Noname manuscript No.
(will be inserted by the editor)

Knapsack in Graph Groups

Markus Lohrey · Georg Zetzsche

Abstract It is shown that the knapsack problem, which was introduced by Myasnikov et al. for arbitrary finitely generated groups, can be solved in NP for every graph group. This result even holds if the group elements are represented in a compressed form by so called straight-line programs, which generalizes the classical NP-completeness result of the integer knapsack problem. If group elements are represented explicitly by words over the generators, then knapsack for a graph group belongs the class LogCFL (a subclass of P) if the graph group can be built up from the trivial group using the operations of free product and direct product with $\mathbb{Z}$. In all other cases, the knapsack problem is NP-complete.

Contents

Georg Zetzsche is supported by a fellowship within the Postdoc-Program of the German Academic Exchange Service (DAAD) and by Labex Digicosme, Univ. Paris-Saclay, project VERICONISS. Markus Lohrey is supported by the DFG project LO 748/12-1.

M. Lohrey
Universität Siegen, Germany
E-mail: lohrey@eti.uni-siegen.de

G. Zetzsche
LSV, CNRS & ENS Paris-Saclay, France
E-mail: zetzsche@lsv.fr

1 Introduction

In their paper [44], Myasnikov, Nikolaev, and Ushakov started the investigation of classical discrete optimization problems, which are formulated over the integers, for arbitrary (possibly non-commutative) groups. The general goal of this line of research is to study to what extent results from the commutative setting can be transferred to the non-commutative setting. Among other problems, Myasnikov et al. introduced for a finitely generated group G the *knapsack problem* and the *subset sum problem*. The input for the knapsack problem is a sequence of group elements $g_1, \ldots, g_k, g \in G$ (specified by finite words over the generators of G) and it is asked whether there exists a solution $(x_1, \ldots, x_k) \in \mathbb{N}^k$ of the equation $g_1^{x_1} \cdots g_k^{x_k} = g$. For the subset sum problem one restricts the solution to $\{0,1\}^k$.

For the particular case $G = \mathbb{Z}$ (where the additive notation $x_1 \cdot g_1 + \cdots + x_k \cdot g_k = g$ is usually preferred) these problems are NP-complete if the numbers $g_1, \ldots, g_k, g$ are encoded in binary representation. For subset sum, this is a classical result from Karp's seminal paper [31] on NP-completeness. Knapsack for integers is usually formulated in a more general form in the literature; NP-completeness of the above form (for binary encoded integers) was shown in [23], where the problem was called MULTISUBSET SUM.[1] Interestingly, if we consider subset sum for the group $G = \mathbb{Z}$, but encode the input numbers $g_1, \ldots, g_k, g$ in unary notation, then the problem is in DLOGTIME-uniform TC^0 (a small subclass of polynomial time and even of logarithmic space that captures the complexity of multiplication of binary encoded numbers; see e.g. the book [50] for more details) [17], and the same holds for knapsack (see Theorem 4.29). Related results can be found in [29].

In [44], Myasnikov et al. encode elements of the finitely generated group G by words over the group generators and their inverses, which corresponds to the unary encoding of integers. Among others, the following results were shown in [44]:

[1] Note that if we ask for a solution $(x_1, \ldots, x_k)$ in $\mathbb{Z}^k$, then knapsack can be solved in polynomial time (even for binary encoded integers) by checking whether $\gcd(g_1, \ldots, g_k)$ divides g.

- Subset sum and knapsack can be solved in polynomial time for every hyperbolic group.
- Subset sum for a virtually nilpotent group (a finite extension of a nilpotent group) can be solved in polynomial time.
- For the following groups, subset sum is NP-complete (whereas the word problem can be solved in polynomial time): free metabelian non-abelian groups of finite rank, the wreath product $\mathbb{Z} \wr \mathbb{Z}$, Thompson's group F, and the Baumslag-Solitar group $\mathrm{BS}(1, 2)$.

Further results on knapsack and subset sum have been recently obtained in [33]:

- For a virtually nilpotent group, subset sum belongs to NL (nondeterministic logspace).
- There is a nilpotent group of class 2 (in fact, a direct product of sufficiently many copies of the discrete Heisenberg group $H_3(\mathbb{Z})$), for which knapsack is undecidable.
- The knapsack problem for the discrete Heisenberg group $H_3(\mathbb{Z})$ is decidable. In particular, together with the previous point it follows that decidability of knapsack is not preserved under direct products.
- There is a polycyclic group with an NP-complete subset sum problem. Recently it has been shown that subset sum is NP-complete for every polycyclic group that is not virtually nilpotent [45].
- The knapsack problem is decidable for every co-context-free group.

In recent years, group-theoretic problems began to be studied in the setting where group elements are encoded in a succinct (or compressed) way. A particularly popular succinct representation are so called *straight-line programs* (SLP). These are context-free grammars that produce a single word, see [35, 36] for surveys. Over a unary alphabet, one can achieve for every word exponential compression with SLPs: The word a^n can be produced by an SLP of size $O(\log n)$. This shows that knapsack and subset sum for the group $\mathbb{Z}$ with SLP-compressed group elements correspond to the classical knapsack and subset sum problem with binary encoded numbers. To distinguish between the two variants, we will speak in this introduction of *uncompressed knapsack* (resp., *subset sum*) if the input group elements are given explicitly by words over the generators. On the other hand, if these words are represented by SLPs, we will speak of *compressed knapsack* (resp., *subset sum*). In the main part of this paper, the terms "knapsack" and "subset sum" will refer to the uncompressed version.

In this paper we will study the compressed and uncompressed versions of knapsack and subset sum for the class of *graph groups*. Graph groups are also known as "right-angled Artin groups" or "free partially commutative groups". A graph group is specified by a finite simple graph. The vertices are the generators of the group, and two generators a and b are allowed to commute if and only if a and b are adjacent. Graph groups can be regarded as interpolating between free groups and free abelian groups and constitute a group counterpart of trace monoids (free partially commutative monoids), which have been

used for the specification of concurrent behavior. In combinatorial group theory, graph groups are currently an active area of research, mainly because of their rich subgroup structure (see e.g. [6,10,20]).

Since the word problem for a graph group can be solved in polynomial time, it is clear that for each graph group, the subset sum problem belongs to NP. This result carries over to compressed subset sum, since the compressed word problem (the word problem where the input group element is given by an SLP) for a graph product can be solved in polynomial time [37] (see also [36] for more details). Our first main result states that for every graph group, even compressed knapsack belongs to NP and is in fact NP-complete. This generalizes the classical NP-completeness for knapsack (over $\mathbb{Z}$) to a much wider class of groups. To prove this result, we proceed in two steps:

- We show that if an instance $g_1^{x_1} \cdots g_k^{x_k} = g$, where all group elements $g_1, \ldots, g_k$ are given succinctly by SLPs, has a solution in a graph group, then it has a solution where every x_i is bounded exponentially in the input length (the total length of all SLPs representing the group elements $g_1, \ldots, g_k, g$).
- We then guess the binary encodings of numbers $n_1, \ldots, n_k$ that are bounded by the exponential bound from the previous point and verify in polynomial time the identity $g_1^{n_1} \cdots g_k^{n_k} = g$. The latter problem is an instance of the compressed word problem for a graph group, which can be solved in polynomial time [37].

In fact, our proof yields a stronger result: First, it yields an NP procedure for solving knapsack-like equations $g_1^{x_1} \cdots g_k^{x_k} = g$ where some of the variables $x_1, \ldots, x_k$ are allowed to be identical. We call such an equation an *exponent equation*. Hence, we prove that solvability of exponent equations over a graph group belongs to NP. A by-product of our proof is that the set of all solutions $(x_1, \ldots, x_k) \in \mathbb{N}^k$ of $g_1^{x_1} \cdots g_k^{x_k} = g$ is semilinear, and a semilinear representation can be produced effectively. This seems to be true for many groups, e.g., for all co-context-free groups [33]. On the other hand, the discrete Heisenberg group $H_3(\mathbb{Z})$ is an example of a group for which solvability of exponent equations is decidable, but the set of all solutions of an exponent equation is not semilinear; it is defined by a single quadratic Diophantine equation [33].

The second part of the paper is concerned with with uncompressed knapsack and subset sum for graph groups. In the case of knapsack, we completely determine the complexity and for subset sum, we obtain an almost complete picture. For a finite simple graph Γ, let $\mathbb{G}(\Gamma)$ denote the graph group specified by Γ.

(i) Uncompressed knapsack and subset sum for $\mathbb{G}(\Gamma)$ are complete for TC^0 if Γ is a complete graph (and thus $\mathbb{G}(\Gamma)$ is a free abelian group).[2]
(ii) Uncompressed knapsack and subset sum for $\mathbb{G}(\Gamma)$ are LogCFL-complete if Γ is not a complete graph and neither contains an induced cycle on four nodes (C4) nor an induced path on four nodes (P4).

[2] In the following, TC^0 always refers to its DLOGTIME-uniform version.

(iii) Uncompressed knapsack for $\mathbb{G}(\Gamma)$ is NP-complete if Γ contains an induced C4 or an induced P4 (it is not clear whether this also holds for subset sum).

The result (i) is a straightforward extension of the corresponding fact for $\mathbb{Z}$ [17]. The proofs for (ii) and (iii) are less obvious. Recall that LogCFL is the closure of the context-free languages under logspace reductions; it is contained in the circuit complexity class NC^2.

To show the upper bound in (ii), we use the fact that the graph groups $\mathbb{G}(\Gamma)$, where Γ neither contains an induced C4 nor an induced P4 (these graphs are the so called transitive forests), are exactly those groups that can be built up from $\mathbb{Z}$ using the operations of free product and direct product with $\mathbb{Z}$. We then construct inductively over these operations a logspace-bounded auxiliary pushdown automaton working in polynomial time (these machines accept exactly the languages in LogCFL) that checks whether an acyclic finite automaton accepts a word that is trivial in the graph group. In order to apply this result to knapsack, we finally show that every solvable knapsack instance over a graph group $\mathbb{G}(\Gamma)$ with Γ a transitive forest has a solution with polynomially bounded exponents. This is the most difficult result in the second part of this paper and it might be of independent interest.

For the lower bound in (ii), it suffices to consider the group F_2 (the free group on two generators). Our proof is based on the fact that the context-free languages are exactly those languages that can be accepted by valence automata over F_2. This is a reinterpretation of the classical theorem of Chomsky and Schützenberger. To the authors' knowledge, the result (ii) is the first completeness result for LogCFL in the area of combinatorial group theory.

Finally, for the result (iii) it suffices to show NP-hardness of knapsack for the graph groups $\mathbb{G}(\mathsf{C4})$ (where C4 is a cycle on four nodes) and $\mathbb{G}(\mathsf{P4})$ (where P4 is a cycle on four nodes). Our proof for $\mathbb{G}(\mathsf{C4})$ is based on ideas from [44]. For $\mathbb{G}(\mathsf{P4})$, we apply a technique that was first used by Aalbersberg and Hoogeboom [1] to show that the intersection non-emptiness problem for regular trace languages is undecidable for P4.

This work presents all results with full proofs from the extended abstracts in [39, 40] that concern the knapsack problem and the subset sub problem for graph groups (the paper [39] also contains transfer results on HNN-extensions and free products with amalgamations, which do not appear here).

Related work. Implicitly, the knapsack problem was also studied by Babai et al. [4], where it is shown that knapsack for commutative matrix groups over algebraic number fields can be solved in polynomial time.

The knapsack problem is a special case of the more general *rational subset membership problem*. A rational subset of a finitely generated monoid M is the homomorphic image in M of a regular language over the generators of M. In the rational subset membership problem for M the input consists of a rational subset $L \subseteq M$ (specified by a finite automaton) and an element $m \in M$ and it is asked whether $m \in L$. It was shown in [38] that the rational

subset membership problem for a graph group G is decidable if and only if the corresponding graph has (i) no induced cycle on four nodes ($\mathsf{C4}$) and (ii) no induced path on four nodes ($\mathsf{P4}$). For the decidable cases, the precise complexity is open.

Knapsack for G can be also viewed as the question, whether a word equation $z_1 z_2 \cdots z_n = 1$, where $z_1, \ldots, z_n$ are variables, together with constraints of the form $\{g^n \mid n \geq 0\}$ for the variables has a solution in G. Such a solution is a mapping $\varphi\colon \{z_1, \ldots, z_n\} \to G$ such that $\varphi(z_1 z_2 \cdots z_n)$ evaluates to 1 in G and all constraints are satisfied. For another class of constraints (so-called normalized rational constraints, which do not cover constraints of the form $\{g^n \mid n \geq 0\}$), solvability of general word equations was shown to be decidable (PSPACE-complete) for graph groups by Diekert and Muscholl [14]. This result was extended in [13] to a transfer theorem for graph products. A graph product is specified by a finite simple graph where every node is labeled with a group. The associated group is obtained from the free product of all vertex groups by allowing elements from adjacent groups to commute. Note that decidability of knapsack is not preserved under graph products: It is not even preserved under direct products (see the above mentioned results from [33]).

2 Basic concepts

2.1 Complexity classes

We assume that the reader is familiar with the complexity classes P and NP, see e.g. [3] for details. The class TC^0 is a very low circuit complexity class; it is contained for instance in NC^1 and deterministic logspace. We will use this class only in Section 4.4, and even that part can be understood without the precise definition of TC^0. Nevertheless, for completeness we include the formal definition of TC^0.

A language $L \subseteq \{0,1\}^*$ belongs to TC^0 if there exists a family $(C_n)_{n \geq 0}$ of Boolean circuits with the following properties:

- C_n has n distinguished input gates $x_1, \ldots, x_n$ and a distinguished output gate o.
- C_n accepts exactly the words from $L \cap \{0,1\}^n$, i.e., if the input gate x_i receives the input $a_i \in \{0,1\}$, then the output gate o evaluates to 1 if and only if $a_1 a_2 \cdots a_n \in L$.
- Every circuit C_n is built up from input gates, and-gates, or-gates, and majority-gates, where a majority gate evaluates to 1 if at least half of its input wires carry 1.
- All gates have unbounded fan-in, which means that there is no bound on the number of input wires for a gate.
- There is a polynomial $p(n)$ such that C_n has at most $p(n)$ many gates.
- There is a constant c such that every C_n has depth at most c, where the depth is the length of a longest path from an input gate x_i to the output gate o.

This is in fact the definition of non-uniform TC^0. Here "non-uniform" means that the mapping $n \mapsto C_n$ is not restricted in any way. In particular, it can be non-computable. For algorithmic purposes one usually adds some uniformity requirement to the above definition. The most "uniform" version of TC^0 is $\mathsf{DLOGTIME}$-uniform TC^0. For this, one encodes the gates of each circuit C_n by bit strings of length $O(\log n)$. Then the circuit family $(C_n)_{n\geq 0}$ is called $\mathsf{DLOGTIME}$-*uniform* if (i) there exists a deterministic Turing machine that computes for a given gate $u \in \{0,1\}^*$ of C_n ($|u| \in O(\log n)$) in time $O(\log n)$ the type (of gate u, where the types are $x_1, \ldots, x_n$, and, or, majority) and (ii) there exists a deterministic Turing machine that decides for two given gate $u, v \in \{0,1\}^*$ of C_n ($|u|, |v| \in O(\log n)$) in time $O(\log n)$ whether there is a wire from gate u to gate v. In the following, we always implicitly refer to $\mathsf{DLOGTIME}$-uniform TC^0.

If the language L in the above definition of TC^0 is defined over a non-binary alphabet Σ then one first has to fix a binary encoding of words over Σ. When talking about hardness for TC^0, one has to use reductions, whose computational power are below TC^0, e.g. AC^0-Turing-reductions. The precise definition of these reductions is not important for our purpose. Important problems that are complete for TC^0 are:

- The languages $\{w \in \{0,1\}^* \mid |w|_0 \leq |w|_1\}$ and $\{w \in \{0,1\}^* \mid |w|_0 = |w|_1\}$, where $|w|_a$ denotes the number of occurrences of a in w, see e.g. [50].
- The computation (of a certain bit) of the binary representation of the product of two (or any number of) binary encoded integers [25].
- The computation (of a certain bit) of the binary representation of the integer quotient of two binary encoded integers [25].
- The word problem for every infinite solvable linear group [32].
- The conjugacy problem for the Baumslag-Solitar group $\mathsf{BS}(1,2)$ [15].

The class LogCFL consists of all problems that are logspace reducible to a context-free language. The class LogCFL is included in the parallel complexity class NC^2 and has several alternative characterizations (see e.g. [48,50]):

- logspace bounded alternating Turing-machines with polynomial tree size,
- semi-unbounded Boolean circuits of polynomial size and logarithmic depth, and
- logspace bounded auxiliary pushdown automata with polynomial running time.

For our purposes, the last characterization is most suitable. An AuxPDA (for auxiliary pushdown automaton) is a nondeterministic pushdown automaton with a two-way input tape and an additional work tape. Here we only consider AuxPDA with the following two restrictions:

- The length of the work tape is restricted to $O(\log n)$ for an input of length n (logspace bounded).
- There is a polynomial $p(n)$, such that every computation path of the AuxPDA on an input of length n has length at most $p(n)$ (polynomially time bounded).

Whenever we speak of an AuxPDA in the following, we implicitly assume that the AuxPDA is logspace bounded and polynomially time bounded. Deterministic AuxPDA are defined in the obvious way. The class of languages that are accepted by AuxPDA is exactly LogCFL, whereas the class of languages accepted by deterministic AuxPDA is LogDCFL (the closure of the deterministic context-free languages under logspace reductions) [48].

2.2 Finite automata

We will use standard notions from automata theory. We define a *nondeterministic finite automaton* (NFA) as a tuple $\mathcal{A} = (Q, \Sigma, \Delta, q_0, F)$, where Q is a finite set of states, Σ is the *input alphabet*, $q_0 \in Q$ is the *initial state*, $F \subseteq Q$ is the *final state*, and $\Delta \subseteq Q \times \Sigma^* \times Q$ is a finite set of *transitions*. Note that such an automaton can read several (including zero) many symbols in a transition. A *spelling* NFA is an NFA $\mathcal{A} = (Q, \Sigma, \Delta, q_0, F)$, where $\Delta \subseteq Q \times \Sigma \times Q$. The language accepted by $\mathcal{A}$ is denoted with $L(\mathcal{A})$. If we allow ε-transitions of the form (q, ε, p), which is the case for general (non-spelling) NFA, then we can assume that the set of final states F consists of a unique state q_f, in which case we write $\mathcal{A} = (Q, \Sigma, \Delta, q_0, q_f)$.

An *acyclic NFA* is a (not necessarily spelling) NFA $\mathcal{A} = (Q, \Sigma, \Delta, q_0, q_f)$ such that the relation $\{(p, q) \mid \exists w \in \Sigma^* : (p, w, q) \in \Delta\}$ is acyclic. An *acyclic loop NFA* is a (not necessarily spelling) NFA $\mathcal{A} = (Q, \Sigma, \Delta, q_0, q_f)$ such that there exists a linear order $\preceq$ on Δ having the property that for all $(p, u, q), (q, v, r) \in \Delta$ it holds $(p, u, q) \preceq (q, v, r)$. Thus, an acyclic loop NFA is obtained from an acyclic NFA by attaching to some of the states a unique loop.

