented 3-step solvable groups with an undecidable word problem. On the other hand, for every f.g. linear group the word problem can be solved in deterministic logarithmic space by results of Lipton and Zalcstein [16] and Simon [23]. This applies in particular to polycyclic groups. Robinson proved in his thesis that the word problem for a polycyclic group belongs to TC^0 [21], but his circuits are not uniform. Waack considered in [27] arbitrary f.g. solvable linear groups (which include the polycyclic groups) and proved that their word problems belong to logspace-uniform NC^1. In [14] we combine Waack's technique with the famous division breakthrough result by Hesse, Allender, and Barrington [9] to show that for every f.g. solvable linear group the word problem belongs to $\mathsf{DLOGTIME}$-uniform TC^0.

5 Straight-line programs and the compressed word problem

A straight-line program (briefly, SLP) is basically a multiplicative circuit over a monoid. We define an SLP over the finite alphabet Σ as a triple $\mathcal{G} = (V, S, \mathsf{rhs})$, where V is a finite set of variables (or gates), $S \in V$ is the start variable (or output gate), and rhs maps every variable to a right-hand side $\mathsf{rhs}(A)$, which is either a symbol $a \in \Sigma$, or of the form BC, where $B, C \in V$. As for arithmetical circuits we require that there is a linear order $<$ on V such that $B < A$, whenever B occurs in $\mathsf{rhs}(A)$. The terminology "(start) variable" (instead of "(output) gate") comes from the fact that an SLP is quite often defined as a context-free grammar that produces a single string over Σ. This string is defined in the obvious way by iteratively replacing variables by the corresponding right-hand sides, starting with the start variable. We denote this string with $\mathsf{val}(\mathcal{G})$. The unique string over Σ, derived from the variable $A \in V$, is denoted with $\mathsf{val}_{\mathcal{G}}(A)$. We will also allow more general right-hand sides from $(V \cup \Sigma)^*$, but by introducing new variables we can always obtain an equivalent SLP in the above form.

If we have a monoid M, which is finitely generated by the set Σ, then there exists a canonical monoid homomorphism $h : \Sigma^* \to M$. Then, an SLP $\mathcal{G}$ over the alphabet Σ can be evaluated over the monoid M, which yields the monoid element $h(\mathsf{val}(\mathcal{G}))$. In this paper, we are only interested in the case that the monoid M is a f.g. group G. Let G be finitely generated as a group by Σ. An SLP over the alphabet $\Sigma \cup \Sigma^{-1}$ is also called an SLP over the group G. In this case, we will quite often identify the string $\mathsf{val}(\mathcal{G}) \in (\Sigma \cup \Sigma^{-1})^*$ with the group element $g \in G$ to which it evaluates. We will briefly write "$\mathsf{val}(\mathcal{G}) = g$ in G" in this situation.

The main computational problem we are interested in is the *compressed word problem* for a f.g. group G (with a finite generating set Σ), briefly $\mathsf{CWP}(G)$. The input for this problem is an SLP $\mathcal{G}$ over the alphabet $\Sigma \cup \Sigma^{-1}$, and it is asked whether $\mathsf{val}(\mathcal{G}) = 1$ in G (where of course 1 denotes the group identity). The term "compressed word problem" comes from the fact that this problem can be seen as a succinct version of the classical word problem for G, where the input is an explicitly given string $w \in (\Sigma \cup \Sigma^{-1})^*$ instead of an SLP-compressed string.

The compressed word problem is related to the classical word problem. For instance, the classical word problem for a f.g. subgroup of the automorphism group of a group G can be reduced to the compressed word problem for G, and similar results are known for certain group extensions, see [19] for more details. There are several important classes of groups, for which the compressed word problem can be solved in polynomial time, and for finitely generated linear groups the compressed word problem belongs to co-randomized polynomial time, see the introduction. In [6] the parallel complexity of the compressed word problem (there, called the circuit evaluation problem) for finite groups was studied, and the following result was shown:

Theorem 1 ([6]). *Let G be a finite group. If G is solvable, then $\mathsf{CWP}(G)$ belongs to the class $\mathsf{DET} \subseteq \mathsf{NC}^2$. If G is not solvable, then $\mathsf{CWP}(G)$ is P-complete.*

6 CWP for finitely generated nilpotent groups

In [19] it was shown that the compressed word problem for a finitely generated nilpotent group can be solved in polynomial time. The main result of this section is:

Theorem 2. *Let $G \neq 1$ be a f.g. torsion-free nilpotent group. Then $\mathsf{CWP}(G)$ is complete for the class $\mathsf{C}_=\mathsf{L}$.*

For the lower bound let G be a non-trivial f.g. torsion-free nilpotent group. Since $G \neq 1$, G contains $\mathbb{Z}$. Hence, it suffices to prove the following:

Lemma 3. *$\mathsf{CWP}(\mathbb{Z})$ is hard for $\mathsf{C}_=\mathsf{L}$.*

Proof. An SLP $\mathcal{G}$ over the generator 1 of $\mathbb{Z}$ and its inverse -1 is nothing else than a variable-free arithmetical circuit $\mathcal{C}$ without multiplication gates. Using Lemma 1 we can construct in logspace two addition circuits $\mathcal{C}_1$ and $\mathcal{C}_2$ such that $\mathsf{val}(\mathcal{C}) = 0$ if and only if $\mathsf{val}(\mathcal{C}_1) = \mathsf{val}(\mathcal{C}_2)$. Checking the latter is complete for $\mathsf{C_=L}$ as remarked in Sec. 3. $\square$

For the upper bound in Thm. 2, we use the fact that every torsion-free f.g. nilpotent group can be represented by unitriangluar integer matrices. Let A be a $(d \times d)$-matrix over $\mathbb{Z}$. With $A[i,j]$ we denote the entry of A in row i and column j. The matrix A is *triangular* if $A[i,j] = 0$ whenever $i > j$, i.e., all entries below the main diagonal are 0. A *unitriangular matrix* is a triangular matrix A such that $A[i,i] = 1$ for all $1 \le i \le d$, i.e., all entries on the main diagonal are 1. We denote the set of unitriangular $(d \times d)$-matrices over $\mathbb{Z}$ with $\mathsf{UT}_d(\mathbb{Z})$. This is a group with respect to matrix multiplication. Let $1 \le i < j \le d$. With $T_{i,j}$ we denote the matrix from $\mathsf{UT}_d(\mathbb{Z})$ such that $T_{i,j}[i,j] = 1$ and $T_{i,j}[k,l] = 0$ for all k,l with $1 \le k < l \le d$ and $(k,l) \neq (i,j)$. The notation $T_{i,j}$ does not specify the dimension d of the matrix, but the dimension will be always clear from the context. The group $\mathsf{UT}_d(\mathbb{Z})$ is generated by the finite set $\Gamma_d = \{T_{i,i+1} \mid 1 \le i < d\}$, see e.g. [7]. For every torsion-free f.g. nilpotent group G there exists some $d \ge 1$ such that $G \le \mathsf{UT}_d(\mathbb{Z})$ [13, Thm. 17.2.5]. Hence, the upper bound in Thm. 2 follows from:

Lemma 4. *For every $d \ge 1$, $\mathsf{CWP}(\mathsf{UT}_d(\mathbb{Z}))$ belongs to $\mathsf{C_=L}$.*

For the rest of this section let us fix a number $d \ge 1$ and consider the unitriangluar matrix group $\mathsf{UT}_d(\mathbb{Z})$. Consider an SLP $\mathcal{G} = (V, S, \mathsf{rhs})$ over the alphabet $\Gamma_d \cup \Gamma_d^{-1}$, where Γ_d is the finite generating set of $\mathsf{UT}_d(\mathbb{Z})$ from Sec. 4. Note that for every variable $A \in V$, $\mathsf{val}_{\mathcal{G}}(A)$ is a word over the alphabet $\Gamma_d \cup \Gamma_d^{-1}$. We identify in the following this word with the matrix to which it evaluates. Thus, $\mathsf{val}_{\mathcal{G}}(A) \in \mathsf{UT}_d(\mathbb{Z})$.

Assume we have given an arithmetical circuit $\mathcal{C}$. A partition $\biguplus_{i=1}^{m} V_i$ of the set of all multiplication gates of $\mathcal{C}$ is called *structure-preserving* if for all multiplication gates u, v of $\mathcal{C}$ the following holds: If there is a non-empty path from u to v in (the dag corresponding to) $\mathcal{C}$ then there exist $1 \le i < j \le d$ such that $u \in V_i$ and $v \in V_j$. In a first step, we transform our SLP $\mathcal{G}$ in logspace into a variable-free arithmetical circuit $\mathcal{C}$ of multiplication depth at most d such that $\mathcal{G}$ evaluates to the identity matrix if and only if $\mathcal{C}$ evaluates to 0. Moreover, we also compute a structure-preserving partition of the multiplication gates of $\mathcal{C}$. This partition will be needed for the further computations. The degree bound in the following lemma will be needed in Sec. 7.

