Efficient computation in groups via compression

Markus Lohrey[1] and Saul Schleimer[2]

[1] Universität Stuttgart, FMI, Germany
[2] School of Mathematics and Statistics, Rutgers University, Mathematics Department, New Brunswick, New Jersey, USA
lohrey@informatik.uni-stuttgart.de, saulsch@math.rutgers.edu

Abstract. We study the *compressed word problem*: a variant of the word problem for finitely generated groups where the input word is given by a context-free grammar that generates exactly one string. We show that finite extensions and free products preserve the complexity of the compressed word problem. Also, the compressed word problem for a graph group can be solved in polynomial time. These results allow us to obtain new upper complexity bounds for the word problem for certain automorphism groups and group extensions.

1 Introduction

The *word problem for finitely generated groups* is a fundamental computational problem linking group theory, topology, mathematical logic, and computer science. For a group G, finitely generated by Σ, it is asked whether a word over Σ and the inverses of Σ represents the identity element of G. The word problem was introduced in the pioneering work of Dehn from 1910 in relation with topological questions. It took about 45 years until Novikov and later independently Boone proved the existence of a finitely presented group with an undecidable word problem, see [22, 31] for references. Despite this negative result, many natural classes of groups with decidable word problems were found. Prominent examples are finitely generated linear groups, automatic groups [12], and one-relator groups. With the advent of computational complexity theory the complexity of word problems became an active research area. For instance, it was shown that for a finitely generated linear group the word problem can be solved in logarithmic space [20, 30], that automatic groups have quadratic time word problems [12], and that the word problem for a one-relator group is primitive recursive [5].

Group theoretic operations, which preserve (or moderately increase) the complexity of the word problem are useful for constructing groups with efficiently solvable word problems. An example of such a construction is the free product: it is not hard to see that the word problem for a free product $G * H$ can be reduced in polynomial time to the word problem for G and H. In this paper, we introduce a new technique for obtaining upper complexity bounds for word problems. This technique is based on data compression. More precisely, we use compressed representations of strings — so called *straight-line programs*, briefly SLPs — which are able to achieve exponential compression rates for strings with repeated subpatterns. Formally, an SLP $\mathbb{A}$ is a context-free grammar which generates exactly one string $\mathrm{eval}(\mathbb{A})$. Recently, SLPs turned out to be a very flexible compressed representation of strings, which is well-suited for studying algorithms on compressed data. For instance, several polynomial time algorithms for the pattern matching problem on SLP-compressed input strings were developed [13, 19, 23]. In [21], the first author started to investigate the *compressed word problem* for a finitely generated group G with finite generating set Σ. For a given SLP $\mathbb{A}$ that generates a string over Σ and the inverses of Σ it is asked whether $\mathrm{eval}(\mathbb{A})$ represents the identity element in G (actually, in [21] the compressed word problem for finitely generated monoids was studied). This problem is equivalent to the well-known circuit evaluation problem, where we ask whether a circuit over a finitely generated group G (i.e., an acyclic directed graph with leafs labeled by generators of G and internal nodes labeled by the group multiplication) evaluates to the identity element of G. In [3] this problem was investigated for finite groups, and it was shown that there exist finite groups, for which the circuit evaluation problem is complete for P (deterministic polynomial time).

In [3, 21] the main motivation for studying the compressed word problem came from computational complexity theory. Since the input in the compressed word problem is given in a more compact form than in the ordinary word problem it can be expected that the compressed word problem is, in general, more difficult than the ordinary word problem. For instance, whereas the word problem for a finitely generated free group belongs to the class L (deterministic logspace) [20], the compressed word problem for a finitely generated free group of rank at least two is P-complete [21].[1]

In [28], the second author used the polynomial time algorithm for the compressed word problem for a free group in order to present a polynomial time algorithm for the ordinary word problem for the automorphism group of a free group, which answered a question from [17]. Hence, the compressed word problem is used in order to obtain better algorithms for the ordinary word problem. In this paper, we will continue this program and obtain efficient algorithms for a variety of word problems. In order to achieve this goal, we proceed in two steps:

In the first step (Section 3) we give connections between the compressed word problem for a group G and the word problem for some group derived from G. We prove three results of this kind:

- If H is a finitely generated subgroup of the automorphism group of a group G, then the word problem for H is logspace reducible to the compressed word problem for G (Prop. 2). This result is a straight-forward extension of Thm. 5.2 from [28].
- The word problem for the semidirect product $K \rtimes_\varphi Q$ of two finitely generated groups K and Q is logspace reducible to (i) the word problem for Q and (ii) the compressed word problem for K (Prop. 3).
- If K is a finitely generated normal subgroup of G such that the quotient G/K is an automatic group, then the word problem for G is polynomial time reducible to the compressed word problem for K (Prop. 4).