2.3 Vectors and semilinear sets

Vectors will be column vectors, unless we explicitly speak of row vectors. For a vector $x \in \mathbb{Z}^k$ we denote with x^T the corresponding row vector. Given a vector $x = (x_1, \ldots, x_k)^T \in \mathbb{Z}^k$, we use three different standard norms:

$$\|x\|_\infty = \max\{|x_i| \mid 1 \leq i \leq k\}, \tag{1}$$

$$\|x\|_2 = \sqrt{\sum_{1 \leq i \leq k} x_i^2}, \tag{2}$$

$$\|x\|_1 = \sum_{i=1}^{m} |x_i|. \tag{3}$$

For a subset $T \subseteq \mathbb{N}^k$, we write $T^\oplus$ for the smallest subset of $\mathbb{N}^k$ that contains $T \cup \{0\}$ and is closed under addition. A subset $S \subseteq \mathbb{N}^k$ is called *linear* if there is a vector $x \in \mathbb{N}^k$ and a finite set $F \subseteq \mathbb{N}^k$ such that $S = x + F^\oplus$. Note that a set is linear if and only if it can be written as $x + A\mathbb{N}^t$ for

some $x \in \mathbb{N}^k$ and some matrix $A \in \mathbb{N}^{k \times t}$. Here, $A\mathbb{N}^t$ denotes the set of all vectors Ay for $y \in \mathbb{N}^t$. A *semilinear set* is a finite union of linear sets. If $S = \bigcup_{i=1}^{n} x_i + F_i^{\oplus}$ for $x_1, \ldots, x_n \in \mathbb{N}^k$ and finite sets $F_1, \ldots, F_n \subseteq \mathbb{N}^k$, then the tuple $(x_1, F_1, \ldots, x_n, F_n)$ is a *semilinear representation* of S. Saying that a set S is *effectively semilinear* means that a semilinear representation for S can be computed from certain input data.

2.4 Words and straight-line programs

For a word w we denote with $\mathrm{alph}(w)$ the set of symbols occurring in w. The length of the word w is $|w|$.

A *straight-line program*, briefly *SLP*, is basically a context-free grammar that produces exactly one string. To ensure this, the grammar has to be acyclic and deterministic (every variable has a unique production where it occurs on the left-hand side). Formally, an SLP is a tuple $\mathcal{G} = (V, \Sigma, \mathrm{rhs}, S)$, where V is a finite set of *variables* (or *nonterminals*), Σ is the *terminal alphabet*, $S \in V$ is the *start variable*, and rhs maps every variable to a *right-hand side* $\mathrm{rhs}(A) \in (V \cup \Sigma)^*$. We require that there is a linear order $<$ on V such that $B < A$ whenever $B \in N \cap \mathrm{alph}(\mathrm{rhs}(A))$. Every variable $A \in V$ derives to a unique string $\mathrm{val}_{\mathcal{G}}(A)$ by iteratively replacing variables by the corresponding right-hand sides, starting with A. Finally, the string *derived by* $\mathcal{G}$ is $\mathrm{val}(\mathcal{G}) = \mathrm{val}_{\mathcal{G}}(S)$.

Let $\mathcal{G} = (V, \Sigma, \mathrm{rhs}, S)$ be an SLP. The *size* of $\mathcal{G}$ is $|\mathcal{G}| = \sum_{A \in V} |\mathrm{rhs}(A)|$, i.e., the total length of all right-hand sides. A simple induction shows that for every SLP $\mathcal{G}$ of size m one has $|\mathrm{val}(\mathcal{G})| \leq \mathcal{O}(3^{m/3}) \subseteq 2^{\mathcal{O}(m)}$ [9, proof of Lemma 1]. On the other hand, it is straightforward to define an SLP $\mathcal{H}$ of size $2n$ such that $|\mathrm{val}(\mathcal{H})| \geq 2^n$. This justifies to see an SLP $\mathcal{G}$ as a compressed representation of the string $\mathrm{val}(\mathcal{G})$, and exponential compression rates can be achieved in this way. More details on SLPs can be found in the survey [35].

2.5 Groups

We assume that the reader has some basic knowledge concerning (finitely generated) groups (see e.g. [41] for further details). Let G be a finitely generated group, and let A be a finite generating set for G. Then, elements of G can be represented by finite words over the alphabet $A^{\pm 1} = A \cup A^{-1}$. The free group generated by A is denoted by $F(A)$ and we also write F_n for $F(A)$ if $|A| = n$. Elements of $F(A)$ can be identified with *irreducible* words over $A^{\pm 1}$, i.e., words that do not contain a factor of the form aa^{-1} or $a^{-1}a$ for $a \in A$. The length $|g|$ of $g \in F(A)$ is the the length of the irreducible word corresponding to g.

A group G is *finitely presented* if there exists a finite alphabet A and a finite set of words $R \subseteq (A^{\pm 1})^*$ (which we identify with a subset of $F(A)$) such that G is isomorphic to $F(A)/N$, where N is the smallest normal subgroup of $F(A)$ that contains R. The group $F(A)/N$ is usually denoted by $\langle A \mid R \rangle$ and

the pair (A, R) is called a *presentation* of G. Elements of R are called *relators*. With $\langle A \mid u_1 = v_1, \ldots, u_k = v_k \rangle$ we denote the group $\langle A \mid u_1 v_1^{-1}, \ldots, u_k v_k^{-1} \rangle$.

It is a standard fact that an element $g \in F(A)$ represents the identity of $G = \langle A \mid R \rangle$ if and only if g can be written in $F(A)$ as a product $\prod_{i=1}^{n} c_i^{-1} r_i c_i$, where $c_i \in F(A)$ and $r_i \in R \cup R^{-1}$. The minimal such n is called the *area* of g (with respect to the presentation (A, R)). The *Dehn function* of the presentation (A, R) is the function $f \colon \mathbb{N} \to \mathbb{N}$ that maps n to the maximal area of an element $g \in F(A)$ such that $g = 1$ in $F(A)$ and $|g| \leq n$. We say that a finitely presented group G *has a polynomial Dehn function* if there exists a presentation for G whose Dehn function is bounded by a polynomial.

2.6 Knapsack and exponent equations

Let G be a finitely generated group, and fix a generating set A for G. An *exponent equation* over G is an equation of the form

$$v_0 u_1^{x_1} v_1 u_2^{x_2} v_2 \cdots u_n^{x_n} v_n = 1 \tag{4}$$

where $u_1, u_2, \ldots, u_n, v_0, v_1, \ldots, v_n \in G$ are group elements that are given by finite words over the alphabet $A^{\pm 1}$ and $x_1, x_2, \ldots, x_n$ are not necessarily distinct variables. Such an exponent equation is *solvable* if there exists a mapping $\sigma \colon \{x_1, \ldots, x_n\} \to \mathbb{N}$ such that $v_0 u_1^{\sigma(x_1)} v_1 u_1^{\sigma(x_2)} v_2 \cdots u_n^{\sigma(x_n)} v_n = 1$ in the group G. If there is no danger of confusion, we will simplify notation and not distinguish between variables and solutions. *Solvability of exponent equations over G* is the following computational problem:

Input: An exponent equation E over G (where elements of G are specified by words over the alphabet $A^{\pm 1}$).

Question: Is E solvable?

It suffices to consider exponent equations of the form $u_1^{x_1} u_2^{x_2} \cdots u_n^{x_n} v_n = 1$: using conjugation, we can replace an equation $v_0 u_1^{x_1} v_1 u_2^{x_2} v_2 \cdots u_n^{x_n} v = 1$ by $(v_0 u_1 v_0^{-1})^{x_1} (v_0 v_1) u_2^{x_2} v_2 \cdots u_n^{x_n} v_n = 1$ and then continue in this way. On the other hand, in some of our proof it is convenient to allow the group elements $v_0, v_1, \ldots, v_{n-1}$ in (4).

The *knapsack problem* for the group G is the restriction of solvability of exponent equations over G to exponent equations of the form $u_1^{x_1} u_2^{x_2} \cdots u_n^{x_n} u^{-1} = 1$ or, equivalently, $u_1^{x_1} u_2^{x_2} \cdots u_n^{x_n} = u$ where the exponent variables $x_1, \ldots, x_n$ have to be pairwise different.

We will also study a compressed version of exponent equations over G, where elements of G are given by SLPs over $A^{\pm 1}$. A *compressed exponent equation* is an exponent equation $v_0 u_1^{x_1} v_1 u_2^{x_2} v_2 \cdots u_n^{x_n} v_n = 1$ where the group elements $u_1, u_2, \ldots, u_n, v_0, v_1, \ldots, v_n \in G$ are given by SLPs over the terminal alphabet $A^{\pm 1}$. The sum of the sizes of these SLPs is the size of the compressed exponent equation.

Let us define *solvability of compressed exponent equations over G* as the following computational problem:

Input: A compressed exponent equation E over G.

Question: Is E solvable?

The *compressed knapsack problem* for G is defined analogously. Note that with this terminology, the classical knapsack problem for binary encoded integers is the compressed knapsack problem for the group $\mathbb{Z}$. The binary encoding of an integer can be easily transformed into an SLP over the alphabet $\{a, a^{-1}\}$ (where a is a generator of $\mathbb{Z}$) and vice versa. Here, the number of bits in the binary encoding and the size of the SLP are linearly related.

Remark 2.1 Let us comment on the difference between the knapsack problem and solvability of exponent equations. The main concern of this work is the knapsack problem, where all variables are distinct. However, the methods we use in section 3 for obtaining the NP upper bound apply to general (even compressed) exponent equations. Our proof of the LogCFL upper bound in section 4, on the other hand, only works for the knapsack problem.

Nevertheless, all our (complexity) lower bounds are with respect to the knapsack problem.

It is a simple observation that the decidability and complexity of solvability of (compressed) exponent equations over G as well as the (compressed) knapsack problem for G does not depend on the chosen finite generating set for the group G. Therefore, we do not have to mention the generating set explicitly in these problems.

Remark 2.2 Since we are dealing with a group, one might also allow solution mappings $\sigma\colon \{x_1, \ldots, x_n\} \to \mathbb{Z}$ to the integers. But this variant of solvability of (compressed) exponent equations (knapsack, respectively) can be reduced to the above version, where σ maps to $\mathbb{N}$, by simply replacing a power $u_i^{x_i}$ by $u_i^{x_i}(u_i^{-1})^{y_i}$, where y_i is a fresh variable.

This work is concerned with decidability and complexity of solvability of exponent equations for so-called graph groups, which will be introduced in the next section.

2.7 Traces and graph groups

Let (A, I) be a finite simple graph. In other words, the edge relation $I \subseteq A \times A$ is irreflexive and symmetric. It is also called the *independence relation*, and (A, I) is called an *independence alphabet*. The relation $D = (A \times A) \setminus I$ is also called the associated *dependence relation* and (A, D) is called the associated *dependence alphabet*. We consider the monoid $\mathbb{M}(A, I) = A^*/\equiv_I$, where $\equiv_I$ is the smallest congruence relation on the free monoid A^* that contains all pairs (ab, ba) with $a, b \in A$ and $(a, b) \in I$. This monoid is called a *trace monoid* or *partially commutative free monoid*; it is cancellative, i.e., $xy = xz$ or $yx = zx$ implies $y = z$. Elements of $\mathbb{M}(A, I)$ are called *Mazurkiewicz traces* or simply *traces*. The trace represented by the word u is denoted by $[u]_I$, or

simply u if no confusion can arise. For a language $L \subseteq A^*$ we denote with $[L]_I = \{u \in A^* \mid \exists v \in L : u \equiv_I v\}$ its *partially commutative closure*. The length of the trace $[u]_I$ is $|[u]_I| = |u|$ and its alphabet is $\mathrm{alph}([u]_I) = \mathrm{alph}(u)$. It is easy to see that these definitions do not depend on the concrete word that represents the trace $[u]_I$. For subsets $B, C \subseteq A$ we write BIC for $B \times C \subseteq I$. If $B = \{a\}$ we simply write aIC. For traces s, t we write sIt for $\mathrm{alph}(s)I\mathrm{alph}(t)$. The empty trace $[\varepsilon]_I$ is the identity element of the monoid $\mathbb{M}(A, I)$ and is denoted by 1. A trace t is *connected* if we cannot factorize t as $t = uv$ with $u \neq 1 \neq v$ and uIv.

A trace $t \in \mathbb{M}(A, I)$ can be visualized by its *dependence graph* D_t. To define D_t, choose an arbitrary word $w = a_1 a_2 \cdots a_n$, $a_i \in A$, with $t = [w]_I$ and define $D_t = (\{1, \ldots, n\}, E, \lambda)$, where $E = \{(i, j) \mid i < j, (a_i, a_j) \in D\}$ and $\lambda(i) = a_i$. If we identify isomorphic dependence graphs, then this definition is independent of the chosen word representing t. Moreover, the mapping $t \mapsto D_t$ is injective. As a consequence of the representation of traces by dependence graphs, one obtains Levi's lemma for traces (see e.g. [16, p. 74]), which is one of the fundamental facts in trace theory. The formal statement is as follows.

Lemma 2.3 (Levi's lemma) *Let $u_1, \ldots, u_m, v_1, \ldots, v_n \in \mathbb{M}(A, I)$. Then*

$$u_1 u_2 \cdots u_m = v_1 v_2 \cdots v_n$$

if and only if there exist $w_{i,j} \in \mathbb{M}(A, I)$ $(1 \leq i \leq m,\ 1 \leq j \leq n)$ such that

- *$u_i = w_{i,1} w_{i,2} \cdots w_{i,n}$ for every $1 \leq i \leq m$,*
- *$v_j = w_{1,j} w_{2,j} \cdots w_{m,j}$ for every $1 \leq j \leq n$, and*
- *$w_{i,j} I w_{k,\ell}$ if $1 \leq i < k \leq m$ and $n \geq j > \ell \geq 1$.*

The situation in the lemma will be visualized by a diagram of the following kind. The i–th column corresponds to u_i, the j–th row corresponds to v_j, and the intersection of the i–th column and the j–th row represents $w_{i,j}$. Furthermore $w_{i,j}$ and $w_{k,\ell}$ are independent if one of them is left-above the other one.

v_n	$w_{1,n}$	$w_{2,n}$	$w_{3,n}$	$\cdots$	$w_{m,n}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
v_3	$w_{1,3}$	$w_{2,3}$	$w_{3,3}$	$\cdots$	$w_{m,3}$
v_2	$w_{1,2}$	$w_{2,2}$	$w_{3,2}$	$\cdots$	$w_{m,2}$
v_1	$w_{1,1}$	$w_{2,1}$	$w_{3,1}$	$\cdots$	$w_{m,1}$
	u_1	u_2	u_3	$\cdots$	u_m

A consequence of Levi's Lemma is that trace monoids are cancellative, i.e., $usv = utv$ implies $s = t$ for all traces $s, t, u, v \in \mathbb{M}(A, I)$.

Let $s, t \in \mathbb{M}(A, I)$ be traces. We say that s is a *prefix* of t if there is a trace $r \in \mathbb{M}(A, I)$ with $sr = t$. Moreover, we define $\rho(t)$ as the number of prefixes of t. We will use the following statement from [5].

Lemma 2.4 *Let $t \in \mathbb{M}(A, I)$ be a trace of length n. Then $\rho(t) \in \mathcal{O}(n^\alpha)$, where α is the size of a largest clique of the associated dependence alphabet (A, D).*

With an independence alphabet (A, I) we associate the finitely presented group

$$\mathbb{G}(A, I) = \langle A \mid ab = ba \ ((a, b) \in I)\rangle.$$

Such a group is called a *graph group*, or *right-angled Artin group*[3], or *free partially commutative group*. Here, we use the term graph group. Graph groups received a lot of attention in group theory during the last few years, mainly due to their rich subgroup structure [6,10,20], and their relationship to low dimensional topology (via so-called virtually special groups) [2,24,51]. We represent elements of $\mathbb{G}(A, I)$ by traces over an extended independence alphabet. For this, let $A^{-1} = \{a^{-1} \mid a \in A\}$ be a disjoint copy of the alphabet A, and let $A^{\pm 1} = A \cup A^{-1}$. We define $(a^{-1})^{-1} = a$ and for a word $w = a_1 a_2 \cdots a_n$ with $a_i \in A^{\pm 1}$ we define $w^{-1} = a_n^{-1} \cdots a_2^{-1} a_1^{-1}$. This defines an involution (without fixed points) on $(A^{\pm 1})^*$. We extend the independence relation I to $A^{\pm 1}$ by $(a^x, b^y) \in I$ for all $(a, b) \in I$ and $x, y \in \{-1, 1\}$. Then, there is a canonical surjective morphism $h\colon \mathbb{M}(A^{\pm 1}, I) \to \mathbb{G}(A, I)$ that maps every symbol $a \in A^{\pm 1}$ to the corresponding group element. Of course, h is not injective, but we can easily define a subset $\mathrm{IRR}(A^{\pm 1}, I) \subseteq \mathbb{M}(A^{\pm 1}, I)$ of *irreducible traces* such that h restricted to $\mathrm{IRR}(A^{\pm 1}, I)$ is bijective. The set $\mathrm{IRR}(A^{\pm 1}, I)$ consists of all traces $t \in \mathbb{M}(A^{\pm 1}, I)$ such that t does not contain a factor $[aa^{-1}]_I$ with $a \in A^{\pm 1}$, i.e., there do not exist $u, v \in \mathbb{M}(A^{\pm 1}, I)$ and $a \in A^{\pm 1}$ such that in $\mathbb{M}(A^{\pm 1}, I)$ we have a factorization $t = u[aa^{-1}]_I v$. For every trace t there exists a corresponding *irreducible normal form* that is obtained by removing from t factors $[aa^{-1}]_I$ with $a \in A^{\pm 1}$ as long as possible. It can be shown that this reduction process is terminating (which is trivial since it reduces the length) and confluent (in [34] a more general confluence lemma for graph products of monoids is shown). Hence, the irreducible normal form of t does not depend on the concrete order of reduction steps. For a group element $g \in \mathbb{G}(A, I)$ we denote with $|g|$ the length of the unique trace $t \in \mathrm{IRR}(A^{\pm 1}, I)$ such that $h(t) = g$.

For a trace $t = [u]_I$ ($u \in (A^{\pm 1})^*$) we can define $t^{-1} = [u^{-1}]_I$. This is well-defined, since $u \equiv_I v$ implies $u^{-1} \equiv_I v^{-1}$. The following lemma will be important; see also [14, Lemma 23].

Lemma 2.5 *Let $s, t \in \mathrm{IRR}(A^{\pm 1}, I)$. Then there exist unique factorizations $s = up$ and $t = p^{-1}v$ in $\mathbb{M}(A^{\pm 1}, I)$ such that $uv \in \mathrm{IRR}(A^{\pm 1}, I)$. Hence, uv is the irreducible normal form of st.*

3 Compressed knapsack and exponent equations

The goal of this section is to show the following result:

Theorem 3.1 *Let (A, I) be a fixed independence alphabet. Solvability of compressed exponent equations over the graph group $\mathbb{G}(A, I)$ is in* NP.

[3] This term comes from the fact that right-angled Artin groups are exactly the Artin groups corresponding to right-angled Coxeter groups.

As a byproduct, we will prove that the set of solutions of an exponent equation over a fixed graph group is semilinear.

In Section 3.1 we will analyze the possible factorizations of a power u^x, where u is a connected trace and x is a natural number, into a product $y_1y_2\cdots y_m$ of traces. In Section 3.2 we will describe the set of solution of a trace equation $pu^xs = qv^yt$, where u and v are connected traces and $x, y \in \mathbb{N}$ are variables. In Section 3.3 we state an auxiliary result on linear diophantine equations that follows easily from a result of [19]. In Section 3.4 we prove the key lemma that allows us to reduce an exponent equations over a graph group to a system of trace equations. Finally, in Section 3.5 we prove Theorem 3.1. Section 3.6 deals with the compressed solvability of exponent equations in a graph group that is also part of the input (and given by the defining independence alphabet).