Lemma 5. *From the SLP $\mathcal{G} = (V, S, \mathsf{rhs})$ we can compute in logspace a variable-free arithmetical circuit $\mathcal{C}$ with $\mathsf{mdepth}(\mathcal{C}) \le d$ and $\deg(\mathcal{C}) \le 2(d-1)$, such that $\mathsf{val}(\mathcal{G}) = \mathsf{Id}_d$ if and only if $\mathsf{val}(\mathcal{C}) = 0$. In addition we can compute in logspace a structure-preserving partition $\biguplus_{i=1}^{d} V_i$ of the set of all multiplication gates of $\mathcal{C}$.*

Proof. The set of gates of $\mathcal{C}$ is $W = \{A_{i,j} \mid A \in V, 1 \le i < j \le d\} \uplus \{T\}$, where T is the output gate. The idea is simple: Gate $A_{i,j}$ will evaluate to the matrix entry $\mathsf{val}_{\mathcal{G}}(A)[i,j]$. To achieve this, we define the right-hand side mapping of the circuit $\mathcal{G}$ (which we denote again with rhs) as follows: If $\mathsf{rhs}(A) = M \in \Gamma_d \cup \Gamma_d^{-1}$, then $\mathsf{rhs}(A_{i,j}) = M[i,j] \in \{-1, 0, 1\}$, and if $\mathsf{rhs}(A) = BC$, then $\mathsf{rhs}(A_{i,j}) = B_{i,j} + C_{i,j} + \sum_{i<k<j} B_{i,k} \cdot C_{k,j}$ (which is the rule for matrix multiplication taking into account

that all matrices are unitriangular). Finally, we set $\mathsf{rhs}(T) = \sum_{1 \le i < j \le d} S_{i,j}^2$. Then, $\mathsf{val}(\mathcal{C}) = 0$ iff $\mathsf{val}_{\mathcal{G}}(S)[i,j] = 0$ for all $1 \le i < j \le d$ iff $\mathsf{val}(\mathcal{G})$ is the identity matrix.

Concerning the multiplication depth, note that the multiplication depth of the gate $A_{i,j}$ is bounded by $j - i$: The only multiplications in $\mathsf{rhs}(A_{i,j})$ are of the form $B_{i,k}C_{k,j}$ (and these multiplications are not nested). Hence, by induction, the multiplication depth of $A_{i,j}$ is bounded by $1 + \max\{k - i, j - k \mid i < k < j\} = j - i$. It follows that every gate $S_{i,j}$ has multiplication depth at most $d - 1$, which implies that the output gate T has multiplication depth at most d. Similarly, it can be shown by induction that $\mathsf{deg}(A_{i,j}) \le j - i$. Hence, $\mathsf{deg}(A_{i,j}) \le d - 1$ for all $1 \le i < j \le d$, which implies that the formal degree of the circuit is bounded by $2(d-1)$.

The structure-preserving partition $\biguplus_{i=1}^{d} V_i$ of the set of all multiplication gates of $\mathcal{C}$ can be defined as follows: All gates corresponding to multiplications $B_{i,k} \cdot C_{k,j}$ in $\mathsf{rhs}(A_{i,j})$ are put into the set V_{j-i}. Finally, all gates corresponding to multiplications $S_{i,j}^2$ in $\mathsf{rhs}(T)$ are put into V_d. It is obvious that this partition is structure-preserving. □

In a second step we apply Lemma 1 and construct from the above circuit $\mathcal{C}$ two variable-free positive circuits $\mathcal{C}_1$ and $\mathcal{C}_2$, both having multiplication depth at most d such that $\mathsf{val}(\mathcal{C}) = \mathsf{val}(\mathcal{C}_1) - \mathsf{val}(\mathcal{C}_2)$. Hence, our input SLP $\mathcal{G}$ evaluates to the indentity matrix if and only if $\mathsf{val}(\mathcal{C}_1) = \mathsf{val}(\mathcal{C}_2)$. Moreover, using the construction from Lemma 1 it is straightforward to compute in logspace a structure-preserving partition $\biguplus_{i=1}^{d} V_{k,i}$ of the the set of all multiplication gates of $\mathcal{C}_k$ ($k \in \{1,2\}$).