In the second step (Section 4) we concentrate on the compressed word problem. We prove the following results:

- If K is a finitely generated subgroup of G such that the index $[G : K]$ is finite, then the compressed word problem for G is polynomial time reducible to the compressed word problem for K (Thm. 1).

[1] It is believed, although not proven, that L is a proper subclass of P.

- The compressed word problem for a free product $G_1 * G_2$ is polynomial time reducible (under Turing reductions) to the compressed word problem for G_1 and G_2 (Thm. 2). This result even holds for the more general graph product construction [14] (Thm. 4).
- The compressed word problem for a graph group [11] can be solved in polynomial time (Thm. 3). In a graph group, every defining relation is of the form $ab = ba$ for generators a and b.
- The compressed word problem for a finitely generated linear group belongs to the complexity class coRP (Thm. 5), which is the complementary class of randomized polynomial time. See Section 4.4 for the definition.

We end this paper with a few direct applications of the above results. Let us mention one of them concerning topology, see [31] for definitions: Crisp and Wiest [7] have shown shown that the fundamental group of any orientable surface (and of most non-orientable surfaces) embeds in a graph group. This gives a new proof that, for all closed surfaces, the word problem for the automophism group of the fundamental group can be solved in polynomial time.

A long version containing all proofs can be obtained from the authors.

2 Preliminaries

Let Σ be a finite alphabet. Let ε denote the empty word. We use $\Sigma^{-1} = \{a^{-1} \mid a \in \Sigma\}$ to denote a disjoint copy of Σ. Let $\Sigma^{\pm 1} = \Sigma \cup \Sigma^{-1}$. For background in complexity theory see [24]. For languages K, L we write $K \leq_m^P L$ (resp. $K \leq_m^{\log} L$) if there exists a polynomial time (resp. logspace) many-one reduction from K to L. We write $K \leq_T^P L$ if there exists a polynomial time Turing reduction from K to L, which means that K can be solved in deterministic polynomial time on a Turing machine with oracle access to the language L. Let $\preceq \in \{\leq_m^P, \leq_m^{\log}, \leq_T^P\}$. In case $K \preceq L_1 \times \cdots \times L_n$ we write $K \preceq (L_1, \ldots, L_n)$. Clearly, if $L_1, \ldots, L_n$ belong to the class P (deterministic polynomial time) and $K \leq_T^P (L_1, \ldots, L_n)$, then K belongs to P as well.

2.1 Groups

For background in combinatorial group theory see [22, 31]. Let G be a *finitely generated group* and let Σ be a finite *group generating set* for G. Hence, $\Sigma^{\pm 1}$ is a finite *monoid generating set* for G and there exists a canonical monoid homomorphism $h : (\Sigma^{\pm 1})^* \to G$, which maps a word $w \in (\Sigma^{\pm 1})^*$ to the group element represented by w. For $u, v \in (\Sigma^{\pm 1})^*$ we will also say that $u = v$ in G in case $h(u) = h(v)$.

The *word problem* for G with respect to Σ is the following decision problem:

INPUT: A word $w \in (\Sigma^{\pm 1})^*$.
QUESTION: $w = 1$ in G, i.e., $h(w) = 1$?

It is well known that if Γ is another finite generating set for G, then the word problem for G with respect to Σ is logspace many-one reducible to the word problem for G with respect to Γ. This justifies one to speak just of the word problem for the group G. The word problem for G is also denoted by $\mathrm{WP}(G)$. The *free group* $F(\Sigma)$ generated by Σ can be defined as the quotient monoid

$$F(\Sigma) = (\Sigma^{\pm 1})^*/\{aa^{-1} = a^{-1}a = \varepsilon \mid a \in \Sigma\}.$$

As usual, the *free product* of two groups G_1 and G_2 is denoted by $G_1 * G_2$. The *automorphism group* of a group G is denoted by $\mathrm{Aut}(G)$. For the standard definition of *automatic groups*, see [12]. Every automatic group G is finitely presented and its word problem can be solved in time $O(n^2)$.