3.1 Factorizations of powers

Based on Levi's lemma we prove in this section a factorization result for powers of a connected trace. We start with the case that we factorize such a power into two factors.

Lemma 3.2 *Let $u \in \mathbb{M}(A, I) \setminus \{1\}$ be a connected trace. Then, for all $x \in \mathbb{N}$ and all traces y_1, y_2 the following two statements are equivalent:*

(i) $u^x = y_1y_2$

(ii) *There exist $l, k, c \in \mathbb{N}$ and traces s, p such that: $y_1 = u^ls$, $y_2 = pu^k$, $sp = u^c$, $l + k + c = x$, and $c \leq |A|$.*

Proof That (ii) implies (i) is clear. It remains to prove that (i) implies (ii). Assume that $u^x = y_1y_2$ holds. The case that $x \leq |A|$ is trivial. Hence, assume that $x \geq |A|+1$. We apply Levi's lemma (Lemma 2.3) to the identity $u^x = y_1y_2$:

y_2	$u_{1,2}$	$u_{2,2}$	$u_{3,2}$	$u_{4,2}$	$\cdots$	$u_{x-1,2}$	$u_{x,2}$
y_1	$u_{1,1}$	$u_{2,1}$	$u_{3,1}$	$u_{4,1}$	$\cdots$	$u_{x-1,1}$	$u_{x,1}$
	u	u	u	u	$\cdots$	u	u

Let $A_i = \mathrm{alph}(u_{1,2}\cdots u_{i,2})$. Then $A_i \subseteq A_{i+1}$. If $A_1 = \emptyset$ then $u_{1,2} = 1$ and we can go to Case 2 below. Otherwise, assume that $A_1 \neq \emptyset$. In that case there must exist $1 \leq i \leq |A|$ such that $A_i = A_{i+1}$, which implies $\mathrm{alph}(u_{i+1,2}) \subseteq A_i$. Since $u_{i+1,1}I(u_{1,2}\cdots u_{i,2})$ we also have $u_{i+1,1}Iu_{i+1,2}$. Since u is connected, we have $u_{i+1,1} = 1$ or $u_{i+1,2} = 1$. We can therefore distinguish the following two cases:

Case 1. There exists $1 \leq i \leq |A| + 1$ such that $u_{i,1} = 1$. Then $u_{i,2} = u$, which implies $u_{j,1} = 1$ for all $j > i$ (since $u_{i,2}Iu_{j,1}$):

y_2	$u_{1,2}$	$u_{2,2}$	$\cdots$	$u_{i-1,2}$	u	u	$\cdots$	u	u
y_1	$u_{1,1}$	$u_{2,1}$	$\cdots$	$u_{i-1,1}$	1	1	$\cdots$	1	1
	u	u	$\cdots$	u	u	u	$\cdots$	u	u

Let $s = u_{1,1}u_{2,1}\cdots u_{i-1,1}$ and $p = u_{1,2}u_{2,2}\cdots u_{i-1,2}$. Thus, $y_1 = u^0 s$, $y_2 = pu^{x-i+1}$ and $sp = u^{i-1}$ with $i-1 \leq |A|$, and the conclusion of the lemma holds.

Case 2. There exists $1 \leq i \leq |A|+1$ such that $u_{i,2} = 1$. Then, $u_{j,2} = 1$ for all $j < i$ (since $u_{i,1} = u$ and $u_{j,2} I u_{i,1}$):

y_2	1	1	$\cdots$	1	1	$u_{i+1,2}$	$\cdots$	$u_{x-1,1}$	$u_{x,2}$
y_1	u	u	$\cdots$	u	u	$u_{i+1,1}$	$\cdots$	$u_{x-1,1}$	$u_{x,1}$
	u	u	$\cdots$	u	u	u	$\cdots$	u	u

Let $y_1' = u_{i+1,1}\cdots u_{x,1}$. Hence, $u^{x-i} = y_1' y_2$. We can use induction to get factorizations $y_1' = u^l s$, $y_2 = pu^k$, and $sp = u^c$ with $c \leq |A|$ and $k+l+c = x-i$. Finally, we have $y_1 = u^i y_1' = u^{i+l} s$, which shows the conclusion of the lemma. □

Now we lift Lemma 3.2 to an arbitrary number of factors.

Lemma 3.3 *Let $u \in \mathbb{M}(A,I) \setminus \{1\}$ be a connected trace and $m \in \mathbb{N}$, $m \geq 2$. Then, for all $x \in \mathbb{N}$ and traces $y_1, \ldots, y_m$ the following two statements are equivalent:*

(i) $u^x = y_1 y_2 \cdots y_m$.

(ii) *There exist traces $p_{i,j}$ $(1 \leq j < i \leq m)$, s_i $(1 \leq i \leq m)$ and numbers $x_i, c_j \in \mathbb{N}$ $(1 \leq i \leq m,\ 1 \leq j \leq m-1)$ such that:*
- $y_i = (\prod_{j=1}^{i-1} p_{i,j}) u^{x_i} s_i$ *for all* $1 \leq i \leq m$,
- $p_{i,j} I p_{k,l}$ *if* $j < l < k < i$ *and* $p_{i,j} I (u^{x_k} s_k)$ *if* $j < k < i$,[4]
- $s_m = 1$ *and for all* $1 \leq j < m$, $s_j \prod_{i=j+1}^{m} p_{i,j} = u^{c_j}$,
- $c_j \leq |A|$ *for all* $1 \leq j \leq m-1$,
- $x = \sum_{i=1}^{m} x_i + \sum_{i=1}^{m-1} c_i$.

Proof Let us first show that (ii) implies (i). Assume that (ii) holds. Then we get

$$y_1 y_2 \cdots y_m = \prod_{i=1}^{m} \Big((\prod_{j=1}^{i-1} p_{i,j}) u^{x_i} s_i \Big).$$

The independencies $p_{i,j} I p_{k,l}$ for $j < l < k < i$ and $p_{i,j} I (u^{x_k} s_k)$ for $j < k < i$ yield

$$\begin{aligned} &\prod_{i=1}^{m} \Big((\prod_{j=1}^{i-1} p_{i,j}) u^{x_i} s_i \Big) \\ &= u^{x_1} s_1 p_{2,1} \cdots p_{m,1} u^{x_2} s_2 p_{3,2} \cdots p_{m,2} u^{x_3} s_3 \cdots u^{x_{m-1}} s_{m-1} p_{m,m-1} u^{x_m} s_m \\ &= u^{x_1} u^{c_1} u^{x_2} u^{c_2} u^{x_3} \cdots u^{c_{m-1}} u^{x_m} = u^x. \end{aligned}$$

We now prove that (i) implies (ii) by induction on m. So, assume that $u^x = y_1 y_2 \cdots y_m$. The case $m = 2$ follows directly from Lemma 3.2. Now assume that

[4] Note that since $\mathrm{alph}(p_{i,j}) \subseteq \mathrm{alph}(u)$, we must have $p_{i,j} = 1$ or $x_k = 0$ whenever $j < k < i$.

$m \geq 3$. By Lemma 3.2 there exist factorizations $y_1 = u^{x_1}s_1$, $y_2 \cdots y_m = p_1 u^{x'}$, and $s_1 p_1 = u^{c_1}$ with $c_1 \leq |A|$ and $x_1 + x' + c_1 = x$. Levi's lemma applied to $y_2 \cdots y_m = p_1 u^{x'}$ gives the following diagram:

y_m	$p_{m,1}$	y'_m					
$\vdots$	$\vdots$	$\vdots$					
y_3	$p_{3,1}$	y'_3					
y_2	$p_{2,1}$	y'_2					
	p_1	u	u	u	$\ldots$	u	u

There exist y'_i with $y_i = p_{i,1}y'_i$ $(2 \leq i \leq m)$, $y'_2 \cdots y'_m = u^{x'}$, and $y'_j I p_{i,1}$ for $j < i$. By induction on m we get factorizations

$$y'_i = \prod_{j=2}^{i-1} p_{i,j} u^{x_i} s_i$$

for $2 \leq i \leq m$ such that for all $2 \leq j < i \leq m$:

- $p_{i,j} I p_{k,l}$ if $j < l < k < i$ and $p_{i,j} I (u^{x_k} s_k)$ if $j < k < i$,
- $s_m = 1$ and for all $2 \leq j < m$, $s_j \prod_{i=j+1}^{m} p_{i,j} = u^{c_j}$ for some $c_j \leq |A|$,
- $x' = \sum_{i=2}^{m} x_i + \sum_{i=2}^{m-1} c_i$.

Since $y'_j I p_{i,1}$ for $j < i$ we get $p_{i,1} I p_{j,k}$ for $1 < k < j < i$ and $p_{i,1} I u^{x_j} s_j$ for $1 < j < i$. Finally, we have

$$s_1 \prod_{i=2}^{m} p_{i,1} = s_1 p_1 = u^{c_1}$$

and

$$x = x_1 + c_1 + x' = x_1 + c_1 + \sum_{i=2}^{m} x_i + \sum_{i=2}^{m-1} c_i = \sum_{i=1}^{m} x_i + \sum_{i=1}^{m-1} c_i.$$

This proves the lemma. □

Remark 3.4 In Section 3.5 we will apply Lemma 3.3 in order to replace an equation $u^x = y_1 y_2 \cdots y_m$ (where $x, y_1, \ldots, y_m$ are variables and u is a concrete connected trace) by an equivalent disjunction. Note that the length of all factors $p_{i,j}$ and s_i above is bounded by $|A| \cdot |u|$. Hence, one can guess these traces as well as the numbers $c_j \leq |A|$ (the guess results in a disjunction). We can also guess which of the numbers x_i are zero and which are greater than zero. After these guesses we can verify the independencies $p_{i,j} I p_{k,l}$ $(j < l < k < i)$ and $p_{i,j} I (u^{x_k} s_k)$ $(j < k < i)$, and the identities $s_m = 1$, $s_j \prod_{i=j+1}^{m} p_{i,j} = u^{c_j}$ $(1 \leq j < m)$. If one of them does not hold, the specific guess does not contribute to the disjunction. In this way, we can replace the equation $u^x = y_1 y_2 \cdots y_m$ by a disjunction of formulas of the form

$$\exists x_i > 0\ (i \in K) : x = \sum_{i \in K} x_i + c \wedge \bigwedge_{i \in K} y_i = p_i u^{x_i} s_i \wedge \bigwedge_{i \in [1,m] \setminus K} y_i = p_i s_i,$$

where $K \subseteq [1,m]$, $c \leq |A| \cdot (m-1)$ and the p_i, s_i are concrete traces of length at most $|A| \cdot (m-1) \cdot |u|$. The number of disjuncts in the disjunction will not be important for our purpose.

3.2 Automata for partially commutative closures

In this section, we present several automata constructions that are well-known from the theory of recognizable trace languages [11, Chapter 2]. For our purpose we need upper bounds on the size (the size of an automaton is its number of states) of the constructed automata. In our specific situation we can obtain better bounds than those obtained from the known constructions. Therefore, we present the constructions in detail.

Let us fix an independence alphabet (A, I) and let $\mathcal{A} = (Q, A, \Delta, q_0, F)$ be an NFA over the alphabet A. Then, $\mathcal{A}$ is an *I-diamond NFA* if for all $(a,b) \in I$ and all transitions $(p,a,q), (q,b,r) \in \Delta$ there exists a state q' such that $(p,b,q'), (q',a,r) \in \Delta$. For an I-diamond automaton we have $L(\mathcal{A}) = [L(\mathcal{A})]_I$. The NFA $\mathcal{A}$ is *memorizing* if (i) every state is accessible from the initial state q_0 and (ii) there is a mapping $\alpha\colon Q \to 2^A$ such that for every word $w \in A^*$, if $q_0 \xrightarrow{w}_{\mathcal{A}} q$, then $\alpha(q) = \mathrm{alph}(w)$.

Lemma 3.5 *Let $\mathcal{A}_1$ and $\mathcal{A}_2$ be I-diamond NFA and let n_i be the number of states of $\mathcal{A}_i$. Assume that $\mathcal{A}_2$ is memorizing. Then there exists an I-diamond NFA for $[L(\mathcal{A}_1)L(\mathcal{A}_2)]_I$ with $n_1 \cdot n_2$ many states.*

Proof Let $\mathcal{A}_i = (Q_i, A, \Delta_i, q_{0,i}, F_i)$ for $i \in \{1,2\}$. Let $\alpha_2\colon Q_2 \to 2^A$ be the map witnessing the fact that $\mathcal{A}_2$ is memorizing. Then, let

$$\mathcal{A} = (Q_1 \times Q_2, A, \Delta, \langle q_{0,1}, q_{0,2}\rangle, F_1 \times F_2),$$

where

$$\begin{aligned}\Delta = {} & \{(\langle p_1,p_2\rangle, a, \langle q_1,p_2\rangle) \mid (p_1,a,q_1) \in \Delta_1, aI\alpha_2(p_2)\} \cup \\ & \{(\langle p_1,p_2\rangle, a, \langle p_1,q_2\rangle) \mid (p_2,a,q_2) \in \Delta_2\}.\end{aligned}$$

Claim 1. $\mathcal{A}$ is an I-diamond NFA.

To show this claim, let us consider two consecutive transitions in $\mathcal{A}$ labelled with independent letters. The following four cases can be distinguished, where we assume $(a,b) \in I$ in all four cases:

Case 1. $\langle p_1,p_2\rangle \xrightarrow{a}_{\mathcal{A}} \langle q_1,p_2\rangle \xrightarrow{b}_{\mathcal{A}} \langle r_1,p_2\rangle$, where $p_1 \xrightarrow{a}_{\mathcal{A}_1} q_1 \xrightarrow{b}_{\mathcal{A}_1} r_1$ and $aI\alpha_2(p_2), bI\alpha_2(q_2)$. Since $\mathcal{A}_1$ is an I-diamond NFA, there exists a state $q_1' \in Q_1$ such that $p_1 \xrightarrow{b}_{\mathcal{A}_1} q_1' \xrightarrow{a}_{\mathcal{A}_1} r_1$. We get $\langle p_1,p_2\rangle \xrightarrow{b}_{\mathcal{A}} \langle q_1',p_2\rangle \xrightarrow{a}_{\mathcal{A}} \langle r_1,p_2\rangle$.

Case 2. $\langle p_1,p_2\rangle \xrightarrow{a}_{\mathcal{A}} \langle p_1,q_2\rangle \xrightarrow{b}_{\mathcal{A}} \langle p_1,r_2\rangle$, where $p_2 \xrightarrow{a}_{\mathcal{A}_2} q_2 \xrightarrow{b}_{\mathcal{A}_2} r_2$. We can conclude as in Case 1 using the fact that $\mathcal{A}_2$ is an I-diamond NFA.

Case 3. $\langle p_1, p_2\rangle \xrightarrow{a}_{\mathcal{A}} \langle q_1, p_2\rangle \xrightarrow{b}_{\mathcal{A}} \langle q_1, q_2\rangle$, where $p_1 \xrightarrow{a}_{\mathcal{A}_1} q_1$, $p_2 \xrightarrow{b}_{\mathcal{A}_2} q_2$, and $aI\alpha_2(p_2)$. Since $\alpha_2(q_2) = \alpha_2(p_2) \cup \{b\}$ and $(a,b) \in I$ we have $aI\alpha_2(q_2)$. We get $\langle p_1, p_2\rangle \xrightarrow{b}_{\mathcal{A}} \langle p_1, q_2\rangle \xrightarrow{a}_{\mathcal{A}} \langle q_1, q_2\rangle$.

Case 4. $\langle p_1, p_2\rangle \xrightarrow{a}_{\mathcal{A}} \langle p_1, q_2\rangle \xrightarrow{b}_{\mathcal{A}} \langle q_1, q_2\rangle$, where $p_2 \xrightarrow{a}_{\mathcal{A}_2} q_2$, $p_1 \xrightarrow{b}_{\mathcal{A}_1} q_1$, and $bI\alpha_2(q_2)$. Since $\alpha_2(q_2) = \alpha_2(p_2) \cup \{a\}$, we also have $bI\alpha_2(p_2)$. This implies $\langle p_1, p_2\rangle \xrightarrow{b}_{\mathcal{A}} \langle q_1, p_2\rangle \xrightarrow{a}_{\mathcal{A}} \langle q_1, q_2\rangle$.

This concludes the proof of Claim 1. To show that $L(\mathcal{A}) = [L(\mathcal{A}_1)L(\mathcal{A}_2)]_I$ it suffices to show the following claim:

Claim 2. For all $w \in A^*$, $p_1 \in Q_1$, and $p_2 \in Q_2$, the following two statements are equivalent :

(i) $\langle q_{0,1}, q_{0,2}\rangle \xrightarrow{w}_{\mathcal{A}} \langle p_1, p_2\rangle$
(ii) There are $w_1, w_2 \in A^*$ such that $w \equiv_I w_1 w_2$, $q_{0,1} \xrightarrow{w_1}_{\mathcal{A}_1} p_1$, and $q_{0,2} \xrightarrow{w_2}_{\mathcal{A}_2} p_2$.

Let us first prove that (i) implies (ii). The case $w = \varepsilon$ is clear. Hence, let $w = w'a$. Then there exist $p_1' \in Q_1$, $p_2' \in Q_2$ such that

$$\langle q_{0,1}, q_{0,2}\rangle \xrightarrow{w'}_{\mathcal{A}} \langle p_1', p_2'\rangle \xrightarrow{a}_{\mathcal{A}} \langle p_1, p_2\rangle.$$

By induction, there exists a factorization $w' \equiv_I w_1' w_2'$ such that $q_{0,1} \xrightarrow{w_1'}_{\mathcal{A}_1} p_1'$ and $q_{0,2} \xrightarrow{w_2'}_{\mathcal{A}_2} p_2'$. Note that $\mathrm{alph}(w_2') = \alpha_2(p_2')$. There are two cases:

Case 1. $p_1' \xrightarrow{a}_{\mathcal{A}_1} p_1$, $p_2 = p_2'$, and $aI\alpha_2(p_2')$. Thus, aIw_2'. We get $w = w'a \equiv_I w_1'w_2'a \equiv_I (w_1'a)w_2'$. Let $w_1 = w_1'a$ and $w_2 = w_2'$. We get $q_{0,1} \xrightarrow{w_1}_{\mathcal{A}_1} p_1$ and $q_{0,2} \xrightarrow{w_2}_{\mathcal{A}_2} p_2$.

Case 2. $p_2' \xrightarrow{a}_{\mathcal{A}_2} p_2$ and $p_1 = p_1'$. Let $w_1 = w_1'$ and $w_2 = w_2'a$. Thus, $w = w'a \equiv_I w_1'w_2'a = w_1w_2$. Moreover, we have $q_{0,1} \xrightarrow{w_1}_{\mathcal{A}_1} p_1$ and $q_{0,2} \xrightarrow{w_2}_{\mathcal{A}_2} p_2$.