The following lemma concludes the proof that $\mathsf{CWP}(\mathsf{UT}_d(\mathbb{Z}))$ belongs to $\mathsf{C_{=}L}$. For the proof one eliminates in a single phase all multiplication gates in a layer. This can be achieved by a logspace reduction, and since the total number of layers is constant, the whole elimination procedure works in logspace.

Lemma 6. *Let d be constant. From a given variable-free positive circuit $\mathcal{C}$ of multiplication depth d together with a structure-preserving partition $\biguplus_{i=1}^{d} V_i$ of the set of all multiplication gates of $\mathcal{C}$, we can compute in logarithmic space a variable-free addition circuit $\mathcal{D}$ such that* $\mathsf{val}(\mathcal{C}) = \mathsf{val}(\mathcal{D})$.

So far, we have restricted to *torsion-free* f.g. nilpotent groups. For general f.g. nilpotent groups, we use the fact that every f.g. nilpotent group contains a torsion-free normal f.g. nilpotent subgroup of finite index [13, Thm. 17.2.2], in order to show that the compressed word problem for every f.g. nilpotent group belongs to the complexity class DET: To do this we need the following result. For the proof one can adopt the proof of [19, Thm.4.4], where the statement is shown for polynomial time many-one reducibility instead of AC^0-reducibility.

Theorem 3. *Let G be a finitely generated group. For every normal subgroup H of G with a finite index,* $\mathsf{CWP}(G)$ *is* AC^0*-reducible to* $\mathsf{CWP}(H)$ *and* $\mathsf{CWP}(G/H)$.

We can now show:

Theorem 4. *For every f.g. nilpotent group, the compressed word problem is in* DET.

Proof. Let G be a f.g. nilpotent group. If G is finite, then the result follows from Thm. 1 (every nilpotent group is solvable). If G is infinite, then G has a f.g. torsion-free normal subgroup H of finite index [13, Thm. 17.2.2]. Subgroups and quotients of nilpotent groups are nilpotent too [22, Chapter 5], hence H and G/H are nilpotent; moreover H is finitely generated. By Thm. 2, $\mathsf{CWP}(H)$ belongs to $\mathsf{C_{=}L} \subseteq \mathsf{DET}$. Moreover, by Thm. 1, $\mathsf{CWP}(G/H) \in \mathsf{DET}$ as well. Finally, Thm. 3 implies $\mathsf{CWP}(G) \in \mathsf{DET}$. $\square$

Actually, Thm. 4 can be slightly extended to groups that are (f.g. nilpotent)-by-(finite solvable) (i.e., groups that have a normal subgroup, which is f.g. nilpotent, and where the quotient is finite solvable. This follows from Thm. 3 and the fact that the compressed word problem for a finite solvable group belongs to DET (Thm. 1).

7 The uniform CWP for unitriangular groups

For Lemma 4 it is crucial that the dimension d is a constant. In this section, we consider a uniform variant of the compressed word problem for $\mathsf{UT}_d(\mathbb{Z})$. We denote this problem with $\mathsf{CWP}(\mathsf{UT}_*(\mathbb{Z}))$. The input consists of a unary encoded number d and an SLP, whose terminal symbols are generators of $\mathsf{UT}_d(\mathbb{Z})$ or their inverses. Alternatively, we can assume that the terminal symbols are arbitrary matrices from $\mathsf{UT}_d(\mathbb{Z})$ with binary encoded entries (given such a matrix M, it is easy to construct an SLP over the generator matrices that produces M). The question is whether the SLP evaluates to the identity matrix. We show that this problem is complete for the complexity class $\mathsf{C_{=}LogCFL}$.

Theorem 5. *The problem* $\mathsf{CWP}(\mathsf{UT}_*(\mathbb{Z}))$ *is complete for* $\mathsf{C_{=}LogCFL}$.