2.2 Trace monoids and graph groups

In the following we introduce some notions from trace theory, see [8, 10] for more details. This material will be only needed in Section 4.3. An *independence alphabet* is just a finite undirected graph (Σ, I) without loops. Hence, $I \subseteq \Sigma \times \Sigma$ is an irreflexive and symmetric relation. The *trace monoid* $\mathbb{M}(\Sigma, I)$ is defined as the quotient monoid

$$\mathbb{M}(\Sigma, I) = \Sigma^*/\{ab = ba \mid (a, b) \in I\}.$$

It is a cancellative monoid. Elements of $\mathbb{M}(\Sigma, I)$ are called *traces*. The trace represented by the word $s \in \Sigma^*$ is also denoted by $[s]_I$. The graph group $\mathbb{G}(\Sigma, I)$ is defined as the quotient group

$$\mathbb{G}(\Sigma, I) = F(\Sigma)/\{ab = ba \mid (a, b) \in I\}.$$

Note that $(a, b) \in I$ implies $a^{-1}b = ba^{-1}$ in $\mathbb{G}(\Sigma, I)$. Thus, the graph group $\mathbb{G}(\Sigma, I)$ can be also defined as the quotient monoid

$$\mathbb{G}(\Sigma, I) = \mathbb{M}(\Sigma^{\pm 1}, I)/\{[aa^{-1}]_I = [a^{-1}a]_I = [\varepsilon]_I \mid a \in \Sigma\}.$$

Here, we implicitly extend $I \subseteq \Sigma \times \Sigma$ to $I \subseteq \Sigma^{\pm 1} \times \Sigma^{\pm 1}$ by setting $(a^\alpha, b^\beta) \in I$ if and only if $(a, b) \in I$ for $a, b \in \Sigma$ and $\alpha, \beta \in \{1, -1\}$.

Free groups and free abelian groups arise as special cases of graph groups; note that $\mathbb{G}(\Sigma, \emptyset) = F(\Sigma)$ and $\mathbb{G}(\Sigma, (\Sigma \times \Sigma) \setminus \mathrm{id}_\Sigma) = \mathbb{Z}^{|\Sigma|}$. Graph groups were studied e.g. in [11]; they are also known as *free partially commutative groups* [9, 32], *right-angled Artin groups* [4, 7], and *semifree groups* [1].

2.3 Grammar based compression

In this section we introduce straight-line programs, which are used as a compressed representation of strings with reoccuring subpatterns. Following [26], a *straight-line program (SLP) over the alphabet* Γ is a context-free grammar $\mathbb{A} = (V, \Gamma, S, P)$, where V is the set of *nonterminals*, Γ is the set of *terminals*, $S \in V$ is the *initial nonterminal*, and $P \subseteq V \times (V \cup \Gamma)^*$ is the set of *productions*, such that (i) for every $X \in V$ there is exactly one $\alpha \in (V \cup \Gamma)^*$ with $(X, \alpha) \in P$ and (ii) there is no cycle in the relation $\{(X, Y) \in V \times V \mid \exists \alpha : (X, \alpha) \in P, Y \text{ occurs in } \alpha\}$. A production (X, α) is also written as $X \to \alpha$. The language generated by the SLP $\mathbb{A}$ contains exactly one word that is denoted by $\mathrm{eval}(\mathbb{A})$. More generally, every nonterminal $X \in V$ produces exactly one word that is denoted by $\mathrm{eval}_{\mathbb{A}}(X)$. We omit the index $\mathbb{A}$ if the underlying SLP is clear from the context. The size of $\mathbb{A}$ is $|\mathbb{A}| = \sum_{(X,\alpha)\in P} |\alpha|$. The length of $\mathrm{eval}(\mathbb{A})$ may be exponentially larger than $|\mathbb{A}|$; hence $\mathbb{A}$ may be seen as a compressed representation of $\mathrm{eval}(\mathbb{A})$. Every SLP can be transformed in polynomial time into an equivalent SLP that is in *Chomsky normal form* (as a context-free grammar). This means that all productions have the form $A \to BC$ or $A \to a$ for nonterminals A, B, and C and a terminal a.

In recent years, the complexity of many decision problems on strings, when the input is represented by SLPs, was investigated, see e.g. [13, 19, 21, 23, 25]. A seminal result of Plandowski [25] states that for given SLPs $\mathbb{A}$ and $\mathbb{B}$ it can be checked in polynomial time whether $\mathrm{eval}(\mathbb{A}) = \mathrm{eval}(\mathbb{B})$. The currently best known algorithm for this problem has a cubic running time [19].