Let us now prove that (ii) implies (i). Assume that $w \equiv_I w_1w_2$, $q_{0,1} \xrightarrow{w_1}_{\mathcal{A}_1} p_1$, and $q_{0,2} \xrightarrow{w_2}_{\mathcal{A}_2} p_2$. We have to show that $\langle q_{0,1}, q_{0,2}\rangle \xrightarrow{w}_{\mathcal{A}} \langle p_1, p_2\rangle$. But since $\mathcal{A}$ is an I-diamond NFA, it suffices to show that $\langle q_{0,1}, q_{0,2}\rangle \xrightarrow{w_1w_2}_{\mathcal{A}} \langle p_1, p_2\rangle$, which follows directly from the assumption and the definition of $\mathcal{A}$ (note that $\alpha_2(q_{0,2}) = \emptyset$). This concludes the proof of Claim 2 and hence the proof of the lemma. □

In general, for a regular language $L \subseteq A^*$, the partially commutative closure $[L]_I$ is not regular. For instance, if $A = \{a, b\}$ and aIb, then $[(ab)^*]_I$ consists of all words with the same number of a's as b's. On the other hand, it is well known that if $[v]_I$ is a connected trace, then $[v^*]_I$ is regular (in fact, there is a more general result, known as Ochmanski's theorem [11, Section 2.3]). For our purpose we need an upper on the size of an I-diamond NFA for $[v^*]_I$ (with $[v]_I$ connected). Recall that $\rho(u)$ is the number of different prefixes of the trace u.

Lemma 3.6 *Let $u \in A^* \setminus \{\varepsilon\}$ such that the trace $[u]_I$ is connected. There is a memorizing I-diamond NFA for $[u^*]_I$ of size $2 \cdot \rho([u]_I)^{|A|}$.*

Proof The following construction can be found in [43, Proposition 5] for the more general case of the partially commutative closure of a so-called loop-connected automaton. We present the construction in our simplified situation, since the NFA gets slightly smaller.

In the following, we identify u with the trace $[u]_I$. We first define a non-memorizing I-diamond NFA $\mathcal{A}$ for $[u^*]_I$ of size $\rho(u)^{|A|}$. Then, we show that by adding an additional bit to all states, we can get a memorizing I-diamond NFA $\mathcal{A}$ for $[u^*]_I$ of size $2 \cdot \rho(u)^{|A|}$. The idea for the construction of $\mathcal{A}$ is implicitly contained in the proof of Lemma 3.2: Assume that the automaton wants to read a word $w \in [u^*]_I$ and a prefix y_1 of w is already read. Then $[y_1]_I$ must be of the form $u^k s$, where s is a prefix of u^c for some $c \leq |A|$. Moreover, by choosing k maximal, we can assume that u is not a prefix of the trace s.

We define $\mathcal{A} = (Q, A, \Delta, q_0, F)$, where Q is the set of all prefixes s of some trace in u^* such that u is not a prefix of s.

Let us estimate $|Q|$. Observe that if $s \in Q$, then Lemma 3.2 tells us that $s = u^k s'$, where s' is a prefix of u^c with $c \leq |A|$. Since u is not a prefix of s, we have $k = 0$ and hence $s = s'$ is a prefix of u^c. According to Levi's lemma, s is of the form $u_1 u_2 \cdots u_c$ such that every trace u_i is a non-empty prefix of u. Hence $|Q| \leq \rho(u)^{|A|}$.

Note that Q is prefix closed. Moreover, if $|u| = 1$, then 1 is the only state. The initial state as well as the final state is the empty trace 1. The set of transition tuples is

$$\Delta = \{(s, a, sa) \mid s, sa \in Q, a \in A\} \cup \{(s, a, t) \mid s, t \in Q, sa = ut \text{ in } \mathbb{M}(A, I)\}.$$

Claim 1. $\mathcal{A}$ is an I-diamond NFA.

We can distinguish the following four cases, where $(a, b) \in I$ in all four cases:

Case 1. $(s, a, sa), (sa, b, sab) \in \Delta$. Since $sab = sba \in Q$, we must have $sb \in Q$. Hence, we have $(s, b, sb), (sb, a, sba) \in \Delta$.

Case 2. $(s, a, t), (t, b, v) \in \Delta$, where $sa = ut$ and $tb = uv$. From $sa = ut$ and the fact that u is not a prefix of s we obtain with Levi's lemma the factorizations $s = u't$ and $u = u'a$ with aIt. From aIt and $(a, b) \in I$ we get $aItb$, in contradiction to $tb = uv = u'av$. Hence, Case 2 cannot occur.

Case 3. $(s, a, t), (t, b, tb) \in \Delta$, where $sa = ut$. As above, we get factorizations $s = u't$ and $u = u'a$ with aIt. We claim that $sb \in Q$. First, u is not a prefix of sb: If $sb = uv$ for some trace v, then $u'tb = sb = uv = u'av$. Hence $tb = av$, in contradiction to $aItb$.

It remains to show that sb is a prefix of a trace in u^*. Since $tb \in Q$ there exists a trace x such that $tbx \in u^*$. Hence, $sbax = sabx = utbx \in u^*$, i.e., sb is a prefix of a trace in u^*. Thus, $sb \in Q$ and hence $(s, b, sb) \in \Delta$. Moreover $sba = sab = utb$. Thus, $(sb, a, tb) \in \Delta$.

Case 4. $(s,a,sa),(sa,b,t) \in \Delta$, where $sab = ut$. We get $sa = u't$, $u = u'b$ and bIt for some trace u'. We distinguish two subcases. First assume that u is not a prefix of sb. We claim that $sb \in Q$. Since $t \in Q$, there exist a trace x with $tx \in u^*$. Hence, $sbax = sabx = utx \in u^*$. Thus, sb is a prefix of a trace in u^* and does not have u as a prefix. Hence $sb \in Q$ and $(s,b,sb) \in \Delta$. Moreover, $sba = sab = ut$, and thus $(sb,a,t) \in \Delta$.

Now assume that u is a prefix of sb. Let $sb = uv$. Since u is not a prefix of $s \in Q$, we get $s = u''v$, $u = u''b$ and bIv for some trace u''. Hence, $u''b = u = u'b$, i.e., $u'' = u'$ and $s = u'v$. Thus, $u't = sa = u'va$, which implies $t = va$. Since $t \in Q$, we have $v \in Q$. We get $(s,b,v),(v,a,t) \in \Delta$. This concludes the proof of Claim 1.

The following claim shows that $L(\mathcal{A}) = [u^*]_I$:

Claim 2. For every state $s \in Q$ and every $w \in A^*$ the following two statements are equivalent:

(i) $1 \xrightarrow{w}_{\mathcal{A}} s$
(ii) $[w]_I = u^k s$ for some $k \geq 0$

Let us first show by induction on $|w|$ that (i) implies (ii). The case $w = \varepsilon$ is clear. So, assume that $w = w'a$. There must exist a state $s' \in Q$ such that

$$1 \xrightarrow{w'}_{\mathcal{A}} s' \xrightarrow{a}_{\mathcal{A}} s.$$

By induction, we get $[w']_I = u^\ell s'$ for some $\ell \geq 0$. The definition of the transitions of $\mathcal{A}$ implies that $[w]_I = [w'a]_I = u^\ell s'a = u^k s$, where $k \in \{\ell, \ell+1\}$.

For the direction from (ii) to (i) assume that $[w]_I = u^k s$ for some $k \geq 0$. We have to show that $1 \xrightarrow{w}_{\mathcal{A}} s$. Let $s' \in A^*$ such that $s = [s']_I$. Hence, $w \equiv_I u^k s'$. Since $\mathcal{A}$ is an I-diamond NFA, it suffices to show that $1 \xrightarrow{u^k s'}_{\mathcal{A}} s$. But this follows directly from the definition of $\mathcal{A}$.

To make $\mathcal{A}$ memorizing, we first keep only those states that are accessible from the initial state 1. Then, we add an extra bit to every state that indicates whether we have already seen a completed occurrence of u. Thus, the new set of states is $Q \times \{0,1\}$, the initial state is the pair $(1,0)$, and the final states are $(1,0)$ and $(1,1)$. The new set of transitions is

$$\{((s,i),a,(t,i)) \mid (s,a,t) \in \Delta\} \cup \{((s,i),a,(t,1)) \mid s,t \in Q, sa = ut, i \in \{0,1\}\}.$$

Then, we can define the α-mapping by $\alpha(s,i) = \mathrm{alph}(u^i s)$. The resulting NFA is still an I-diamond NFA. □

A direct consequence of Lemma 3.5 and 3.6 is:

Lemma 3.7 *Let $p,u,s \in A^*$ with $u \neq \varepsilon$ and $[u]_I$ connected. There is an NFA for $[pu^*s]_I$ of size $2 \cdot \rho([p]_I) \cdot \rho([s]_I) \cdot \rho([u]_I)^{|A|}$.*

Proof We first construct an I-diamond NFA for $[p]_I$ (which is identified here with the set of words $\{w \in A^* \mid w \equiv_I p\}$) with $\rho([p]_I)$ many states by taking the set of all prefixes of $[p]_I$ as states. Then, we construct a memorizing I-diamond NFA for $[u^*]_I$ with $2{\cdot}\rho([u]_I)^{|A|}$ states using Lemma 3.6. By Lemma 3.5 we get an I-diamond automaton for $[pu^*]_I$ with $2 \cdot \rho([p]_I) \cdot \rho([u]_I)^{|A|}$ many states. Finally, we construct an I-diamond NFA for $[s]_I$ with $\rho([s]_I)$ many states by taking the set of all prefixes of $[s]_I$ as states. This NFA is also memorizing. Hence, we can apply Lemma 3.5 to get an NFA for $[pu^*s]_I$ with $2 \cdot \rho([p]_I) \cdot \rho([s]_I) \cdot \rho([u]_I)^{|A|}$ many states. □

The main lemma from this section that will be needed later is:

Lemma 3.8 *Let $p, q, u, v, s, t \in \mathbb{M}(A, I)$ with $u \neq 1$ and $v \neq 1$ connected. Let $m = \max\{\rho(p), \rho(q), \rho(s), \rho(t)\}$ and $n = \max\{\rho(u), \rho(v)\}$. Then the set*

$$L(p, u, s, q, v, t) := \{(x, y) \in \mathbb{N} \times \mathbb{N} \mid pu^x s = qv^y t\}$$

is semilinear and is a union of $\mathcal{O}(m^8 \cdot n^{4|A|})$ many linear sets of the form $\{(a + bz, c + dz) \mid z \in \mathbb{N}\}$ with $a, b, c, d \in \mathcal{O}(m^8 \cdot n^{4|A|})$.

Proof We identify the traces p, q, u, v, s, t with words representing these traces. By Lemma 3.6 there exists an NFA for $[pu^*s]_I$ of size

$$k = 2 \cdot \rho(p) \cdot \rho(s) \cdot \rho(u)^{|A|} \leq 2 \cdot m^2 \cdot n^{|A|}$$

and an NFA for $[qv^*t]_I$ of size

$$\ell = 2 \cdot \rho(q) \cdot \rho(t) \cdot \rho(v)^{|A|} \leq 2 \cdot m^2 \cdot n^{|A|}.$$

Then, we obtain an NFA $\mathcal{A}$ for $L = [pu^*s]_I \cap [qv^*t]_I$ with $k \cdot \ell$ states. We are only interested in the length of words from L. Hence, we replace in $\mathcal{A}$ every transition label by the symbol a. The resulting NFA $\mathcal{B}$ is defined over a unary alphabet. Let $P = \{n \mid a^n \in L(\mathcal{B})\}$. By [49, Theorem 1], the set P can be written as a union

$$P = \bigcup_{i=1}^{r} \{b_i + c_i \cdot z \mid z \in \mathbb{N}\}$$

with $r \in \mathcal{O}(k^2\ell^2) \subseteq \mathcal{O}(m^8 \cdot n^{4|A|})$ and $b_i, c_i \in \mathcal{O}(k^2\ell^2) \subseteq \mathcal{O}(m^8 \cdot n^{4|A|})$. For every $1 \leq i \leq r$ and $z \in \mathbb{N}$ there must exist a pair $(x, y) \in \mathbb{N} \times \mathbb{N}$ such that

$$b_i + c_i \cdot z = |ps| + |u| \cdot x = |qt| + |v| \cdot y.$$

In particular, $b_i \geq |ps|$, $b_i \geq |qt|$, $|u|$ divides $b_i - |ps|$ and c_i, and $|v|$ divides $b_i - |qt|$ and c_i. We get:

$$L(p, u, s, q, v, t) = \bigcup_{i=1}^{r} \left\{ \left(\frac{b_i - |ps|}{|u|} + \frac{c_i}{|u|} \cdot z, \frac{b_i - |qt|}{|v|} + \frac{c_i}{|v|} \cdot z \right) \;\middle|\; z \in \mathbb{N} \right\}$$

This shows the lemma. □

3.3 Linear Diophantine equations

We will also need a bound on the norm of a smallest vector in a certain kind of semilinear sets. We will easily obtain this bound from a result from [19].

Lemma 3.9 *Let $A \in \mathbb{Z}^{n\times m}$, $\overline{a} \in \mathbb{Z}^n$, $C \in \mathbb{N}^{k\times m}$, $\overline{c} \in \mathbb{N}^k$. Let β be an upper bound for the absolute value of all entries in A, $\overline{a}$, C, $\overline{c}$. The set*

$$L = \{C\overline{z} + \overline{c} \mid \overline{z} \in \mathbb{N}^m, A\overline{z} = \overline{a}\} \subseteq \mathbb{N}^k \tag{5}$$

is semilinear. Moreover, if $L \neq \emptyset$ then there is $y \in L$ with

$$\|y\|_\infty \leq \beta + (\sqrt{m})^n \cdot m \cdot (m+1) \cdot \beta^{n+1}.$$

Proof Semilinearity of L is clear since the set is Presburger-definable. For the size bound, we use a result from [19] to bound the size of a smallest positive solution of the system $A\overline{z} = \overline{a}$. Let $A \in \mathbb{Z}^{n\times m}$, $B \in \mathbb{Z}^{p\times m}$, $\overline{a} \in \mathbb{Z}^{n\times 1}$, $\overline{b} \in \mathbb{Z}^{p\times 1}$. Let $r = \mathrm{rank}(A)$, and $s = \mathrm{rank}\begin{pmatrix} A \\ B \end{pmatrix}$. Let M be an upper bound on the absolute values of all $(s-1)\times(s-1)$- or $(s\times s)$-subdeterminants of the $(n+p)\times(m+1)$-matrix $\begin{pmatrix} A\ \overline{a} \\ B\ \overline{b} \end{pmatrix}$, which are formed with at least r rows from the matrix $(A\ \overline{a})$. Then by the main result of [19], the system $A\overline{z} = \overline{a}$, $B\overline{z} \geq \overline{b}$ has an integer solution if and only if it has an integer solution $\overline{z}$ such that the absolute value of every entry of $\overline{z}$ is bounded by $(m+1)M$.

In our situation, we set $p = m$, B is the m-dimensional identity matrix, and $\overline{b}$ is the vector with all entries equal to zero (then $B\overline{z} \geq \overline{b}$ expresses that all entries of z are positive). Since $\begin{pmatrix} A \\ B \end{pmatrix}$ is an $(n+m)\times m$-matrix we get $s = \mathrm{rank}\begin{pmatrix} A \\ B \end{pmatrix} \leq m$. We claim that the absolute values of all $(s\times s)$-subdeterminants (and also all $(s-1)\times(s-1)$-subdeterminants) of the matrix $\begin{pmatrix} A\ \overline{a} \\ B\ \overline{b} \end{pmatrix}$ are bounded by $(\sqrt{m})^n \cdot \beta^n$. To see this, select s rows and s columns from $\begin{pmatrix} A\ \overline{a} \\ B\ \overline{b} \end{pmatrix}$ and consider the resulting submatrix D.

Let $d_1, \ldots, d_s$ be the row vectors of D. By Hadamard's inequality we have

$$\det(D) \leq \prod_{i=1}^{s} \|d_i\|_2.$$

Assume that the row vectors $d_1, \ldots, d_{s'}$ ($s' \leq n$) of D are from the $n\times(m+1)$-submatrix $(A, \overline{a})$. The remaining row vectors $d_{s'+1}, \ldots, d_s$ of D are from $(B, \overline{b})$. Then, every d_i ($s'+1 \leq i \leq s$) is a zero vector or a unit vector and hence has Euklidean norm 0 or 1. We therefore have

$$\det(D) \leq \prod_{i=1}^{s} \|d_i\|_2 \leq \prod_{i=1}^{s'} \|d_i\|_2 \leq (\sqrt{m})^n \cdot \beta^n.$$

It follows that if $A\overline{z} = \overline{a}$ has a positive solution, then it has a positive solution where every entry is bounded by $(m+1) \cdot (\sqrt{m})^n \cdot \beta^n$.

By substituting every entry of $\overline{z}$ by $(\sqrt{m})^n \cdot \beta^n$ in $C\overline{z} + \overline{c}$, it follows that if the set L in (5) is non-empty, then it contains a vector with all entries bounded by $\beta + (\sqrt{m})^n \cdot m \cdot (m+1) \cdot \beta^{n+1}$. □

3.4 Reduction from graph groups to trace monoids

As usual, we fix an independence alphabet (A, I). In the following we will consider reduction rules on sequences of traces. For better readability we separate the consecutive traces in such a sequence by commas. Let $u_1, u_2, \ldots, u_n \in \mathrm{IRR}(A^{\pm 1}, I)$ be irreducible traces. The sequence $u_1, u_2, \ldots, u_n$ is *I-freely reducible* if the sequence $u_1, u_2, \ldots, u_n$ can be reduced to the empty sequence ε by the following rules:

- $u_i, u_j \to u_j, u_i$ if $u_i I u_j$
- $u_i, u_j \to \varepsilon$ if $u_i = u_j^{-1}$ in $\mathbb{M}(A^{\pm 1}, I)$
- $u_i \to \varepsilon$ if $u_i = \varepsilon$.

A concrete sequence of these rewrite steps leading to the empty sequence is a *reduction* of the sequence $u_1, u_2, \ldots, u_n$. Such a reduction can be seen as a witness for the fact that $u_1 u_2 \cdots u_n = 1$ in $\mathbb{G}(A, I)$. On the other hand, $u_1 u_2 \cdots u_n = 1$ does not necessarily imply that $u_1, u_2, \ldots, u_n$ has a reduction. For instance, the sequence a^{-1}, ab, b^{-1} has no reduction. But we can show that every sequence which multiplies to 1 in $\mathbb{G}(A, I)$ can be refined (by factorizing the elements of the sequence) such that the resulting refined sequence has a reduction. For getting an NP-algorithm, it is important to bound the length of the refined sequence exponentially in the length of the initial sequence.

Lemma 3.10 *Let $n \geq 2$ and $u_1, u_2, \ldots, u_n \in \mathrm{IRR}(A^{\pm 1}, I)$. If $u_1 u_2 \cdots u_n = 1$ in $\mathbb{G}(A, I)$, then there exist factorizations $u_i = u_{i,1} \cdots u_{i,k_i}$ in $\mathbb{M}(A^{\pm 1}, I)$ such that the sequence*

$$u_{1,1}, \ldots, u_{1,k_1},\ u_{2,1}, \ldots, u_{2,k_2},\ \ldots, u_{n,1}, \ldots, u_{n,k_n}$$

is I-freely reducible. Moreover, $\sum_{i=1}^{n} k_i \leq 2^n - 2$.