Proof. We start with the upper bound. Consider an SLP $\mathcal{G}$, whose terminal symbols are generators of $\mathsf{UT}_d(\mathbb{Z})$ or their inverses. The dimension d is clearly bounded by the input size. Consider the variable-free arithmetic circuit $\mathcal{C}$ constructed from $\mathcal{G}$ in Lemma 5 and let $\mathcal{C}_1$ and $\mathcal{C}_2$ be the two variable-free positive arithmetic circuits obtained from $\mathcal{C}$ using Lemma 1. Then $\mathcal{G}$ evaluates to the identity matrix if and only if $\mathsf{val}(\mathcal{C}_1) = \mathsf{val}(\mathcal{C}_2)$. Moreover, the formal degrees $\deg(\mathcal{C}_1)$ and $\deg(\mathcal{C}_2)$ are bounded by $2(d-1)$, i.e., polynomially bounded in the input length. Finally, we compose a logspace machine that computes from the input SLP $\mathcal{G}$ the circuit $\mathcal{C}_i$ with the NAuxPDA from Lemma 2 to get an NAuxPDA $\mathcal{P}_i$ such that the number of accepting computation paths of $\mathcal{P}_i$ on input $\mathcal{G}$ is exactly $\mathsf{val}(\mathcal{C}_i)$. Moreover, the running time of $\mathcal{P}_i$ on input $\mathcal{G}$ is bounded polynomially in $(2d-1) \cdot \mathsf{depth}(\mathcal{C}_i) \in O(d \cdot |\mathcal{G}|)$.

Let us now show that $\mathsf{CWP}(\mathsf{UT}_*(\mathbb{Z}))$ is hard for $\mathsf{C_{=}LogCFL}$. Let $(\mathcal{C}_{1,n})_{n \geq 0}$ and $(\mathcal{C}_{2,n})_{n \geq 0}$ be two logspace-uniform families of positive arithmetical circuits of polynomially bounded size and formal degree. Let $w = a_1 a_2 \cdots a_n \in \{0,1\}^n$ be an input for the circuits $\mathcal{C}_{1,n}$ and $\mathcal{C}_{2,n}$. Let $\mathcal{C}_i$ be the variable-free positive arithmetical circuit obtained from $\mathcal{C}_{i,n}$ by replacing every x_j-labelled input gate by $a_j \in \{0,1\}$. By [3, Lemma 3.2] we can assume that every gate of $\mathcal{C}_i$ is labelled by its formal degree. By adding if necessary additional multiplication gates, where one input is set to 1, we can assume that $\mathcal{C}_1$ and $\mathcal{C}_2$ have the same formal degree $d \leq p(n)$ for a polynomial p. Analogously, we can assume that if A is an addition gate in $\mathcal{C}_1$ or $\mathcal{C}_2$ with right-hand side $B+C$, then $\deg(B) = \deg(C) = \deg(A)$. All these preprocessing steps can be carried out in logarithmic space.

We will construct in logarithmic space an SLP $\mathcal{G}$ over the alphabet $\Gamma_{d+1} \cup \Gamma_{d+1}^{-1}$, where Γ_{d+1} is our canonical generating set for the matrix group $\mathsf{UT}_{d+1}(\mathbb{Z})$, such that $\mathcal{G}$ evaluates to the identity matrix if and only if $\mathsf{val}(\mathcal{C}_1) = \mathsf{val}(\mathcal{C}_2)$. Let v_i be the output value of $\mathcal{C}_i$. We first construct in logspace an SLP $\mathcal{G}_1$ that evaluates to the matrix $T_{1,d}^{v_1}$. In the same way we can construct in logspace a second SLP $\mathcal{G}_2$ that evaluates to $T_{1,d}^{-v_2}$. Then, by concatenating the two SLPs $\mathcal{G}_1$ and $\mathcal{G}_2$ we obtain the desired SLP.