The *compressed word problem* for the finitely generated group G with respect to the finite generating set Σ is the following problem:

INPUT: An SLP $\mathbb{A}$ over the terminal alphabet $\Sigma^{\pm 1}$.
QUESTION: Does $\mathrm{eval}(\mathbb{A}) = 1$ hold in G?

Here, the input size is $|\mathbb{A}|$. Also, it is easy to see that the complexity of the compressed word problem does not depend on the chosen generating set. This allows one to speak of the compressed word problem for the group G. The compressed word problem for G is also denoted by $\mathrm{CWP}(G)$. The following fact is trivial:

Proposition 1. *Assume that H is a finitely generated subgroup of the finitely generated group G. Then $\mathrm{CWP}(H) \leq_m^{\log} \mathrm{CWP}(G)$.*

3 Connections between the word problem and the compressed word problem

The three propositions from this section establish a link between the word problem and the compressed word problem. For their proofs, the following fact is crucial: Let Σ be a finite generating set for the group G and let $\varphi_1, \ldots, \varphi_n \in \mathrm{Aut}(G)$ be automorphisms of G which are taken from some fixed finite subset of $\mathrm{Aut}(G)$. Then, for every $a \in \Sigma^{\pm 1}$, we can construct an SLP $\mathbb{A}$ over the terminal alphabet $\Sigma^{\pm 1}$ such that (i) $\mathrm{eval}(\mathbb{A})$ is a word that represents the group element $\varphi_1 \cdots \varphi_n(a)$ and (ii) $|\mathbb{A}| \in O(n)$; see [28].

Proposition 2 (cf [28]). *Let G be a finitely generated group and let H be a finitely generated subgroup of $\mathrm{Aut}(G)$. Then $\mathrm{WP}(H) \leq_m^{\log} \mathrm{CWP}(G)$.*

Proposition 3. *Let K and Q be finitely generated groups and let $\varphi : Q \to \mathrm{Aut}(K)$ be a homomorphism. Then, for the semidirect product $K \rtimes_\varphi Q$ we have $\mathrm{WP}(K \rtimes_\varphi Q) \leq_m^{\log} (\mathrm{WP}(Q), \mathrm{CWP}(K))$.*

The semidirect product $G = K \rtimes_\varphi Q$ is an extension of K by Q, i.e., K is a normal subgroup of G with quotient $G/K \simeq Q$. A reasonable generalization of Prop. 3 would be $\mathrm{WP}(G) \leq_m^{\log} (\mathrm{WP}(G/K), \mathrm{CWP}(K))$. But this cannot be true: there exist finitely generated groups G, Q, K such that (i) $Q = G/K$, (ii) Q and K have a decidable word problem, and (iii) G has an undecidable word problem [2]. On the other hand, if we require additionally, that Q is finitely presented (in fact, Q recursively presented suffices), then G must have a decidable word problem [6]. For the special case that the quotient $Q = G/K$ is automatic (and hence finitely presented), we can prove the following:

Proposition 4. *Let K be a finitely generated normal subgroup of G such that the quotient $Q = G/K$ is an automatic group. Then* $\mathrm{WP}(G) \leq_m^P \mathrm{CWP}(K)$.

4 Upper bounds for compressed word problems

4.1 Finite extensions

Since every finite group is automatic, Prop. 4 applies to the case that the quotient Q is finite. In this situation, we even obtain a polynomial time reduction from the *compressed* word problem of G to the compressed word problem of K.

Theorem 1. *Assume that K is a finitely generated subgroup of the group G such that the index $[G : K]$ is finite. Then* $\mathrm{CWP}(G) \leq_m^P \mathrm{CWP}(K)$.

For the proof of Thm. 1 one proceeds in two steps. For a given SLP $\mathbb{A}$ over the generators of G one first checks whether $\mathrm{eval}(\mathbb{A})$ represents an element of the subgroup K. This is possible in polynomial time using the coset automaton (whose states are the cosets of K) and the fact that it can be checked in polynomial time whether a given finite automaton accepts $\mathrm{eval}(\mathbb{A})$ for a given SLP $\mathbb{A}$ [26]. Then, in a second step one transforms $\mathbb{A}$ in polynomial time into a new SLP $\mathbb{B}$ over generators for K such that $\mathrm{eval}(\mathbb{A})$ and $\mathrm{eval}(\mathbb{B})$ represent the same group element.