Proof We prove the lemma by induction on n. The case $n = 2$ is trivial (we must have $u_2 = u_1^{-1}$). If $n \geq 3$ then by Lemma 2.5 we can factorize u_1 and u_2 as $u_1 = ps$ and $u_2 = s^{-1}t$ in $\mathbb{M}(A^{\pm 1}, I)$ such that $v := pt$ is irreducible. Hence, $v u_3 \cdots u_n = 1$ in $\mathbb{G}(A, I)$. By induction, we obtain factorizations $pt = v = v_1 \cdots v_k$ and $u_i = v_{i,1} \cdots v_{i,k_i}$ $(3 \leq i \leq n)$ in $\mathbb{M}(A^{\pm 1}, I)$ such that the sequence

$$v_1, \ldots, v_k,\ v_{3,1}, \ldots, v_{3,k_3}, \ldots, v_{n,1}, \ldots, v_{n,k_n} \tag{6}$$

is I-freely reducible. Moreover,

$$k + \sum_{i=3}^{n} k_i \leq 2^{n-1} - 2.$$

By applying Levi's lemma to the trace identity $pt = v_1 v_2 \cdots v_k$, we obtain factorizations $v_i = u_{i,1}u_{i,2}$ in $\mathbb{M}(A^{\pm 1}, I)$ such that $p = u_{1,1} \cdots u_{k,1}$, $t = u_{1,2} \cdots u_{k,2}$, and $u_{i,2} I u_{j,1}$ for $1 \leq i < j \leq k$.

Fix a concrete reduction of the sequence (6). We now consider the following sequence

$$u_{1,1}, \ldots, u_{k,1}, s,\ s^{-1}, u_{1,2}, \ldots, u_{k,2},\ \tilde{v}_{3,1}, \ldots, \tilde{v}_{3,k_3}, \ldots, \tilde{v}_{n,1}, \ldots, \tilde{v}_{n,k_n}, \quad (7)$$

where the subsequence $\tilde{v}_{i,j}$ is $u_{l,2}^{-1}, u_{l,1}^{-1}$ if $v_{i,j}$ cancels against v_l in our fixed reduction of (6) (which, in particular, implies that $v_{i,j} = v_l^{-1} = u_{l,2}^{-1} u_{l,1}^{-1}$ in $\mathbb{M}(A^{\pm 1}, I)$). Otherwise (i.e., if $v_{i,j}$ does not cancel against any v_l in our fixed reduction), we set $\tilde{v}_{i,j} = v_{i,j}$.

Note that $u_{1,1} \cdots u_{k,1} s = ps = u_1$, $s^{-1} u_{1,2} \cdots u_{k,2} = s^{-1} t = u_2$ and the concatenation of all traces in $\tilde{v}_{i,1}, \ldots, \tilde{v}_{i,k_i}$ is u_i for $3 \leq i \leq n$. Hence, it remains to show that the sequence (7) is I-freely reducible. First of all, $u_{1,1}, \ldots, u_{k,1}, s, s^{-1}, u_{1,2}, \ldots, u_{k,2}$ reduces to $u_{1,1}, \ldots, u_{k,1}, u_{1,2}, \ldots, u_{k,2}$, which can be rearranged to $u_{1,1}, u_{1,2}, u_{2,1}, u_{2,2}, \ldots, u_{k,1}, u_{k,2}$ using the fact that $u_{i,2} I u_{j,1}$ for $1 \leq i < j \leq k$. Finally, the sequence

$$u_{1,1}u_{1,2}, u_{2,1}u_{2,2}, \ldots, u_{k,1}u_{k,2}, \tilde{v}_{3,1}, \ldots, \tilde{v}_{3,k_3}, \ldots, \tilde{v}_{n,1}, \ldots, \tilde{v}_{n,k_n}$$

is I-freely reducible. The definition of $\tilde{v}_{i,j}$ allows to basically apply the fixed reduction of (6) to this sequence.

The number of traces in the sequence (7) can be estimated as

$$2k + 2 + 2 \cdot \sum_{i=3}^{n} k_i \leq 2 \cdot (2^{n-1} - 2) + 2 = 2^n - 2.$$

This concludes the proof of the lemma. □

3.5 Semilinearity, exponential bounds, and NP-membership

We now come to the main technical result of Section 3. Let $\alpha \leq |A|$ be the size of a largest clique of the dependence alphabet (A, D) corresponding to (A, I).

Theorem 3.11 *Let $u_1, u_2, \ldots, u_n \in \mathbb{G}(A, I) \setminus \{1\}$, $v_0, v_1, \ldots, v_n \in \mathbb{G}(A, I)$ and let $x_1, \ldots, x_n$ be variables (we may have $x_i = x_j$ for $i \neq j$) ranging over $\mathbb{N}$. Then, the set of solutions of the exponent equation*

$$v_0 u_1^{x_1} v_1 u_2^{x_2} v_2 \cdots u_n^{x_n} v_n = 1$$

is effectively semilinear. Moreover, if there is a solution, then there is a solution with $x_i \in \mathcal{O}(2^{3(\alpha n)^2 + 7\alpha n} \cdot \mu^{8\alpha(n+1)} \cdot \nu^{8\alpha|A|(n+1)})$, where

- $\mu \in \mathcal{O}(|A|^\alpha \cdot 2^{2\alpha^2 n} \cdot \lambda^\alpha)$,
- $\nu \in \mathcal{O}(\lambda^\alpha)$, *and*
- $\lambda = \max\{|u_1|, |u_2|, \ldots, |u_n|, |v_0|, |v_1|, \ldots, |v_n|\}$.

Proof Let us choose irreducible traces for $u_1, u_2, \ldots, u_n, v_0, v_1, \ldots, v_n$; we denote these traces with the same letters as the group elements. A trace u is called *cyclically reduced* if there do not exist $a \in A^{\pm 1}$ and v such that $u = ava^{-1}$. For every trace u there exist unique traces p, w such that $u = pwp^{-1}$ and w is cyclically reduced (since the reduction relation $a^{-1}xa \to x$ is terminating and confluent [12, Lemma 16]). These traces p and w can be computed in polynomial time. Note that for a cyclically reduced irreducible trace w, every power w^n is irreducible. Let $u_i = p_i w_i p_i^{-1}$ with w_i cyclically reduced. Note that w_i cannot be the empty trace since $u_i \neq 1$ in $\mathbb{G}(A, I)$. By replacing every $u_i^{x_i}$ by $p_i w_i^{x_i} p_i^{-1}$, we can assume that all u_i are cyclically reduced, irreducible, and non-empty. In case one of the traces u_i is not connected, we can write u_i as $u_i = u_{i,1}u_{i,2}$ with $u_{i,1} I u_{i,2}$ and $u_{i,1} \neq 1 \neq u_{i,2}$. Thus, we can replace the power $u_i^{x_i}$ by $u_{i,1}^{x_i} u_{i,2}^{x_i}$. Note that $u_{i,1}$ and $u_{i,2}$ are still irreducible and cyclically reduced. By doing this, the number n from the theorem multiplies by at most α (which is the maximal number of pairwise independent letters). In order to keep the notation simple we still use the letter n for the number of u_i, but at the end of the proof we have to multiply n by α in the derived bound. Hence, for the further proof we can assume that all u_i are connected, irreducible and cyclically reduced. Let λ be the maximal length of one of the traces $u_1, u_2, \ldots, u_n, v_0, v_1, \ldots, v_n$, which does not increase by the above preprocessing.

We now apply Lemma 3.10 to the equation

$$v_0 u_1^{x_1} v_1 u_2^{x_2} v_2 \cdots u_n^{x_n} v_n = 1, \tag{8}$$

where every $u_i^{x_i}$ is viewed as a single factor. Note that by our preprocessing, all factors $u_1^{x_1}, u_2^{x_2}, \ldots, u_n^{x_n}, v_0, \ldots, v_n$ are irreducible (for all choices of the x_i). By taking a disjunction over (i) all possible factorizations of the $2n+1$ factors $u_1^{x_1}, u_2^{x_2}, \ldots, u_n^{x_n}, v_0, \ldots, v_n$ into totally at most $2^{2n+1} - 2$ factors and (ii) all possible reductions of the resulting refined factorization of $v_0 u_1^{x_1} v_1 u_2^{x_2} v_2 \cdots u_n^{x_n} v_n$, it follows that (8) is equivalent to a disjunction of statements of the following form: There exist traces $y_{i,1}, \ldots, y_{i,k_i}$ $(1 \leq i \leq n)$ and $z_{i,1}, \ldots, z_{i,l_i}$ $(0 \leq i \leq n)$ such that in $\mathbb{M}(A^{\pm 1}, I)$ the following hold:

(a) $u_i^{x_i} = y_{i,1} \cdots y_{i,k_i}$ $(1 \leq i \leq n)$
(b) $v_i = z_{i,1} \cdots z_{i,l_i}$ $(0 \leq i \leq n)$
(c) $y_{i,j} I y_{k,l}$ for all $(i,j,k,l) \in J_1$
(d) $y_{i,j} I z_{k,l}$ for all $(i,j,k,l) \in J_2$
(e) $z_{i,j} I z_{k,l}$ for all $(i,j,k,l) \in J_3$
(f) $y_{i,j} = y_{k,l}^{-1}$ for all $(i,j,k,l) \in M_1$
(g) $y_{i,j} = z_{k,l}^{-1}$ for all $(i,j,k,l) \in M_2$
(h) $z_{i,j} = z_{k,l}^{-1}$ for all $(i,j,k,l) \in M_3$

Here, the numbers k_i and l_i sum up to at most $2^{2n+1} - 2$ (hence, some k_i can be exponentially large, whereas l_i can be bound by the length of v_i, which is at most λ). The tuple sets J_1, J_2, J_3 collect all independencies between the factors $y_{i,j}$, $z_{k,l}$ that are necessary to carry out the chosen reduction of the

refined left-hand side in (8). Similarly, the tuple sets M_1, M_2, M_3 tell us which of the factors $y_{i,j}$, $z_{k,l}$ cancels against which of the factors $y_{i,j}$, $z_{k,l}$ in our chosen reduction of the refined left-hand side in (8). Note that every factor $y_{i,j}$ (resp., $z_{k,l}$) appears in exactly one of the identities (f), (g), (h) (since in the reduction every factor cancels against another unique factor). Let us also remark that in the rest of proof we no longer work in the graph group $\mathbb{G}(A, I)$. All statements refer to the trace monoid $\mathbb{M}(A^{\pm 1}, I)$.

Next, we simplify our statements. Since the v_i are concrete traces (of length at most λ), we can take a disjunction over all possible factorizations $v_i = v_{i,1} \cdots v_{i,l_i}$ $(1 \leq i \leq n+1)$ such that $v_{i,j} I v_{k,l}$ for all $(i,j,k,l) \in J_3$ and $v_{i,j} = v_{k,l}^{-1}$ for all $(i,j,k,l) \in M_3$. This allows to replace every variable $z_{i,j}$ by a concrete trace $v_{i,j}$. Statements of the form $v_{i,j} I v_{k,l}$ and $v_{i,j} = v_{k,l}^{-1}$ can, of course, be eliminated. Moreover, if there is an identity $y_{i,j} = v_{k,l}^{-1}$ then we can replace the variable $y_{i,j}$ by the concrete trace $v_{k,l}^{-1}$ (of length at most λ).

In the next step, we eliminate trace equations of the form $u_i^{x_i} = y_{i,1} \cdots y_{i,k_i}$ $(1 \leq i \leq n)$. Note that some of the variables $y_{i,j}$ might have been replaced by concrete traces of length at most λ. We apply to each of these trace equations Lemma 3.3, or better Remark 3.4. This allows us to replace every equation $u_i^{x_i} = y_{i,1} \cdots y_{i,k_i}$ $(1 \leq i \leq n)$ by a disjunction of statements of the following form: There exist numbers $x_{i,j} > 0$ $(1 \leq i \leq n,\ j \in K_i)$ such that

- $x_i = c_i + \sum_{j \in K_i} x_{i,j}$ for all $1 \leq i \leq n$,
- $y_{i,j} = p_{i,j} u_i^{x_{i,j}} s_{i,j}$ for all $1 \leq i \leq n$, $j \in K_i$,
- $y_{i,j} = p_{i,j} s_{i,j}$ for all $1 \leq i \leq n$, $j \in [1, k_i] \setminus K_i$.

Here, $K_i \subseteq [1, k_i]$, the c_i are concrete numbers with $c_i \leq |A| \cdot (k_i - 1)$, and the $p_{i,j}, s_{i,j}$ are concrete traces of length at most $|A| \cdot (k_i - 1) \cdot |u_i| \leq |A| \cdot (2^{2n+1} - 3) \cdot \lambda$. Hence, the lengths of these traces can be exponential in n.

Note that since $x_i > 0$, we know the alphabet of $y_{i,j} = p_{i,j} u_i^{x_{i,j}} s_{i,j}$ (resp., $y_{i,j} = p_{i,j} s_{i,j}$). This allows us to replace all independencies of the form $y_{i,j} I y_{k,l}$ for $(i,j,k,l) \in J_1$ (see (c)) and $y_{i,j} I z_{k,l}$ for $(i,j,k,l) \in J_2$ (see (d)) by concrete truth values. Note that all variables $z_{k,l}$ have already been replaced by concrete traces. If $y_{i,j}$ was already replaced by a concrete trace, then we can determine from an equation $y_{i,j} = p_{i,j} u_i^{x_{i,j}} s_{i,j}$ the exponent $x_{i,j}$. Since $y_{i,j}$ was replaced by a trace of length at most λ (a small number), we get $x_{i,j} \leq \lambda$, and we can replace $x_{i,j}$ in $x_i = \sum_{j \in K_i} x_{i,j} + c_i$ by a concrete number of size at most λ. Finally, if $y_{i,j}$ was replaced by a concrete trace, and we have an equation of the form $y_{i,j} = p_{i,j} s_{i,j}$, then the resulting identity is either true or false and can be eliminated.

After this step, we obtain a disjunction of statements of the following form: There exist numbers $x_{i,j} > 0$ $(1 \leq i \leq n,\ j \in K_i')$ such that

(a') $x_i = c_i + \sum_{j \in K_i'} x_{i,j}$ for all $1 \leq i \leq n$, and
(b') $p_{i,j} u_i^{x_{i,j}} s_{i,j} = s_{k,l}^{-1} (u_k^{-1})^{x_{k,l}} p_{k,l}^{-1}$ for all $(i,j,k,l) \in M$.

Here, $K_i' \subseteq K_i$ is a set of size at most $k_i \leq 2^{2n+1} - 2$, $c_i \leq |A| \cdot (k_i - 1) + \lambda \cdot k_i < (|A| + \lambda) \cdot (2^{2n+1} - 2)$, and the $p_{i,j}, s_{i,j}$ are concrete traces of length at most

$|A| \cdot (2^{2n+1} - 3) \cdot \lambda$. The set M specifies a matching in the sense that for every exponent $x_{a,b}$ ($1 \leq a \leq n$, $b \in K_i'$) there is a unique $(i,j,k,l) \in M$ such that $(i,j) = (a,b)$ or $(k,l) = (a,b)$.

We now apply Lemma 3.8 to the trace identities

$$p_{i,j} u_i^{x_{i,j}} s_{i,j} = s_{k,l}^{-1} (u_k^{-1})^{x_{k,l}} p_{k,l}^{-1}.$$

Each such identity can be replaced by a disjunction of constraints

$$(x_{i,j}, x_{k,l}) \in \{(a_{i,j,k,l} + b_{i,j,k,l} \cdot z_{i,j,k,l}, c_{i,j,k,l} + d_{i,j,k,l} \cdot z_{i,j,k,l}) \mid z_{i,j,k,l} \in \mathbb{N}\}.$$

For the numbers $a_{i,j,k,l}, b_{i,j,k,l}, c_{i,j,k,l}, d_{i,j,k,l}$ we obtain the bound

$$a_{i,j,k,l}, b_{i,j,k,l}, c_{i,j,k,l}, d_{i,j,k,l} \in \mathcal{O}(\mu^8 \cdot \nu^{8|A|})$$

(the alphabet of the traces is $A^{\pm 1}$ which has size $2|A|$, therefore, we have to multiply in Lemma 3.8 $|A|$ by 2), where, by Lemma 2.4,

$$\mu = \max\{\rho(p_{i,j}), \rho(p_{k,l}), \rho(s_{i,j}), \rho(s_{k,l})\} \in \mathcal{O}(|A|^\alpha \cdot 2^{2\alpha n} \cdot \lambda^\alpha) \tag{9}$$

and

$$\nu = \max\{\rho(u_i), \rho(u_k)\} \in \mathcal{O}(\lambda^\alpha). \tag{10}$$

Note that $\rho(t) = \rho(t^{-1})$ for every trace t. The above condition (a') for x_i can be now written as

$$x_i = c_i + \sum_{(i,j,k,l) \in M} (a_{i,j,k,l} + b_{i,j,k,l} \cdot z_{i,j,k,l}) + \sum_{(k,l,i,j) \in M} (c_{k,l,i,j} + d_{k,l,i,j} \cdot z_{k,l,i,j}).$$

Note that the two sums in this equation contain in total $|K_i'| \leq 2^{2n+1}$ many summands (since for every $j \in K_i'$ there is a unique pair (k,l) with $(i,j,k,l) \in M$ or $(k,l,i,j) \in M$).

Hence, after a renaming of symbols, the initial equation (8) becomes equivalent to a finite disjunction of statements of the form: There exist $z_1, \ldots, z_m \in \mathbb{N}$ (these z_i are the above $z_{i,j,k,l}$ and $m = \max_i |K_i'|$) such that

$$x_i = a_i + \sum_{j=1}^{m} a_{i,j} z_j \text{ for all } 1 \leq i \leq n. \tag{11}$$

Moreover, we have the following size bounds:

- $m = \max_i |K_i'| \leq 2^{2n+1}$,
- $a_i \in \mathcal{O}(c_i + |K_i'| \cdot \mu^8 \cdot \nu^{8|A|}) \subseteq \mathcal{O}(2^{2n}(|A| + \lambda + \mu^8 \cdot \nu^{8|A|})) \subseteq \mathcal{O}(2^{2n} \cdot \mu^8 \cdot \nu^{8|A|})$
- $a_{i,j} \in \mathcal{O}(\mu^8 \cdot \nu^{8|A|})$

Recall that some of the variables x_i can be identical. W.l.o.g. assume that $x_1, \ldots, x_k$ are pairwise different and for all $k+1 \leq i \leq n$, $x_i = x_{f(i)}$, where $f\colon [k+1, n] \to [1, k]$. Then, the system of equations (11) is equivalent to

$$x_i = a_i + \sum_{j=1}^{m} a_{i,j} z_j \text{ for all } 1 \leq i \leq k$$

$$a_i - a_{f(i)} = \sum_{j=1}^{m} (a_{f(i),j} - a_{i,j}) z_j \text{ for all } k+1 \leq i \leq n.$$

The set of all $(x_1, \ldots, x_k) \in \mathbb{N}^k$ for which there exist $z_1, \ldots, z_m \in \mathbb{N}$ satisfying these equations is semilinear by Lemma 3.9, and if it is non-empty then it contains a vector $(x_1, \ldots, x_k) \in \mathbb{N}^k$ such that (note that $(\sqrt{m})^n \in \mathcal{O}(2^{n^2+n})$)

$$x_i \in \mathcal{O}((\sqrt{m})^n \cdot m^2 \cdot 2^{2n(n+1)} \cdot \mu^{8(n+1)} \cdot \nu^{8|A|(n+1)}) \tag{12}$$

$$\subseteq \mathcal{O}(2^{3n^2+7n} \cdot \mu^{8(n+1)} \cdot \nu^{8|A|(n+1)}). \tag{13}$$

Recall that in this bound we have to replace n by $\alpha \cdot n$ due to the initial preprocessing. This proves the theorem. □

Proof of Theorem 3.1. Consider a compressed exponent equation

$$E = (v_0 u_1^{x_1} v_1 u_2^{x_2} v_2 \cdots u_n^{x_n} v_n = 1),$$

where $u_i = \mathrm{val}(\mathcal{G}_i)$ and $v_i = \mathrm{val}(\mathcal{H}_i)$ for given SLPs $\mathcal{G}_1, \ldots, \mathcal{G}_n, \mathcal{H}_0, \ldots, \mathcal{H}_n$. Let $m = \max\{|\mathcal{G}_1|, \ldots, |\mathcal{G}_n|, |\mathcal{H}_0|, \ldots, |\mathcal{H}_n|\}$. By Theorem 3.11 we know that if there exists a solution for E then there exists a solution $(x_1, \ldots, x_n)$ with $x_i \in \mathcal{O}(2^{3(\alpha n)^2+7\alpha n} \cdot \mu^{8\alpha(n+1)} \cdot \nu^{8\alpha|A|(n+1)})$, where

- $\mu \in \mathcal{O}(|A|^\alpha \cdot 2^{2\alpha^2 n} \cdot \lambda^\alpha)$,
- $\nu \in \mathcal{O}(\lambda^\alpha)$,
- $\lambda = \max\{|u_1|, |u_2|, \ldots, |u_n|, |v_0|, |v_1|, \ldots, |v_n|\} \in 2^{\mathcal{O}(m)}$, and
- $\alpha \leq |A|$.