The variables of $\mathcal{G}_1$ are $A_{i,j}^b$, where A is a gate of $\mathcal{C}_1$, $b \in \{-1, 1\}$, and $1 \leq i < j \leq d$ such that $j - i$ is the formal degree of A. The SLP $\mathcal{G}_1$ will be constructed in such a way that $\mathsf{val}_{\mathcal{G}_1}(A_{i,j}^b) = T_{i,j}^{b \cdot v}$, where $v = \mathsf{val}_{\mathcal{C}_1}(A)$. If $\mathsf{rhs}_{\mathcal{C}_1}(A) = 0$, then we set $\mathsf{rhs}_{\mathcal{G}_1}(A_{i,j}^b) = \mathsf{Id}$ and if $\mathsf{rhs}_{\mathcal{C}_1}(A) = 1$, then we set $\mathsf{rhs}_{\mathcal{G}_1}(A_{i,j}^b) = T_{i,j}^b$. If $\mathsf{rhs}_{\mathcal{C}_1}(A) = B + C$, then we set $\mathsf{rhs}_{\mathcal{G}_1}(A_{i,j}^b) = B_{i,j}^b C_{i,j}^b$. Correctness follows immediately by induction. Note that $\deg(B) = \deg(C) = \deg(A) = j - i$, which implies that the gates $B_{i,j}^b$ and $C_{i,j}^b$ exist. Finally, if $\mathsf{rhs}_{\mathcal{C}_1}(A) = B \cdot C$, then we set $\mathsf{rhs}_{\mathcal{G}_1}(A_{i,j}^1) = B_{i,k}^{-1} C_{k,j}^{-1} B_{i,k}^1 C_{k,j}^1$ and $\mathsf{rhs}_{\mathcal{G}_1}(A_{i,j}^{-1}) = C_{k,j}^{-1} B_{i,k}^{-1} C_{k,j}^1 B_{i,k}^1$, where k is such that $\deg(B) = k - i$ and $\deg(B) = j - k$. Such a k exists since $j - i = \deg(A) = \deg(B) + \deg(C)$. Correctness follows by induction and the simple fact that $T_{i,j}^{-a}, T_{j,k}^{-b} T_{i,j}^a, T_{j,k}^b = T_{i,k}^{ab}$ for all $a, b \in \mathbb{Z}$ and $1 \leq i < j < k \leq d$; see [20]. $\square$

8 CWP for polycyclic groups

In this section we look at the compressed word problem for polycyclic groups. Since every polycyclic group is f.g. linear, the compressed word problem for a polycyclic group can be reduced to PIT. Here, we show a lower bound: There is a polycyclic group G such that PIT for skew arithmetical circuits can be reduced to $\mathsf{CWP}(G)$. In this context, it is interesting to note that PIT for arbitrary circuits can be reduced to the compressed word problem to the linear (but not polycyclic) group $\mathsf{SL}_3(\mathbb{Z})$ [19, Thm.4.16].

Let us start with a specific example of a polycyclic group. Consider the two matrices

$$g_a = \begin{pmatrix} a & 0 \\ 0 & 1 \end{pmatrix} \text{ and } h = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}, \tag{1}$$

where $a \in \mathbb{R}$, $a \geq 2$. Let $G_a = \langle g_a, h \rangle \leq \mathsf{GL}_2(\mathbb{R})$. Let us remark that, for instance, the group G_2 is not polycyclic, see e.g. [28, p. 56]. On the other hand, we have:

Proposition 1. *The group $G = G_{1+\sqrt{2}}$ is polycyclic.*

The main result of this section is:

Theorem 6. *Let $a \geq 2$. Polynomial identity testing for skew arithmetical circuits is logspace-reducible to the compressed word problem for the group G_a.*

In particular, there exist polycyclic groups for which the compressed word problem is at least as hard as polynomial identity testing for skew circuits. Recall that it is not known, whether there exists a polynomial time algorithm for polynomial identity testing restricted to skew arithmetical circuits.

For the proof of Thm. 6, we use the following result from [2] (see the proof of [2, Prop. 2.2], where the result is shown for $a = 2$, but the proof works for any $a \geq 2$):

Lemma 7. *Let $\mathcal{C}$ be an arithmetical circuit of size n with variables $x_1, \ldots, x_m$ and let $p(x_1, \ldots, x_m) = \mathsf{val}(\mathcal{C})$. Let $a \geq 2$ be a real number. Then $p(x_1, \ldots, x_n)$ is the zero-polynomial if and only if $p(\alpha_1, \ldots, \alpha_n) = 0$, where $\alpha_i = a^{2^{i \cdot n^2}}$ for $1 \leq i \leq m$.*