The reducibility relation $\leq_m^P$ in Thm. 1 cannot be replaced by the stronger relation $\leq_m^{\log}$ (unless P = L) because there exists a finite group G with a P-complete compressed word problem [3] (take $K = 1$ in Thm. 1).

4.2 Free products

Our main result for free products is:

Theorem 2. *Assume that $G = G_1 * G_2$. Then* $\mathrm{CWP}(G) \leq_T^P (\mathrm{CWP}(G_1), \mathrm{CWP}(G_2))$.

Let Σ_i be a finite generating set for G_i ($i \in \{1, 2\}$), where $\Sigma_1 \cap \Sigma_2 = \emptyset$. In order to reduce $\mathrm{CWP}(G)$ to $\mathrm{CWP}(G_1)$ and $\mathrm{CWP}(G_2)$, we follow the strategy for free groups [21], where *composition systems* were used. Composition systems extend SLPs by allowing also productions of the form $A \to B[i : j]$ for nonterminals A and B and $i, j \in \mathbb{N}$. Then $\mathrm{eval}(A)$ is the substring of $\mathrm{eval}(B)$ from position i to j. Hagenah [15] has shown that a given composition system can be transformed in polynomial time into an equivalent SLP. For our proof, we use a special form of composition systems, so called *2-level composition systems*. Such a system is a tuple of the form $\mathbb{A} = (\mathbb{B}, (\mathbb{B}_C)_{C \in W})$, where $\mathbb{B}$ is a composition system, which generates a word over the alphabet W. Moreover, for each $C \in W$, $\mathbb{B}_C$ is an SLP, either over the terminal alphabet $\Sigma_1^{\pm 1}$ or over the terminal alphabet $\Sigma_2^{\pm 1}$. Thus, $\mathbb{A}$ defines in a natural way a string $\mathrm{eval}(\mathbb{A}) \in (\Sigma_1^{\pm 1} \cup \Sigma_2^{\pm 1})^*$.

We transform a given input SLP $\mathbb{A}$ over the terminal alphabet $(\Sigma_1^{\pm 1} \cup \Sigma_2^{\pm 1})^*$ into a 2-level composition system $\mathbb{A}' = (\mathbb{B}, (\mathbb{B}_C)_{C \in W})$ having three additional properties:

(1) $\mathrm{eval}(\mathbb{A}) = \mathrm{eval}(\mathbb{A}')$ in the group $G_1 * G_2$.
(2) for every $C \in W$, $\mathrm{eval}(\mathbb{B}_C) \neq 1$ (either in G_1 or in G_2).
(3) for every nonterminal A of $\mathbb{B}$, if $C \in W$ and $D \in W$ are two consecutive symbols in $\mathrm{eval}_{\mathbb{B}}(A)$, then either $\mathrm{eval}(\mathbb{B}_C) \in (\Sigma_1^{\pm 1})^*$ and $\mathrm{eval}(\mathbb{B}_D) \in (\Sigma_2^{\pm 1})^*$ or $\mathrm{eval}(\mathbb{B}_C) \in (\Sigma_2^{\pm 1})^*$ and $\mathrm{eval}(\mathbb{B}_D) \in (\Sigma_1^{\pm 1})^*$.

Properties (2) and (3) ensures that $\mathrm{eval}(\mathbb{A}')$ is irreducible in the free product $G_1 * G_2$ and hence $\mathrm{eval}(\mathbb{A}) = 1$ in $G_1 * G_2$ if and only if $\mathrm{eval}(\mathbb{A}') = \varepsilon$. In order to enforce (2), we have to solve instances of $\mathrm{CWP}(G_1)$ and $\mathrm{CWP}(G_2)$. Enforcing (3) is the main difficulty. Here we follow the bottom-up procedure for free groups from [21] in order to determine maximal cancellation between strings which are concatenated on the right-hand side of some production of the SLP $\mathbb{A}$.

Again, the reducibility relation $\leq_T^P$ in Thm. 2 cannot be replaced by the stronger relation $\leq_m^{\log}$ (unless P = NC, where NC is Nick's class — the class of all problems that can be solved with polynomially many processors in polylogarithmic time) because the compressed word problem for $\mathbb{Z} * \mathbb{Z}$ is P-complete [21], whereas the compressed word problem for $\mathbb{Z}$ is easily seen to be in NC.