Note that the bound on the x_i is exponential in the input length (the sum of the sizes of all $\mathcal{G}_i$ and $\mathcal{H}_i$). Hence, we can guess in polynomial time the binary encodings of numbers $k_i \in \mathcal{O}(2^{3(\alpha n)^2+7\alpha n} \cdot \mu^{8\alpha(n+1)} \cdot \nu^{8\alpha|A|(n+1)})$ (where $k_i = k_j$ if $x_i = x_j$). Then, we have to verify whether

$$\mathrm{val}(\mathcal{H}_0)\mathrm{val}(\mathcal{G}_1)^{k_1}\mathrm{val}(\mathcal{H}_1)\mathrm{val}(\mathcal{G}_2)^{k_2}\mathrm{val}(\mathcal{H}_2)\cdots\mathrm{val}(\mathcal{G}_n)^{k_n}\mathrm{val}(\mathcal{H}_n) = 1$$

in the graph group $\mathbb{G}(A, I)$. This is an instance of the so-called *compressed word problem* for $\mathbb{G}(A, I)$, where the input consists of an SLP $\mathcal{G}$ over the alphabet $A^{\pm 1}$ and it is asked whether $\mathrm{val}(\mathcal{G}) = 1$ in $\mathbb{G}(A, I)$. Note that the powers $\mathrm{val}(\mathcal{G}_i)^{k_i}$ can be produced with the productions of $\mathcal{G}_i$ and additional $\lceil \log k_i \rceil$ many productions (using iterated squaring). Since the compressed word problem for a graph group can be solved in deterministic polynomial time [36,37] (NP would suffice), the theorem follows. For the last step, it is important that (A, I) is fixed. □

3.6 Solvability of compressed exponent equation for a variable graph group

For the proof of Theorem 3.1 we assumed that the graph group $\mathbb{G}(A, I)$ is fixed. In this section we briefly consider the case, where the independence alphabet (A, I) is part of the input as well. Let *uniform solvability of compressed exponent equations over graph groups* be the following computational problem.

Input: An independence alphabet (A, I) and a compressed exponent equation E over $\mathbb{G}(A, I)$.

Question: Is E solvable?

Note that the bound on the exponents x_i in the proof of Theorem 3.1 is still exponential in the input length if the independence alphabet (A, I) is part of the input as well. The problem is that we do not know whether the *uniform compressed word problem for graph groups* (where the input is an independence alphabet (A, I) together with an SLP over the terminal alphabet $A^{\pm 1}$) can be solved in polynomial time or at least in NP. The latter would suffice to get an NP-algorithm for uniform solvability of compressed exponent equations over a graph groups. On the other hand, we can show that the uniform compressed word problem for graph groups belongs to the complexity class coRP. A language $L \in \Sigma^*$ belongs to the class coRP if there exists a set $P \subseteq \Sigma^* \times \{0,1\}^*$ and a polynomial $p(n)$ such that P can be decided in deterministic polynomial time and for all $x \in \Sigma^*$ with $|x| = n$ the following holds:

- If $x \in L$ then $(x, y) \in P$ for all $y \in \{0,1\}^{p(n)}$.
- If $x \notin L$ then $(x, y) \in P$ for at most $1/3 \cdot 2^{p(n)}$ many $y \in \{0,1\}^{p(n)}$.

Theorem 3.12 *The uniform compressed word problem for graph groups belongs to* coRP.

Proof We make use of a well known embedding of a graph group $\mathbb{G}(A, I)$ with $n = |A|$ into the linear matrix group $\mathsf{GL}_{2n}(\mathbb{Z})$. This embedding is obtained by first embedding $\mathbb{G}(A, I)$ into a so called right-angled Coxeter group followed by a linear embedding for the latter group. A right-angled Coxeter group is obtained by adding to a graph group $\mathbb{G}(A, I)$ all relations $a^2 = 1$ for all generators $a \in A$. Let us denote with $\mathbb{C}(A, I)$ this right-angled Coxeter group.

The following embedding of a graph group $\mathbb{G}(A, I)$ into a right-angled Coxeter group goes back to [26]: Take a disjoint copy $A' = \{a' \mid a \in A\}$ of A and consider the right-angled Coxeter group $\mathbb{C}(A \cup A', J)$ with

$$J = \{(a, b), (a', b), (a, b'), (a', b') \mid (a, b) \in I\}.$$

Then the morphism $g \colon \mathbb{G}(A, I) \to \mathbb{C}(A \cup A', J)$ with $g(a) = aa'$ for $a \in A$ is injective.

Next, a right-angled Coxeter group $\mathbb{C}(A, I)$ with $|A| = n$ can be embedded into $\mathsf{GL}_n(\mathbb{Z})$ by mapping the generator $a \in A$ to the linear map $\sigma_a \colon \mathbb{Z}^A \to \mathbb{Z}^A$

defined by

$$\sigma_a(b) = \begin{cases} -b & \text{if } b = a, \\ b & \text{if } (a,b) \in I, \\ b + 2a & \text{if } a \neq b \text{ and } (a,b) \notin I \end{cases}$$

This is an instance of the standard linear embedding for general Coxeter groups, see e.g. the textbook [8] for more details.

Let us fix a graph group $\mathbb{G}(A, I)$ with $n = |A|$ and let $h\colon \mathbb{G}(A, I) \to \mathsf{GL}_{2n}(\mathbb{Z})$ be the linear embedding that results from the above construction. Note that for a given graph (A, I) we can compute in polynomial time for every generator $a \in A$ the corresponding matrix $h(a) \in \mathsf{GL}_{2n}(\mathbb{Z})$.

Let $\mathcal{G} = (V, \Sigma, \mathrm{rhs}, S)$ be an SLP over the terminal alphabet $A \cup A^{-1}$. Without loss of generality, one can assume that for every variable $X \in V$, the right-hand side $\mathrm{rhs}(X)$ belongs to $A \cup A^{-1} \cup VV$, see e.g. [36, Proposition 3.8]. We now construct an arithmetic circuit[5] that evaluates to 1 if and only if $\mathrm{val}(\mathcal{G}) = 1$ in the graph group $\mathbb{G}(A, I)$. The construction is the same as in [36, Theorem 4.15]. For every nonterminal $X \in V$ we introduce $4n^2$ many $+$-labelled gates $X_{i,j}$ $(1 \leq i, j \leq 2n)$. The idea is that $X_{i,j}$ evaluates to the entry at position (i, j) in the matrix $h(\mathrm{val}_{\mathcal{G}}(X))$. The wires between the gates are defined such that they implement matrix multiplication. If $\mathrm{rhs}(X) = YZ$, then we add an auxiliary $\times$-labelled gate $X_{i,j,k}$ together with the wires $(Y_{i,j}, X_{i,j,k})$, $(Z_{j,k}, X_{i,j,k})$, and $(X_{i,j,k}, X_{i,k})$ for all $1 \leq i, j, k \leq 2n$. If $\mathrm{rhs}(X) = a \in A \cup A^{-1}$, then we set the value of $X_{i,j}$ to the entry at position (i, j) of the matrix $h(a)$.

Assume that the gate $S_{i,j}$ of the constructed arithmetic circuit evaluates to the integer $s_{i,j}$. Then, the matrix $(s_{i,j})_{1 \leq i,j \leq n}$ is $h(\mathrm{val}(\mathcal{G}))$. Thus, we have to check, whether this matrix is the identity matrix. For this,we add an additional gate X (which will be the output gate of the circuit) together with some auxiliary gates to the circuit such that gate X evaluates to the integer $\sum_{i=1}^{n}(s_{i,i}-1)^2 + \sum_{i \neq j} s_{i,j}^2$. Then, $(s_{i,j})_{1 \leq i,j \leq n}$ is the identity matrix if and only if gate X evaluates to zero, We can conclude the proof by using the following well-known result, see e.g. [27]: Whether a given arithmetic circuit evaluates to zero can be decided in coRP. □

There is some evidence in complexity theory for $\mathsf{RP} = \mathsf{coRP} = \mathsf{P}$. Impagliazzo and Wigderson [28] proved that if there exists a language in $\mathsf{DTIME}(2^{\mathcal{O}(n)})$ that has circuit complexity $2^{\Omega(n)}$ (which seems to be plausible) then $\mathsf{RP} = \mathsf{coRP} = \mathsf{P}$ (in fact, $\mathsf{BPP} = \mathsf{P}$).

A language $L \in \Sigma^*$ belongs to the class MA (for Merlin-Arthur protocol) if there exists a set $P \subseteq \Sigma^* \times \{0,1\}^* \times \{0,1\}^*$ and polynomials $p(n), q(n)$ such that P can be decided in deterministic polynomial time and for all $x \in \Sigma^*$ with $|x| = n$ the following holds:

[5] An arithmetic circuit is a finite directed acyclic graph, where every node of indegree zero is labelled with a binary encoded integer, and every node of non-zero indegree is labelled with one of the arithmetic operations $+$ or $\times$. Nodes (resp., edges) of the arithmetic circuit are also called gates (resp., wires) and there is a distinguished gate, called the output gate. Every gate evaluates to an integer (the value of the gate) in the natural way, and the arithmetic circuits evaluates to the value of its output gate.

- If $x \in L$ then there exists $y \in \{0,1\}^{p(n)}$ such that $(x, y, z) \in P$ for all $z \in \{0,1\}^{q(n)}$.
- If $x \notin L$ then for all $y \in \{0,1\}^{p(n)}$ there exist at most $1/3 \cdot 2^{p(n)}$ many $z \in \{0,1\}^{q(n)}$ such that $(x, y, z) \in P$.

The same (unproven) circuit complexity lower bounds that allow to derandomize RP [28] also imply MA = NP.

Corollary 3.13 *Uniform solvability of compressed exponent equations over graph groups belongs to* MA.

Proof We follow to arguments from the proof of Theorem 3.1. As remarked above, the bound on the exponents x_i in the proof of Theorem 3.1 is still exponential in the input length if the independence alphabet (A, I) is part of the input as well. After guessing the values for the x_i in binary representation (this corresponds to the existential quantifier in the definition of MA), we are left with the solution of an instance of the uniform compressed word problem for graph groups, which belongs to coRP by Theorem 3.12. This yields an MA-protocol for uniform solvability of compressed exponent equations over graph groups. □

4 Uncompressed knapsack and subset sum

Since knapsack and subset sum for binary encoded integers is NP-complete, it follows that compressed knapsack and subset sum are NP-hard for every finitely generated group that contains an element of infinite order. In the rest of the paper, we will study the computational complexity of uncompressed knapsack and subset sum for graph groups. In the rest of the paper, the terms "knapsack" and "subset sum" will always refer to the uncompressed variant of the problem.

In Section 4.1, we present a class of graph groups for which knapsack is NP-complete. In Section 4.2, we will show that for all other graph groups, knapsack belongs to LogCFL, which is a subclass of P. In fact, we will show that knapsack is LogCFL-complete for these graph groups, unless they are abelian. Finally, in Section 4.4, we prove TC^0-completeness in the case of abelian graph groups (i.e. free abelian groups). For subset sum, we are not able to exactly locate the border between P-membership and NP-completeness.

4.1 NP-completeness

Figure 1 shows two important independence alphabets that we denote with P4 (path on four nodes) and C4 (cycle on four nodes). Note that $\mathbb{M}(\mathsf{C4}) = \{a, c\}^* \times \{b, d\}^*$ and $\mathbb{G}(\mathsf{C4}) \cong F_2 \times F_2$, where F_2 the free group of rank 2.

A *transitive forest* is an independence that can be inductively obtained as follows:

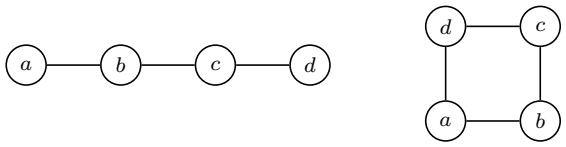

Fig. 1 P4 and C4

- $(\{a\}, \emptyset)$ is a transitive forest.
- If (A_1, I_1) and (A_2, I_2) are transitive forests, then also $(A_1 \cup A_2, I_1 \cup I_2)$ is a transitive forest.
- If (A, I) is a transitive forest and $a \notin I$, then $(A \cup \{a\}, I \cup \{a\} \times A \cup A \times \{a\})$ is a transitive forest.

The name "transitive forest" comes from the fact that these graphs are obtained by taking the transitive closure of a disjoint union of rooted directed trees (a forest) and then forgetting the direction of edges. From the above definition of transitive forests, it is clear that the graph groups $\mathbb{G}(A, I)$ with (A, I) a transitive forest is the smallest class of groups that contains $\mathbb{Z}$ and is closed under free products and direct products with $\mathbb{Z}$. We will use the following alternative characterization of transitive forests by Wolk [52]: (A, I) is a transitive forest if and only if (A, I) neither contains an induced P4 nor and induced C4.

In the rest of Section 4.1, we will prove the following theorem:

Theorem 4.1 *If (A, I) is not a transitive forest, then knapsack is* NP*-complete for $\mathbb{G}(A, I)$.*

By the above mentioned result of Wolk, it suffices to show that knapsack is NP-hard for $\mathbb{G}(\mathsf{C4})$ (Section 4.1.1) and $\mathbb{G}(\mathsf{P4})$ (Section 4.1.2). Note that if (A', I') is an induced subgraph of (A, I), then $\mathbb{G}(A', I')$ is a subgroup of $\mathbb{G}(A, I)$.

4.1.1 Knapsack and subset sum for $\mathbb{G}(\mathsf{C4})$

In this section, we prove that knapsack and subset sum are NP-complete for $\mathbb{G}(\mathsf{C4})$, i.e., for a direct product of two free groups of rank two. This solves an open problem from [18].

Recall that $F(\Sigma)$ denotes the free group generated by the set Σ and $F_2 = F(\{a, b\})$.

Lemma 4.2 *The subset sum problem and the knapsack problem are* NP*-complete for $F_2 \times F_2$. For knapsack,* NP*-hardness already holds for the variant where the exponent variables are allowed to take values from $\mathbb{Z}$ (see Remark 2.2).*

Proof In [44] it was shown that there exists a fixed set $D \subseteq F_2 \times F_2$ such that the following problem (called the bounded submonoid problem) is NP-complete:

Input: A unary encoded number n (i.e., n is given by the string a^n) and an element $g \in F_2 \times F_2$

Question: Do there exist $g_1, \dots g_n \in D$ (not necessarily distinct) such that $g = g_1 g_2 \cdots g_n$ in $F_2 \times F_2$?

Let us briefly explain the NP-hardness proof, since we will reuse it.

Recall the notion of the Dehn function defined in Section 2.5. We start with a finitely presented group $\langle \Sigma \mid R \rangle$ having an NP-complete word problem and a polynomial Dehn function. Such a group was constructed in [7]. To this group, the following classical construction by Mihaĭlova [42] is applied: Let

$$D = \{(r^\epsilon, 1) \mid r \in R, \epsilon \in \{-1, 1\}\} \cup \{(a, a) \mid a \in \Sigma^{\pm 1}\},$$

which is viewed as a subset of $F(\Sigma) \times F(\Sigma)$. Note that D is closed under taking inverses. Let $\langle D \rangle \leq F(\Sigma) \times F(\Sigma)$ be the subgroup generated by D. Mihaĭlova proved that for every word $w \in (\Sigma^{\pm 1})^*$ the following equivalence holds:

$$w = 1 \text{ in } \langle \Sigma, R \rangle \iff (w, 1) \in \langle D \rangle \text{ in } F(\Sigma) \times F(\Sigma).$$

Moreover, based on the fact that $\langle \Sigma, R \rangle$ has a polynomial Dehn function $p(n)$, the following equivalence was shown in [44], where $q(n) = p(n) + 8(c \cdot p(n) + n)$, c is the maximal length of a relator in R, and D^n is the set of all products of n elements from D:

$$w = 1 \text{ in } \langle \Sigma, R \rangle \iff \exists n \leq q(|w|) : (w, 1) \in D^n \text{ in } F(\Sigma) \times F(\Sigma).$$

From these two equivalences it follows directly that the following three statements are equivalent for all words $w \in (\Sigma^{\pm 1})^*$, where $D = \{g_1, g_2, \dots, g_k\}$:

- $w = 1$ in $\langle \Sigma, R \rangle$
- $(w, 1) = \prod_{i=1}^{q(|w|)} (g_1^{a_{1,i}} g_2^{a_{2,i}} \cdots g_k^{a_{k,i}})$ in $F(\Sigma) \times F(\Sigma)$ for $a_{j,i} \in \{0, 1\}$
- $(w, 1) = \prod_{i=1}^{q(|w|)} (g_1^{a_{1,i}} g_2^{a_{2,i}} \cdots g_k^{a_{k,i}})$ in $F(\Sigma) \times F(\Sigma)$ for $a_{j,i} \in \mathbb{Z}$

This shows that the subset sum problem and the knapsack problem are NP-hard for the group $F(\Sigma) \times F(\Sigma)$, where for knapsack we allow integer exponents. To get the same results for $F_2 \times F_2$, we use the fact that F_2 contains a copy of $F(\Sigma)$. □

4.1.2 Knapsack for $\mathbb{G}(\mathsf{P4})$

In this section, we show that knapsack is NP-complete for the graph group $\mathbb{G}(\mathsf{P4})$. Let us fix the copy $(\{a, b, c, d\}, I)$ of P4 shown in Figure 1.

As a first step, we will prove NP-completeness of a certain automata theoretic problem, that will be reduced to knapsack for $\mathbb{G}(\mathsf{P4})$ in a second step. For a trace monoid $\mathbb{M}(A, I)$, the *intersection nonemptiness problem for acyclic loop NFA* is the following computational problem:

Input: Two acyclic loop NFA $\mathcal{A}_1$, $\mathcal{A}_2$ over the input alphabet A (as defined in Section 2.2).

Question: Does $[L(\mathcal{A}_1)]_I \cap [L(\mathcal{A}_2)]_I \neq \emptyset$ hold?

Aalbersberg and Hoogeboom [1] proved that for the trace monoid $\mathbb{M}(\mathsf{P4})$ the intersection nonemptiness problem for arbitrary NFA is undecidable. We use their technique to show:

Lemma 4.3 *For $\mathbb{M}(\mathsf{P4})$, intersection nonemptiness for acyclic loop NFA is* NP*-hard.*

Proof We give a reduction from 3SAT. Let $\varphi = \bigwedge_{i=1}^{m} C_i$ where for every $i \in [1,m]$, $C_i = (L_{i,1} \vee L_{i,2} \vee L_{i,3})$ is a clause consisting of three literals. Let $x_1, \ldots, x_n$ be the boolean variables that occur in φ. In particular, every literal $L_{i,j}$ belongs to the set $\{x_1, \ldots, x_n, \neg x_1, \ldots, \neg x_n\}$.