Proof of Thereom 6. Let us fix a skew arithmetical circuit $\mathcal{C}$ of size n with m variables $x_1, \ldots, x_m$. We will define an SLP $\mathcal{G}$ over the alphabet $\{g_a, g_a^{-1}, h, h^{-1}\}$ such that $\mathsf{val}(\mathcal{G}) = \mathsf{Id}$ in G_a if and only if $\mathsf{val}(\mathcal{C}) = 0$. First of all, using iterated squaring, we can construct an SLP $\mathcal{H}$ with variables $A_1, A_1^{-1} \ldots, A_m, A_m^{-1}$ (and some other auxiliary variables) such that

$$\mathsf{val}_{\mathcal{H}}(A_i) = g_a^{2^{i \cdot n^2}} = \begin{pmatrix} \alpha_i & 0 \\ 0 & 1 \end{pmatrix} \quad \text{and} \quad \mathsf{val}_{\mathcal{H}}(A_i^{-1}) = g_a^{-2^{i \cdot n^2}} = \begin{pmatrix} \alpha_i^{-1} & 0 \\ 0 & 1 \end{pmatrix}.$$

We now construct the SLP $\mathcal{G}$ as follows: The set of variables of $\mathcal{G}$ consists of the gates of $\mathcal{C}$ and the variables of $\mathcal{H}$. We copy the right-hand sides from $\mathcal{H}$ and define the right-hand side for a gate A of $\mathcal{C}$ as follows: (i) $\mathsf{rhs}_{\mathcal{G}}(A) = h^n$ if $\mathsf{rhs}_{\mathcal{C}}(A) = n \in \{0, -1, 1\}$, (ii) $\mathsf{rhs}_{\mathcal{G}}(A) = BC$ if $\mathsf{rhs}_{\mathcal{C}}(A) = B + C$, and (iii) $\mathsf{rhs}_{\mathcal{G}}(A) = A_i B A_i^{-1}$ if $\mathsf{rhs}_{\mathcal{C}}(A) = x_i \cdot B$.

A straightforward induction shows that for every gate A of $\mathcal{C}$ we have the following, where we denote for better readability the polynomial $\mathsf{val}_{\mathcal{C}}(A)$ to which gate A evaluates with p_A:

$$\mathsf{val}_{\mathcal{G}}(A) = \begin{pmatrix} 1 & p_A(\alpha_1, \ldots, \alpha_n) \\ 0 & 1 \end{pmatrix}$$

We finally take the output gate S of the skew circuit $\mathcal{C}$ as the start variable of $\mathcal{G}$. Then, $\mathsf{val}(\mathcal{G})$ yields the identity matrix in the group G_a if and only if $p_S(\alpha_1, \ldots, \alpha_n) = 0$. By Lemma 7 this is equivalent to $\mathsf{val}(\mathcal{C}) = p_S(x_1, \ldots, x_n) = 0$. □

Actually, we can carry out the above reduction for a class of arithmetical circuits that is slightly larger than the class of skew arithmetical circuits. Let us define a *powerful skew circuit* as an arithmetical circuit, where for every multiplication gate A, $\mathsf{rhs}(A)$ is of the form $x_i^e \cdot B$, where $e \geq 0$ is a binary coded number. Such a circuit can be converted into an ordinary arithmetical circuit, which, however is no longer skew. To extend the reduction from the proof of Thereom 6 to powerful skew circuits, we set for a gate A with $\mathsf{rhs}_{\mathcal{C}}(A) = x_i^e \cdot B$: $\mathsf{rhs}_{\mathcal{G}}(A) = A_i^e B A_i^{-e}$. The powers A_i^e and A_i^{-e} can be defined using additional multiplication gates. In our recent paper [15], we introduced powerful skew circuits, and proved that for this class, PIT belongs to coRNC. We applied this result to the compressed word problem for wreath products.

Let us look again at the group $G = G_{1+\sqrt{2}}$ from Prop. 1. A closer inspection (see [14]) shows that $[G, G] \cong \mathbb{Z} \times \mathbb{Z}$ and $G/[G, G] \cong \mathbb{Z} \times \mathbb{Z}_2$. Hence, G has a subnormal series of the form $G \rhd H \rhd \mathbb{Z} \times \mathbb{Z} \rhd \mathbb{Z} \rhd 1$, where H has index 2 in G and $H/(\mathbb{Z} \times \mathbb{Z}) \cong \mathbb{Z}$. The group H is strongly polycyclic and has Hirsch length 3. By Thm. 3 we obtain:

Corollary 1. *There is a strongly polycyclic group H of Hirsch length 3 such that polynomial identity testing for skew circuits is polynomial time reducible to $\mathsf{CWP}(H)$.*

References

1. E. Allender, R. Beals, and M. Ogihara. The complexity of matrix rank and feasible systems of linear equations. *Comput. Complex.*, 8(2):99–126, 1999.