4.3 Graph groups and graph products

The word problem for a graph group can be solved in linear time on a RAM [9, 32]. In order to solve the compressed word problem for a graph group in polynomial time, we follow again the strategy for free groups [21]. For this, it is crucial that there exists a normal form mapping $\mathrm{NF} : \mathbb{M}(\Sigma^{\pm 1}, I) \to \mathbb{M}(\Sigma^{\pm 1}, I)$ on the trace monoid $\mathbb{M}(\Sigma^{\pm 1}, I)$ such that for all $t \in \mathbb{M}(\Sigma^{\pm 1}, I)$: (i) $t = \mathrm{NF}(t)$ in the graph group $\mathbb{G}(\Sigma, I)$ and (ii) the trace $\mathrm{NF}(t)$ cannot be factorized in $\mathbb{M}(\Sigma^{\pm 1}, I)$ as $u[aa^{-1}]_I v$ or $u[a^{-1}a]_I v$ for some $u, v \in \mathbb{M}(\Sigma^{\pm 1}, I)$ and $a \in \Sigma$ [9]. Then, for a given SLP $\mathbb{A}$ over the terminal alphabet $\Sigma^{\pm 1}$ we compute in polynomial time an SLP $\mathbb{B}$ over the terminal alphabet $\Sigma^{\pm 1}$ such that $[\mathrm{eval}(\mathbb{B})]_I = \mathrm{NF}([\mathrm{eval}(\mathbb{A})]_I)$. This calculation is again based on a bottom-up process similarly to [21], but determining the maximal amount of cancellation between composed strings of $\mathbb{A}$ becomes more involved in the presence of partial commutation. Since for every $t \in \mathbb{M}(\Sigma^{\pm 1}, I)$ we have $t = 1$ in $\mathbb{G}(\Sigma, I)$ if and only if $\mathrm{NF}(t) = [\varepsilon]_I$, we obtain:

Theorem 3. *Let (Σ, I) be a fixed independence alphabet. Then $\mathrm{CWP}(\mathbb{G}(\Sigma, I))$ belongs to P (deterministic polynomial time).*

Let us end this section with a generalization of both Thm. 2 and 3. A *graph product* is given by a triple $(\Sigma, I, (G_v)_{v \in \Sigma})$, where (Σ, I) is an independence alphabet and G_v is a group, which is associated with the node $v \in \Sigma$. W.l.o.g. assume that $\Sigma = \{1, \ldots, n\}$. The group $\mathbb{G}(\Sigma, I, (G_v)_{v \in \Sigma})$ defined by this triple is the quotient

$$\mathbb{G}(\Sigma, I, (G_v)_{v \in \Sigma}) = (G_1 * G_2 * \cdots * G_n)/\{xy = yx \mid x \in G_u, y \in G_v, (u, v) \in I\},$$

i.e., we take the free product $(G_1 * G_2 * \cdots * G_n)$, but let elements from adjacent groups commute. Note that $\mathbb{G}(\Sigma, I, (G_v)_{v\in\Sigma})$ is the graph group $\mathbb{G}(\Sigma, I)$ in case every G_v is isomorphic to $\mathbb{Z}$. Moreover, free products and direct products appear as special cases of the graph product construction. Graph products were first studied by Green [14]. By combining ideas from the proof of Thm. 2 and Thm. 3, one can show:

Theorem 4. *Assume that G is a graph product of finitely generated groups $G_1, \ldots, G_n$. Then* $\mathrm{CWP}(G) \leq_T^P (\mathrm{CWP}(G_1), \ldots, \mathrm{CWP}(G_n))$.

4.4 Linear groups

Recall that a language L belongs to the complexity class RP (randomized polynomial time) if there exists a randomized polynomial time algorithm[2] A such that: (i) if $x \notin L$ then Prob$[A$ accepts $x] = 0$ and (ii) if $x \in L$ then Prob$[A$ accepts $x] \geq 1/2$. The choice of the failure probability $1/2$ in case $x \in L$ is arbitrary: By repeating the algorithm c times (where c is some constant), we can reduce the failure probability to $(1/2)^c$ and still have a randomized polynomial time algorithm. A language L belongs to the class coRP, if the complement of L belongs to RP. This means that there exists a randomized polynomial time algorithm A such that: (i) if $x \notin L$ then Prob$[A$ accepts $x] \leq 1/2$ and (ii) if $x \in L$ then Prob$[A$ accepts $x] = 1$.