Let $p_1, p_2, \ldots, p_n$ be a list of the first n prime numbers. So, for each boolean variable x_i we have the corresponding prime number p_i. We encode a valuation $\beta\colon \{x_1, \ldots, x_n\} \to \{0,1\}$ by any natural number N such that $N \equiv 0 \bmod p_i$ if and only if $\beta(x_i) = 1$. For a positive literal x_i let $S(x_i) = \{p_i \cdot n \mid n \in \mathbb{N}\}$ and for a negative literal $\neg x_i$ let $S(\neg x_i) = \{p_i \cdot n + r \mid n \in \mathbb{N}, r \in [1, p_i - 1]\}$. Moreover, for every $i \in [1,m]$ let $S_i = S(L_{i,1}) \cup S(L_{i,2}) \cup S(L_{i,3})$. Thus, S_i is the set of all numbers that encode a valuation, which makes the clause C_i true. Hence, the set $S = \bigcap_{i=1}^{n} S_i$ encodes the set of all valuations that make φ true.

We first construct an acyclic loop NFA $\mathcal{A}_1$ with

$$L(\mathcal{A}_1) = \prod_{i=1}^{m} \{a(bc)^{N_i} d \mid N_i \in S_i\}.$$

Note that φ is satisfiable iff $[L(\mathcal{A}_1)]_I$ contains a trace from $[\{(a(bc)^N d)^m \mid N \in \mathbb{N}\}]_I$. We will ensure this property with a second acyclic loop NFA $\mathcal{A}_2$ that satisfies the equality $L(\mathcal{A}_2) = b^*(ad(bc)^*)^{m-1}adc^*$.

We claim that $[L(\mathcal{A}_1)]_I \cap [L(\mathcal{A}_2)]_I = [\{(a(bc)^N d)^m \mid N \in S\}]_I$. First assume that $w \equiv_I (a(bc)^N d)^m$ for some $N \in S$. We have

$$w \equiv_I (a(bc)^N d)^m \equiv_I b^N (ad(bc)^N)^{m-1} adc^N$$

and thus $[w]_I \in [L(\mathcal{A}_2)]_I$. Moreover, since $N \in S$ we get $[w]_I \in [L(\mathcal{A}_1)]_I$. For the other direction, let $[w]_I \in [L(\mathcal{A}_1)]_I \cap [L(\mathcal{A}_2)]_I$. Thus

$$w \equiv_I \prod_{i=1}^{m} (a(bc)^{N_i} d) \equiv b^{N_1} \Bigg(\prod_{i=1}^{m-1} (adc^{N_i} b^{N_{i+1}}) \Bigg) adc^{N_m},$$

where $N_i \in S_i$ for $i \in [1,m]$. Moreover, the fact that $[w]_I \in [L(\mathcal{A}_2)]_I$ means hat there are $k_0, k_1, \ldots, k_{m-1}, k_m \geq 0$ with

$$\begin{aligned} b^{N_1} \Bigg(\prod_{i=1}^{m-1} (adc^{N_i} b^{N_{i+1}}) \Bigg) adc^{N_m} &\equiv_I b^{k_0} \Bigg(\prod_{i=1}^{m-1} (ad(bc)^{k_i}) \Bigg) adc^{k_m} \\ &\equiv_I b^{k_0} \Bigg(\prod_{i=1}^{m-1} (adb^{k_i} c^{k_i}) \Bigg) adc^{k_m}. \end{aligned}$$

Since every symbol is dependent from a or d, this identity implies $N_i = N_{i+1}$ for $i \in [1, m-1]$. Thus, $[w]_I \in [\{(a(bc)^N d)^m \mid N \in S\}]_I$. □

For a graph group $\mathbb{G}(A, I)$ the *membership problem for acyclic loop NFA* is the following computational problem:

Input: An acyclic loop NFA $\mathcal{A}$ over the input alphabet $A \cup A^{-1}$.

Question: Is there a word $w \in L(\mathcal{A})$ such that $w = 1$ in $\mathbb{G}(A, I)$?

It is straightforward to reduce the intersection nonemptiness problem for acyclic loop NFA over $\mathbb{M}(A, I)$ to the membership problem for acyclic loop NFA over $\mathbb{G}(A, I)$. For the rest of this section let $\Sigma = \{a, b, c, d, a^{-1}, b^{-1}, c^{-1}, d^{-1}\}$ and let $\theta\colon \Sigma^* \to \mathbb{G}(P_4)$ be the canonical homomorphism that maps a word over Σ to the corresponding group element.

Lemma 4.4 *For $\mathbb{G}(\mathsf{P4})$, the membership problem for acyclic loop NFA is* NP*-hard.*

Proof The lemma follows easily from Lemma 4.3. Note that $[L(\mathcal{A}_1)]_I \cap [L(\mathcal{A}_2)]_I \neq \emptyset$ if and only if $1 \in \theta(L(\mathcal{A}_1)L(\mathcal{A}_2)^{-1})$ in the graph group $\mathbb{G}(\mathsf{P4})$. Moreover, it is straightforward to construct from acyclic loop NFA $\mathcal{A}_1$ and $\mathcal{A}_2$ an acyclic loop NFA for $L(\mathcal{A}_1)L(\mathcal{A}_2)^{-1}$. We only have to replace every transition label w in $\mathcal{A}_2$ by w^{-1}, then reverse all transitions in $\mathcal{A}_2$ and concatenate the resulting NFA with $\mathcal{A}_1$ on the left. □

We can now use a construction from [38] to reduce membership for acyclic loop NFA to knapsack.

Lemma 4.5 *Knapsack for the graph group $\mathbb{G}(\mathsf{P4})$ is* NP*-hard.*

Proof By Lemma 4.4 it suffices to reduce for $\mathbb{G}(\mathsf{P4})$ the membership problem for acyclic loop NFA to knapsack. Let $\mathcal{A} = (Q, \Sigma, \Delta, q_0, q_f)$ be an acyclic loop NFA with transitions $\Delta \subseteq Q \times \Sigma^* \times Q$. W.l.o.g. assume that $Q = \{1, \ldots, n\}$.

We reuse a construction from [38], where the rational subset membership problem for $\mathbb{G}(\mathsf{P4})$ was reduced to the submonoid membership problem for $\mathbb{G}(\mathsf{P4})$. For a state $q \in Q$ let $\widetilde{q} = (ada)^q d (ada)^{-q} \in \Sigma^*$. Let us fix the morphism $\varphi\colon \Sigma^* \to \Sigma^*$ with $\varphi(x) = xx$ for $x \in \Sigma$. For a transition $t = (p, w, q) \in \Delta$ let $\tilde{t} = \widetilde{p}\,\varphi(w)\,\widetilde{q}^{-1}$ and define $S = \{\tilde{t} \mid t \in \Delta\}^*$. In [38] it was shown that $1 \in \theta(L(\mathcal{A}))$ if and only if $\theta(\widetilde{q_0}\,\widetilde{q_f}^{-1}) \in \theta(S)$.

We construct in polynomial time a knapsack instance over $\mathbb{G}(P_4)$ from the NFA $\mathcal{A}$ as follows: Let us choose an enumeration $t_1, t_2, \ldots, t_m$ of the transitions of $\mathcal{A}$ such that the following holds, where $t_i = (p_i, w_i, q_i)$: If $q_j = p_k$ then $j \leq k$. Since $\mathcal{A}$ is an acyclic loop NFA, such an enumeration exists. The following claim proves the theorem.

Claim: $1 \in \theta(L(\mathcal{A}))$ if and only if $\theta(\widetilde{q_0}\,\widetilde{q_f}^{-1}) \in \theta(\tilde{t}_1^*\,\tilde{t}_2^* \cdots \tilde{t}_m^*)$.

One direction is clear: If $\theta(\widetilde{q_0}\,\widetilde{q_f}^{-1}) \in \theta(\tilde{t}_1^*\,\tilde{t}_2^* \cdots \tilde{t}_m^*)$, then $\theta(\widetilde{q_0}\,\widetilde{q_f}^{-1}) \in \theta(S)$. Hence, by [38] we have $1 \in \theta(L(\mathcal{A}))$. On the other hand, if $1 \in \theta(L(\mathcal{A}))$, then there exists a path in $\mathcal{A}$ of the form

$$q_0 = s_0 \xrightarrow{a_1} s_1 \xrightarrow{a_2} s_2 \cdots s_{k-1} \xrightarrow{a_k} s_k = q_f$$

such that $\theta(a_1 a_2 \cdots a_k) = 1$. Let $(s_{j-1}, a_j, s_j) = t_{i_j}$, where we refer to the above enumeration of all transitions. Then, we must have $i_1 \leq i_2 \leq \cdots \leq i_k$. Moreover, we have

$$\theta(\widetilde{q_0}\,\widetilde{q_f}^{-1}) = \theta(\widetilde{q_0}\,a_1 a_2 \cdots a_k\,\widetilde{q_f}^{-1}) = \theta(\tilde{t}_{i_1}\tilde{t}_{i_2}\cdots\tilde{t}_{i_k}) \in \theta(\tilde{t}_1^*\,\tilde{t}_2^*\cdots\tilde{t}_m^*).$$

This proves the claim and hence the theorem. □

We are now ready to prove Theorem 4.1.

Proof (Theorem 4.1) If (A, I) is not a transitive forest, then P4 or C4 is an induced subgraph of (A, I) [52]. Thus, $\mathbb{G}(\mathsf{P4})$ or $\mathbb{G}(\mathsf{C4}) \cong F_2 \times F_2$ is a subgroup of $\mathbb{G}(A, I)$. Hence, NP-hardness of knapsack for $\mathbb{G}(A, I)$ follows from Lemma 4.2 or Lemma 4.5. □

4.2 Membership in LogCFL

In this section, we show that if (A, I) is a transitive forest, then knapsack and subset sum belong to LogCFL, which is a subclass of P; see Section 2.1.

Theorem 4.6 *If (A, I) is a transitive forest, then knapsack and subset sum for $\mathbb{G}(A, I)$ belong to* LogCFL.

4.2.1 Membership for acyclic NFA

In the proof of Theorem 4.6 we employ the membership problem for acyclic NFA (see Section 2.2), which has already been studied in connection with the knapsack and subset sum problem [18,33]. For a graph group $\mathbb{G}(A, I)$ the *membership problem for acyclic NFA* is the following computational problem:

Input: An acyclic automaton $\mathcal{A}$ over the input alphabet $A \cup A^{-1}$.

Question: Is there a word $w \in L(\mathcal{A})$ such that $w = 1$ in $\mathbb{G}(A, I)$?

In order to show Theorem 4.6, we reduce knapsack for $\mathbb{G}(A, I)$ with (A, I) a transitive forest to the membership problem for acyclic NFA for $\mathbb{G}(A, I)$ (note that for subset sum this reduction is obvious). Then, we apply the following Proposition 4.7. From work of Frenkel, Nikolaev, and Ushakov [18], it follows that the membership problem for acyclic NFA is in P. We strengthen this to LogCFL:

Proposition 4.7 *If (A, I) is a transitive forest, then the membership problem for acyclic NFA over $\mathbb{G}(A, I)$ is in* LogCFL.

The proof of Proposition 4.7 uses the following lemma:

Lemma 4.8 *For every transitive forest (A, I) with the associated graph group $G = \mathbb{G}(A, I)$ there is a deterministic AuxPDA $\mathcal{P}(G)$ with input alphabet $A^{\pm 1}$ and the following properties:*

- *In each step, the input head for $\mathcal{P}(G)$ either does not move, or moves one step to the right.*
- *If the input word is equal to 1 in G, then $\mathcal{P}(G)$ terminates in the distinguished state q_1 with empty stack. Let us call this state the 1-state of $\mathcal{P}(G)$.*
- *If the input word is not equal to 1 in G, then $\mathcal{P}(G)$ terminates in a state different from q_1 (and the stack is not necessarily empty).*

Proof We construct the AuxPDA $\mathcal{P}(G)$ by induction over the structure of the group G. For this, we consider the three cases that $G = 1$, $G = G_1 * G_2$, and $G = \mathbb{Z} \times G'$. The case that $G = 1$ is of course trivial.

Case $G = \mathbb{Z} \times G'$. We have already constructed the AuxPDA $\mathcal{P}(G')$. The AuxPDA $\mathcal{P}(G)$ simulates the AuxPDA $\mathcal{P}(G')$ on the generators of G'. Moreover, it stores the current value of the $\mathbb{Z}$-component in binary notation on the work tape. If the input word has length n, then $O(\log n)$ bits are sufficient for this. At the end, $\mathcal{P}(G)$ goes into its 1-state if and only if $\mathcal{P}(G')$ is in its 1-state (which implies that the stack will be empty) and the $\mathbb{Z}$-component is zero.

Case $G = G_1 * G_2$. For $i \in \{1, 2\}$, we have already constructed the AuxPDA $\mathcal{P}_i = \mathcal{P}(G_i)$. Let $A_i^{\pm 1}$ be its input alphabet, which is a monoid generating set for G_i. Consider now an input word $w \in (A_1^{\pm 1} \cup A_2^{\pm 1})^*$. Let us assume that $w = u_1 v_1 u_2 v_2 \cdots u_k v_k$ with $u_i \in (A_1^{\pm 1})^+$ and $v_i \in (A_2^{\pm 1})^+$ (other cases can be treated analogously). The AuxPDA $\mathcal{P}(G)$ starts with empty stack and simulates the AuxPDA $\mathcal{P}_1$ on the prefix u_1. If it turns out that $u_1 = 1$ in G_1 (which means that $\mathcal{P}_1$ is in its 1-state) then the stack will be empty and the AuxPDA $\mathcal{P}(G)$ continues with simulating $\mathcal{P}_2$ on v_1. On the other hand, if $u_1 \neq 1$ in G_1, then $\mathcal{P}(G)$ pushes the state together with the work tape content of $\mathcal{P}_1$ reached after reading u_1 on the stack (on top of the final stack content of $\mathcal{P}_1$). This allows $\mathcal{P}(G)$ to resume the computation of $\mathcal{P}_1$ later. Then $\mathcal{P}(G)$ continues with simulating $\mathcal{P}_2$ on v_1.

The computation of $\mathcal{P}(G)$ will continue in this way. More precisely, if after reading u_i (resp. v_i with $i < k$) the AuxPDA $\mathcal{P}_1$ (resp. $\mathcal{P}_2$) is in its 1-state then either

(i) the stack is empty or
(ii) the top part of the stack is of the form sqt (t is the top), where s is a stack content of $\mathcal{P}_2$ (resp. $\mathcal{P}_1$), q is a state of $\mathcal{P}_2$ (resp. $\mathcal{P}_1$) and t is a work tape content of $\mathcal{P}_2$ (resp. $\mathcal{P}_1$).

In case (i), $\mathcal{P}(G)$ continues with the simulation of $\mathcal{P}_2$ (resp. $\mathcal{P}_1$) on the word v_i (resp. u_{i+1}) in the initial configuration. In case (ii), $\mathcal{P}(G)$ continues with the simulation of $\mathcal{P}_2$ (resp. $\mathcal{P}_1$) on the word v_i (resp. u_{i+1}), where the simulation is started with stack content s, state q, and work tape content t. On the other hand, if after reading u_i (resp. v_i with $i < k$) the AuxPDA $\mathcal{P}_1$ (resp. $\mathcal{P}_2$) is not in its 1-state then $\mathcal{P}(G)$ pushes on the stack the state and work tape content of $\mathcal{P}_1$ reached after its simulation on u_i. This concludes the description of the AuxPDA $\mathcal{P}(G)$. It is clear that $\mathcal{P}(G)$ has the properties stated in the lemma. □

We can now prove Proposition 4.7:

Proof of Proposition 4.7. Fix the graph group $G = \mathbb{G}(A, I)$, where (A, I) is a transitive forest. An AuxPDA for the membership problem for acyclic NFA guesses a path in the input NFA $\mathcal{A}$ and thereby simulates the AuxPDA $\mathcal{P}(G)$ from Lemma 4.8. If the final state of the input NFA $\mathcal{A}$ is reached while the AuxPDA $\mathcal{P}(G)$ is in the accepting state q_1, then the overall AuxPDA accepts. It is important that the AuxPDA $\mathcal{P}(G)$ works one-way since the guessed path in $\mathcal{A}$ cannot be stored in logspace. This implies that the AuxPDA cannot re-access the input symbols that already have been processed. Also note that the AuxPDA is logspace bounded and polynomially time bounded since $\mathcal{A}$ is acyclic. □

4.2.2 Bounds on knapsack solutions in transitive forests

As mentioned above, we reduce for graph groups $\mathbb{G}(A, I)$ with (A, I) a transitive forest the knapsack problem to the membership problem for acyclic NFA. To this end, we show that every positive knapsack instance has a polynomially bounded solution. The latter is the most involved proof in our paper.

Frenkel, Nikolaev, and Ushakov [18] call groups with this property *polynomially bounded knapsack groups* and show that this class is closed under taking free products. However, it is not clear if direct products with $\mathbb{Z}$ also inherit this property and we leave this question open.

Hence, we are looking for a property that yields polynomial-size solutions and is passed on to free products and to direct products with $\mathbb{Z}$. It is known that the solution sets are always semilinear. If (A, I) is a transitive forest, this follows from a more general semilinearity property of rational sets [38] and for arbitrary graph groups, this was shown in Theorem 3.11.

Note that it is not true that the solution sets always have polynomial-size semilinear representations. This already fails in the case of $\mathbb{Z}$: The equation $x_1 + \cdots + x_k = k$ has $\binom{2k-1}{k} \geq 2^k$ solutions. We therefore need a weaker property: We will show here that the solution sets have semilinear representations where every occurring number is bounded by a polynomial.

For a semilinear representation $(x_1, F_1, \ldots, x_n, F_n)$ of the semilinear set $S = \bigcup_{i=1}^{n} x_i + F_i^{\oplus}$, the *magnitude* of this representation is defined as the maximum of $\|y\|_\infty$, where y ranges over all vectors of $\bigcup_{i=1}^{n} \{x_i\} \cup F_i$. The *magnitude* of a semilinear set S is the smallest magnitude of a semilinear representation for S.

Definition 4.9 *A group G is called* knapsack tame *if there is a polynomial p such that for every exponent equation $h_0 g_1^{x_1} h_1 g_2^{x_2} h_2 \cdots g_n^{x_k} h_k = 1$ of size n with pairwise distinct variables $x_1, \ldots, x_k$, the set $S \subseteq \mathbb{N}^k$ of solutions is semilinear of magnitude at most $p(n)$.*

Note that here, we only consider exponent equations where each variable occurs at most once. This corresponds to the definition of the knapsack problem as introduced by Myasnikov et. al. [44]. This is in contrast to section 3,

where the methods we used to obtain NP membership work for general exponent equations. In the case of the LogCFL membership proof, however, our techniques only apply to the original version of the knapsack problem. See also Remark 2.1.

Observe that although the size of an exponent equation may depend on the chosen generating set of G, changing the generating set increases the size only by a constant factor. Thus, whether or not a group is knapsack tame is independent of the chosen generating set.