2. E. Allender, P. Bürgisser, J. Kjeldgaard-Pedersen, and P. B. Miltersen. On the complexity of numerical analysis. *SIAM J. Comput.*, 38(5):1987–2006, 2009.
3. E. Allender, J. Jiao, M. Mahajan, and V. Vinay. Non-commutative arithmetic circuits: Depth reduction and size lower bounds. *Theor. Comput. Sci.*, 209(1-2):47–86, 1998.
4. C. Àlvarez and B. Jenner. A very hard log-space counting class. *Theor. Comput. Sci.*, 107(1):3–30, 1993.
5. L. Auslander. On a problem of Philip Hall. *Annals of Mathematics*, 86(2):112–116, 1967.
6. M. Beaudry, P. McKenzie, P. Péladeau, and D. Thérien. Finite monoids: From word to circuit evaluation. *SIAM J. Comput.*, 26(1):138–152, 1997.
7. D. K. Biss and S. Dasgupta. A presentation for the unipotent group over rings with identity. *Journal of Algebra*, 237(2):691–707, 2001.
8. S. A. Cook. A taxonomy of problems with fast parallel algorithms. *Inform. Control*, 64:2–22, 1985.
9. W. Hesse, E. Allender, and D. A. M. Barrington. Uniform constant-depth threshold circuits for division and iterated multiplication. *J. Comput. System Sci.*, 65:695–716, 2002.
10. O. H. Ibarra and S. Moran. Probabilistic algorithms for deciding equivalence of straight-line programs. *J. Assoc. Comput. Mach.*, 30(1):217–228, 1983.
11. R. Impagliazzo and A. Wigderson. P = BPP if E requires exponential circuits: Derandomizing the XOR lemma. In *Proc. STOC 1997*, 220–229. ACM Press, 1997.
12. V. Kabanets and R. Impagliazzo. Derandomizing polynomial identity tests means proving circuit lower bounds. *Comput. Complex.*, 13(1-2):1–46, 2004.
13. M. I. Kargapolov and J. I. Merzljakov. *Fundamentals of the Theory of Groups*. Springer, 1979.
14. D. König M. Lohrey. Evaluating matrix circuits. arXiv.org, 2015.
15. D. König M. Lohrey. Parallel identity testing for algebraic branching programs with big powers and applications. arXiv.org, 2015.
16. R. J. Lipton and Y. Zalcstein. Word problems solvable in logspace. *J. Assoc. Comput. Mach.*, 24(3):522–526, 1977.
17. M. Lohrey. Word problems and membership problems on compressed words. *SIAM J. Comput.*, 35(5):1210 – 1240, 2006.
18. M. Lohrey. Algorithmics on SLP-compressed strings: A survey. *Groups Complexity Cryptology*, 4(2):241–299, 2012.
19. M. Lohrey. *The Compressed Word Problem for Groups*. SpringerBriefs in Mathematics. Springer, 2014.
20. M. Lohrey. Rational subsets of unitriangular groups. *International Journal of Algebra and Computation*, 2015. DOI: 10.1142/S0218196715400068.
21. D. Robinson. *Parallel Algorithms for Group Word Problems*. PhD thesis, UCSD, 1993.
22. J. J. Rotman. *An Introduction to the Theory of Groups (fourth edition)*. Springer, 1995.
23. H.-U. Simon. Word problems for groups and contextfree recognition. In *Proceedings of Fundamentals of Computation Theory, FCT 1979*, pages 417–422. Akademie-Verlag, 1979.
24. R. Swan. Representations of polycyclic groups. *Proc. Am. Math. Soc.*, 18:573–574, 1967.
25. V. Vinay. Counting auxiliary pushdown automata and semi-unbounded arithmetic circuits. In *Proc. Structure in Complexity Theory Conference*, 270–284. IEEE Computer Society, 1991.
26. H. Vollmer. *Introduction to Circuit Complexity*. Springer, 1999.
27. S. Waack. On the parallel complexity of linear groups. *ITA*, 25(4):265–281, 1991.
28. B. A. F. Wehrfritz. *Infinite Linear Groups*. Springer, 1977.