Using results from [20, 30], the compressed word problem for a finitely generated linear group can be reduced to the problem whether a circuit over a polynomial ring $R[x_1, \ldots, x_n]$ (where R is either $\mathbb{Z}$ or the finite field $\mathbb{F}_p$) evaluates to the zero polynomial. This problem belongs to coRP by [16]. Hence, we obtain:

Theorem 5. *For a finitely generated linear group G,* $\mathrm{CWP}(G)$ *belongs to coRP.*

5 Applications

In this section, we present some immediate corollaries to the results from Section 3 and 4. We concentrate on automorphism groups. Since the automorphism group of a graph group is finitely generated [18, 29], Prop. 2 and Thm. 4 imply:

Corollary 1. *For a graph group G,* $\mathrm{WP}(\mathrm{Aut}(G))$ *belongs to P.*

Let S_g be the closed orientable surface of genus g. For example, S_0 is the two-sphere. Let $\pi_1(S_g)$ denote the fundamental group (see [31] for definitions). Crisp and Wiest [7] have shown that for every $g \geq 0$, $\pi_1(S_g)$ can be embedded in a graph group. Hence, by Prop. 1 and Thm. 4, the compressed word problems for these groups can be solved in polynomial time. (This gives a new proof of a result of [28].) Since S_g is a double cover of N_{g+1}, the non-orientable surface, [31, p. 87], it follows that $\pi_1(S_g)$ is an index-2 subgroup of $\pi_1(N_{g+1})$ [31, p. 162]. With Thm. 1 and Prop. 2 we obtain:

[2] A randomized algorithm A may flip coins. Hence, it accepts a given input only with some probability. If there exists a polynomial $p(n)$ such that for every input of length n and every possible outcome of the coin flips, A runs in time at most $p(n)$, then A is a randomized polynomial time algorithm.

Corollary 2. *Let G be the fundamental group of a closed (orientable or nonorientable) surface. Then* $\mathrm{CWP}(G)$ *and* $\mathrm{WP}(\mathrm{Aut}(G))$ *belong to P.*

Automorphism groups of fundamental groups of surfaces play an important role in algebraic topology; they are closely related to mapping class groups.

6 Open problems

We finish this paper with some open problems concerning compressed word problems:

1. Is the compressed word problem for a hyperbolic group solvable in polynomial time? For torsion-free hyperbolic groups one might try to attack this question using the canonical representatives of Rips and Sela [27].
2. What about the compressed word problem for automatic groups? Is it possible to prove a non-trivial lower bound (e.g. NP-hardness or coNP-hardness) for the compressed word problem of some specific automatic group?
3. Is the uniform compressed word problem for graph groups solvable in polynomial time? In this problem, the independence alphabet (Σ, I), which defines the underlying graph group, is also part of the input. Note that in Thm. 3 the independence alphabet (Σ, I) is not part of the input.
4. Can Thm. 2 be generalized from free products to (suitably restricted) amalgamated free products and HNN-extensions?
5. Is it possible to relax the restriction to an automatic quotient group Q in Prop. 4?
6. The *compressed generalized word problem* (CGWP) for a finitely generated group G asks, whether for SLPs $\mathbb{A}, \mathbb{B}_1, \ldots, \mathbb{B}_n$ (over generators for G), the word $\mathrm{eval}(\mathbb{A})$ represents a group element from the subgroup $\langle \mathrm{eval}(\mathbb{B}_1), \ldots, \mathrm{eval}(\mathbb{B}_n)\rangle \leq G$. What is the complexity of CGWP$(F(\{a, b\}))$? We only know an exponential time algorithm for this problem.

References

1. A. Baudisch. Subgroups of semifree groups. *Acta Math. Acad. Sci. Hungar.*, 38:19–28, 1981.
2. G. Baumslag, F. B. Cannonito, and C. F. Miller, III. Infinitely generated subgroups of finitely presented groups. I. *Math. Z.*, 153(2):117–134, 1977.
3. M. Beaudry, P. McKenzie, P. Péladeau, and D. Thérien. Finite monoids: From word to circuit evaluation. *SIAM J. Comput.*, 26(1):138–152, 1997.
4. N. Brady and J. Meier. Connectivity at infinity for right angled Artin groups. *Trans. Amer. Math. Soc.*, 353:117–132, 2001.
5. F. B. Cannonito and R. W. Gatterdam. The word problem and power problem in 1-relator groups are primitive recursive. *Pacific J. Math.*, 61(2):351–359, 1975.
6. W. H. Cockcroft. The word problem in a group extension. *Quart. J. Math., Oxford Ser. (2)*, 2:123–134, 1951.
7. J. Crisp and B. Wiest. Embeddings of graph braid and surface groups in right-angled Artin groups and braid groups. *Algebr. Geom. Topol.*, 4:439–472, 2004.
8. V. Diekert. *Combinatorics on Traces*. LNCS 454, Springer, 1990.
9. V. Diekert. Word problems over traces which are solvable in linear time. *Theoret. Comput. Sci.*, 74:3–18, 1990.

10. V. Diekert and G. Rozenberg, editors. *The Book of Traces*. World Scientific, 1995.
11. C. Droms. Graph groups, coherence and three-manifolds. *J. Algebra*, 106(2):484–489, 1985.
12. D. B. A. Epstein, J. W. Cannon, D. F. Holt, S. V. F. Levy, M. S. Paterson, and W. P. Thurston. *Word processing in groups*. Jones and Bartlett, Boston, 1992.
13. L. Gasieniec, M. Karpinski, W. Plandowski, and W. Rytter. Efficient algorithms for Lempel-Ziv encoding (extended abstract). In *Proc. SWAT 1996*, LNCS 1097, pages 392–403. Springer, 1996.
14. E. R. Green. *Graph Products of Groups*. PhD thesis, The University of Leeds, 1990.
15. C. Hagenah. *Gleichungen mit regulären Randbedingungen über freien Gruppen*. PhD thesis, University of Stuttgart, Institut für Informatik, 2000.
16. O. H. Ibarra and S. Moran. Probabilistic algorithms for deciding equivalence of straight-line programs. *J. Assoc. Comput. Mach.*, 30(1):217–228, 1983.
17. I. Kapovich, A. Myasnikov, P. Schupp, and V. Shpilrain. Generic-case complexity, decision problems in group theory, and random walks. *J. Algebra*, 264(2):665–694, 2003.
18. M. R. Laurence. A generating set for the automorphism group of a graph group. *J. London Math. Soc. (2)*, 52(2):318–334, 1995.
19. Y. Lifshits. Processing compressed texts: a tractability border. To appear in *Proc. CPM 2007*, Springer, 2007.
20. R. J. Lipton and Y. Zalcstein. Word problems solvable in logspace. *J. Assoc. Comput. Mach.*, 24(3):522–526, 1977.
21. M. Lohrey. Word problems and membership problems on compressed words. *SIAM J. Comput.*, 35(5):1210 – 1240, 2006.
22. R. C. Lyndon and P. E. Schupp. *Combinatorial Group Theory*. Springer, 1977.
23. M. Miyazaki, A. Shinohara, and M. Takeda. An improved pattern matching algorithm for strings in terms of straight-line programs. In *Proc CPM 97*, LNCS 1264, pages 1–11. Springer, 1997.
24. C. H. Papadimitriou. *Computational Complexity*. Addison Wesley, 1994.
25. W. Plandowski. Testing equivalence of morphisms on context-free languages. In *Proc. ESA'94*, LNCS 855, pages 460–470. Springer, 1994.
26. W. Plandowski and W. Rytter. Complexity of language recognition problems for compressed words. In *Jewels are Forever, Contributions on Theoretical Computer Science in Honor of Arto Salomaa*, pages 262–272. Springer, 1999.
27. E. Rips and Z. Sela. Canonical representatives and equations in hyperbolic groups. *Invent. Math.*, 120:489–512, 1995.
28. S. Schleimer. Polynomial-time word problems. To appear in *Commentarii Mathematici Helvetici*.
29. H. Servatius. Automorphisms of graph groups. *J. Algebra*, 126(1):34–60, 1989.
30. H.-U. Simon. Word problems for groups and contextfree recognition. In *Proc. FCT'79*, pages 417–422. Akademie-Verlag, 1979.
31. J. Stillwell. *Classical Topology and Combinatorial Group Theory*. Springer, 1995.
32. C. Wrathall. The word problem for free partially commutative groups. *J. Symbolic Comput.*, 6(1):99–104, 1988.