Theorem 4.10 *If (A, I) is a transitive forest, then $\mathbb{G}(A, I)$ is knapsack tame.*

Note that Theorem 4.10 implies in particular that every solvable exponent equation with pairwise distinct variables has a polynomially bounded solution. Theorem 4.10 and Proposition 4.7 easily imply Theorem 4.6.

We prove Theorem 4.10 by showing that knapsack tameness transfers from groups G to $G \times \mathbb{Z}$ (Proposition 4.11) and from G and H to $G * H$ (Proposition 4.17). Since the trivial group is obviously knapsack tame, the inductive characterization of groups $\mathbb{G}(A, I)$ for transitive forests (A, I) immediately yields Theorem 4.10.

4.2.3 Tameness of direct products with $\mathbb{Z}$

In this section, we show the following.

Proposition 4.11 *If G is knapsack tame, then so is $G \times \mathbb{Z}$.*

Linear Diophantine equations. We employ a result of Pottier [47], which bounds the norm of minimal non-negative solutions to a linear Diophantine equation. Recall the definition of the vector norms $\|x\|_\infty$ and $\|x\|_1$ from Section 2.3. Let $A \in \mathbb{Z}^{k \times m}$ be an integer matrix where a_{ij} is the entry of A at row i and column j. We will use the following matrix norms:

$$\|A\|_{1,\infty} = \max_{i \in [1,k]} \Big(\sum_{j \in [1,m]} |a_{ij}| \Big),$$
$$\|A\|_{\infty,1} = \max_{j \in [1,m]} \Big(\sum_{i \in [1,k]} |a_{ij}| \Big),$$
$$\|A\|_{\infty} = \max_{i \in [1,k], j \in [1,m]} |a_{ij}|.$$

A non-trivial solution $x \in \mathbb{N}^m \setminus \{0\}$ to the equation $Ax = 0$ is *minimal* if there is no $y \in \mathbb{N}^m \setminus \{0\}$ with $Ay = 0$ and $y \leq x$, $y \neq x$. Here $y \leq x$ means that $y_i \leq x_i$ for all $i \in [1, m]$. The set of all solutions clearly forms a submonoid of $\mathbb{N}^m$. Let r be the rank of A.

Theorem 4.12 (Pottier [47]) *Each non-trivial minimal solution $x \in \mathbb{N}^m$ to $Ax = 0$ satisfies $\|x\|_1 \leq (1 + \|A\|_{1,\infty})^r$.*

We only need Theorem 4.12 for the case that A is a row vector u^T for $u \in \mathbb{Z}^k$.

Corollary 4.13 *Let $u \in \mathbb{Z}^k$. Each non-trivial minimal solution $x \in \mathbb{N}^k$ to $u^T x = 0$ satisfies $\|x\|_1 \leq 1 + \|u\|_1$.*

By applying Theorem 4.12 to the row vector $(u^T, -b)$ for $b \in \mathbb{Z}$, it is easy to deduce that for each $x \in \mathbb{N}^k$ with $u^T x = b$, there is a $y \in \mathbb{N}^k$ with $u^T y = b$, $y \leq x$, and $\|y\|_1 \leq 1 + \|\binom{u}{b}\|_1 = 1 + \|u\|_1 + |b|$. We reformulate Corollary 4.13 as follows.

Lemma 4.14 *Let $u \in \mathbb{Z}^k$ and $b \in \mathbb{Z}$. Then the set $\{x \in \mathbb{N}^k \mid u^T x = b\}$ admits a decomposition $\{x \in \mathbb{N}^k \mid u^T x = b\} = \bigcup_{i=1}^{s} c_i + C\mathbb{N}^t$, where $c_i \in \mathbb{N}^k$ and $C \in \mathbb{N}^{k \times t}$ with $\|c_i\|_1$ and $\|C\|_{\infty,1}$ bounded by $1 + \|u\|_1 + |b|$.*

Proof Let $\{c_1, \ldots, c_s\}$ be the set of minimal solutions of $u^T x = b$. Then, as explained above, Corollary 4.13 yields $\|c_i\|_1 \leq 1 + \|u\|_1 + |b|$. Moreover, let $C \in \mathbb{N}^{k \times t}$ be the matrix whose columns are the non-trivial minimal solutions of $u^T x = 0$. Then we have $\|C\|_{\infty,1} \leq 1 + \|u\|_1$. This clearly yields the desired decomposition. □

The problem is that we want to apply Lemma 4.14 in a situation where we have no bound on $\|u\|_1$, but only one on $\|u\|_\infty$. The following lemma yields such a bound.

Lemma 4.15 *If $u \in \mathbb{Z}^k$ and $b \in \mathbb{Z}$ with $\|u\|_\infty, |b| \leq M$, then we have a decomposition $\{x \in \mathbb{N}^k \mid u^T x = b\} = \bigcup_{i=1}^{s} c_i + C\mathbb{N}^t$ where $\|c_i\|_1, \|C\|_{\infty,1} \leq 1 + (M+2)M$.*

Proof Write $u^T = (b_1, \ldots, b_k)$ and consider the row vector $v^T = (b'_1, \ldots, b'_{2M+1})$, $v \in \mathbb{Z}^{2M+1}$, with entries $b'_i = i - (M+1)$. Thus, we have

$$v^T = (b'_1, \ldots, b'_{2M+1}) = (-M, -M+1, \ldots, -1, 0, 1, \ldots, M).$$

Moreover, define the matrix $S = (s_{ij}) \in \mathbb{N}^{(2M+1) \times k}$ with

$$s_{ij} = \begin{cases} 1 & \text{if } b_j = b'_i, \\ 0 & \text{otherwise.} \end{cases}$$

Then clearly $u = v^T S$ and $\|v\|_1 = (M+1)M$. Furthermore, observe that we have $\|Sx\|_1 = \|x\|_1$ for every $x \in \mathbb{N}^k$ and that for each $y \in \mathbb{N}^{2M+1}$ the set

$$T_y = \{x \in \mathbb{N}^k \mid Sx = y\}$$

is finite.

According to Lemma 4.14, we can write

$$\{x \in \mathbb{N}^{2M+1} \mid v^T x = b\} = \bigcup_{i=1}^{s'} c'_i + C'\mathbb{N}^{t'} \tag{14}$$

where $\|c_i'\|_1, \|C'\|_{\infty,1} \leq 1 + (M+1)M + M = 1 + (M+2)M$. Let $\{c_1, \ldots, c_s\}$ be the union of all sets $T_{c_i'}$ for $i \in [1, s']$ and let $C \in \mathbb{N}^{k \times t}$ be the matrix whose columns comprise all T_v where $v \in \mathbb{N}^{2M+1}$ is a column of C'. Since we have $\|Sx\|_1 = \|x\|_1$ for $x \in \mathbb{N}^k$, the vectors c_i obey the same bound as the vectors c_i', meaning $\|c_i\|_1 \leq 1 + (M+2)M$. By the same argument, we have $\|C\|_{\infty,1} \leq \|C'\|_{\infty,1} \leq 1 + (M+2)M$. It remains to be shown that the equality from Lemma 4.15 holds.

Suppose $u^T x = b$. Then $v^T S x = b$ and hence $Sx = c_i' + C'y$ for some $y \in \mathbb{N}^{t'}$. Observe that every column of S is either zero or a unit vector. This implies that if $Sz = p + q$ for $p, q \in \mathbb{N}^{2M+1}$, then z decomposes as $z = p' + q'$, $p', q' \in \mathbb{N}^k$, so that $Sp' = p$ and $Sq' = q$. Therefore, we can write $x = x_0 + \cdots + x_n$ with $Sx_0 = c_i'$ and Sx_j is some column of C' for each $j \in [1, n]$. Hence, $x_0 = c_r$ for some $r \in [1, s]$ and for each $j \in [1, n]$, x_j is a column of C. This proves $x \in c_r + C\mathbb{N}^t$.

On the other hand, the definition of $c_1, \ldots, c_s$ and C implies that for each column v of C, Sv is a column of C'. Moreover, for each $i \in [1, s]$, there is a $j \in [1, s']$ with $Sc_i = c_j'$ and thus $Sc_i + SC\mathbb{N}^t \subseteq c_j' + C'\mathbb{N}^{t'}$. Therefore

$$u^T(c_i + C\mathbb{N}^t) = v^T S(c_i + C\mathbb{N}^t) \subseteq v^T(c_j' + C'\mathbb{N}^{t'})$$

and the latter set contains only b because of (14). □

Lemma 4.16 *Let $S \subseteq \mathbb{N}^k$ be a semilinear set of magnitude M and $u \in \mathbb{Z}^k$, $b \in \mathbb{Z}$ with $\|u\|_\infty, |b| \leq m$. Then $\{x \in S \mid u^T x = b\}$ is a semilinear set of magnitude at most $10(kmM)^3$.*

Proof Let $T = \{x \in \mathbb{N}^k \mid u^T x = b\}$. We may assume that S is linear of magnitude M, because if $S = L_1 \cup \cdots \cup L_n$, then $S \cap T = (L_1 \cap T) \cup \cdots \cup (L_n \cap T)$.

Write $S = a + A\mathbb{N}^n$ with $a \in \mathbb{N}^k$ and $A \in \mathbb{N}^{k \times n}$, where $\|a\|_\infty \leq M$ and $\|A\|_\infty \leq M$. Consider the set $U = \{x \in \mathbb{N}^n \mid u^T A x = b - u^T a\}$. Note that $u^T A \in \mathbb{Z}^{1 \times n}$ and

$$\|u^T A\|_\infty \leq k \cdot \|u\|_\infty \cdot \|A\|_\infty \leq kmM,$$

$$|b - u^T a| \leq m + k \cdot \|u\|_\infty \cdot \|a\|_\infty \leq m + kmM.$$

According to Lemma 4.15, we can write $U = \bigcup_{i=1}^s c_i + C\mathbb{N}^t$ where $\|c_i\|_1$ and $\|C\|_{\infty,1}$ are at most $1 + (m + kmM)(m + kmM + 2) \leq 9(kmM)^2$. Observe that

$$a + AU = \bigcup_{i=1}^{s} a + Ac_i + AC\mathbb{N}^t$$

and

$$\|a + Ac_i\|_\infty \leq \|a\|_\infty + \|A\|_\infty \cdot \|c_i\|_1 \leq M + M \cdot 9(kmM)^2 \leq 10(kmM)^3,$$

$$\|AC\|_\infty \leq \|A\|_\infty \cdot \|C\|_{\infty,1} \leq M \cdot 9(kmM)^2 \leq 10(kmM)^3.$$

Finally, note that $S \cap T = a + AU$. □

We are now ready to prove Proposition 4.11.

Proof of Proposition 4.11. Suppose G is knapsack tame with polynomial $\bar{p}$. Let

$$h_0 g_1^{x_1} h_1 g_2^{x_2} h_2 \cdots g_k^{x_k} h_k = 1 \tag{15}$$

be an exponent equation of size n with pairwise distinct variables $x_1, \ldots, x_k$ and with $h_0, g_1, h_1, \ldots, g_k, h_k \in G \times \mathbb{Z}$. Let $h_i = (\bar{h}_i, y_i)$ for $i \in [0,k]$ and $g_i = (\bar{g}_i, z_i)$ for $i \in [1,k]$.

The exponent equation $\bar{h}_0 \bar{g}_1^{x_1} \bar{h}_1 \bar{g}_2^{x_2} \bar{h}_2 \cdots \bar{g}_k^{x_k} \bar{h}_k = 1$ has a semilinear solution set $\bar{S} \subseteq \mathbb{N}^k$ of magnitude at most $\bar{p}(n)$. The solution set of (15) is

$$S = \{(x_1, \ldots, x_k) \in \bar{S} \mid z_1 x_1 + \cdots + z_k x_k = y\},$$

where $y = -(y_0 + \cdots + y_k)$. Note that $|z_i| \leq n$ and $|y| \leq n$. By Lemma 4.16, S is semilinear of magnitude $10(n^2 \bar{p}(n))^3$ (recall that $k \leq n$). □

4.2.4 Tameness of free products

This section is devoted to the proof of the following proposition.

Proposition 4.17 *If G_0 and G_1 are knapsack tame, then so is $G_0 * G_1$.*

Let $G = G_0 * G_1$. Suppose that for $i \in \{0,1\}$, the group G_i is generated by A_i, where w.l.o.g. $A_i^{-1} = A_i$ and let $A = A_0 \uplus A_1$, which generates G. Recall that every $g \in G$ can be written uniquely as $g = g_1 \cdots g_n$ where $n \geq 0$, $g_i \in (G_0 \setminus \{1\}) \cup (G_1 \setminus \{1\})$ for each $i \in [1,n]$ and where $g_j \in G_t$ iff $g_{j+1} \in G_{1-t}$ for $j \in [1, n-1]$. We call g *cyclically reduced* if for some $t \in \{0,1\}$, either $g_1 \in G_t$ and $g_n \in G_{1-t}$ or $g_1, g_n \in G_t$ and $g_n g_1 \neq 1$. Consider an exponent equation

$$h_0 g_1^{x_1} h_1 \cdots g_k^{x_k} h_k = 1, \tag{16}$$

of size n, where g_i is represented by $u_i \in A^*$ for $i \in [1,k]$ and h_i is represented by $v_i \in A^*$ for $i \in [0,k]$. Then clearly $\sum_{i=0}^{k} |v_i| + \sum_{i=1}^{k} |u_i| \leq n$. Let $S \subseteq \mathbb{N}^k$ be the set of all solutions to (16). Every word $w \in A^*$ has a (possibly empty) unique factorization into maximal factors from $A_0^+ \cup A_1^+$, which we call *syllables*. By $\|w\|$, we denote the number of syllables of w. The word w is *reduced* if none of its syllables represents 1 (in G_0 resp. G_1). We define the maps $\lambda, \rho \colon A^+ \to A^+$ ("rotate left/right"), where for each word $w \in A^+$ with its factorization $w = w_1 \cdots w_m$ into syllables, we set $\lambda(w) = w_2 \cdots w_m w_1$ and $\rho(w) = w_m w_1 w_2 \cdots w_{m-1}$.

Consider a word $w \in A^*$ and suppose $w = w_1 \cdots w_m$, $m \geq 0$, where for each $i \in [1,m]$, we have $w_i \in A_j^+$ for some $j \in \{0,1\}$ (we allow that $w_i, w_{i+1} \in A_j^+$). A *cancellation* is a subset $C \subseteq 2^{[1,m]}$ that is

- *a partition*: $\bigcup_{I \in C} I = [1,m]$ and $I \cap J = \emptyset$ for any $I, J \in C$ with $I \neq J$.
- *consistent*: for each $I \in C$, there is an $i \in \{0,1\}$ such that $w_j \in A_i^+$ for all $j \in I$.

- *cancelling*: if $\{i_1, \dots, i_\ell\} \in C$ with $i_1 < \cdots < i_\ell$, then $w_{i_1} \cdots w_{i_\ell}$ represents 1 in G.
- *well-nested*: there are no $I, J \in C$ with $i_1, i_2 \in I$ and $j_1, j_2 \in J$ such that $i_1 < j_1 < i_2 < j_2$.
- *maximal*: if $w_i, w_{i+1} \in A_j^+$ for $j \in \{0,1\}$ then there is an $I \in C$ with $i, i+1 \in I$.

Since C can be regarded as a hypergraph on $[1,m]$, the elements of C will be called *edges*. We have the following simple fact:

Lemma 4.18 *Let $w = w_1 \cdots w_m$, $m \geq 0$, where for each $i \in [1,m]$, we have $w_i \in A_j^+$ for some $j \in \{0,1\}$. Then w admits a cancellation if and only if it represents 1 in G.*

Proof Assume that w represents 1 in the free product G. The case $w = \varepsilon$ is clear; hence assume that $w \neq \varepsilon$. Then there must exist a factor $w_i w_{i+1} \cdots w_j$ representing 1 in G such that (i) $w_i w_{i+1} \cdots w_j \in A_k^+$ for some $k \in \{0,1\}$, (ii) either $i = 1$ or $w_{i-1} \in A_{1-k}^+$, and (iii) either $j = m$ or $w_{j+1} \in A_{1-k}^+$. The word $w' = w_1 \cdots w_{i-1} w_{j+1} \cdots w_m$ also represents 1 in G. By induction, w' admits a cancellation C'. Let C'' be obtained from C' by replacing every occurrence of an index $k \geq i$ in C' by $k + j - i + 1$. Then $C = C'' \cup \{[i,j]\}$ is a cancellation for w.

For the other direction let us call a partition $C \subseteq 2^{[1,m]}$ a *weak cancellation* if it is consistent, cancelling and well-nested (but not necessarily maximal). Then we show by induction that w represents 1, if it has a weak cancellation. So, let C be a weak cancellation of $w \neq \varepsilon$. Then there must exist an interval $[i,j] \in C$ (otherwise C would be not well-nested). Then $w_i w_{i+1} \cdots w_j$ represents 1 in G. Consider the word $w' = w_1 \cdots w_{i-1} w_{j+1} \cdots w_m$. Let C' be obtained from $C \setminus \{[i,j]\}$ by replacing every occurrence of an index $k \geq j+1$ in $C \setminus \{[i,j]\}$ by $k - j + i - 1$. Then C' is a weak cancellation for w'. Hence, w' represents 1, which implies that w represents 1. □

Of course, when showing that the solution set of (16) has a polynomial magnitude, we may assume that $g_i \neq 1$ for any $i \in [1,k]$. Moreover, we lose no generality by assuming that all words u_i, $i \in [1,k]$ and v_i, $i \in [0,k]$ are reduced. Furthermore, we may assume that each g_i is cyclically reduced. Indeed, if some g_i is not cyclically reduced, we can write $g_i = f^{-1} g f$ for some cyclically reduced g and replace h_{i-1}, g_i, and h_i by $h_{i-1} f^{-1}$, $g = f g_i f^{-1}$, and $f h_i$, respectively. This does not change the solution set because

$$h_{i-1} f^{-1} (f g_i f^{-1})^{x_i} f h_i = h_{i-1} g_i^{x_i} h_i.$$

Moreover, if we do this replacement for each g_i that is not cyclically reduced, we increase the size of the instance by at most $2|g_1| + \cdots + 2|g_k| \leq 2n$ (note that $|g| = |g_i|$). Applying this argument again, we may even assume that

$$u_i \in A_0^+ \cup A_1^+ \cup A_0^+ A^* A_1^+ \cup A_1^+ A^* A_0^+ \tag{17}$$

for every $i \in [1,k]$. Note that λ and ρ are bijections on words of this form.

Consider a solution $(x_1, \ldots, x_k)$ to (16). Then the word

$$w = v_0 u_1^{x_1} v_1 \cdots u_k^{x_k} v_k \tag{18}$$

represents 1 in G. We factorize each v_i, $i \in [0,k]$, and each u_i, $i \in [1,k]$, into its syllables. These factorizations define a factorization $w = w_1 \cdots w_m$ and we call this the *block factorization* of w. This is the coarsest refinement of the factorization $w = v_0 u_1^{x_1} v_1 \cdots u_k^{x_k} v_k$ and of w's factorization into syllables. The numbers $1, 2, \ldots, m$ are the *blocks* of w. We fix this factorization $w = w_1 \cdots w_m$ for the rest of this section.

Cycles and certified solutions. In the representation $v_0 u_1^{x_1} v_1 \cdots u_k^{x_k} v_k = 1$ of (16), the words $u_1, \ldots, u_k$ are called the *cycles*. If $u_i \in A_0^+ \cup A_1^+$, the cycle u_i is said to be *simple* and otherwise *mixed* (note that $u_i = \varepsilon$ cannot happen because $g_i \neq 1$). Let p be a block of w. If w_p is contained in some $u_i^{x_i}$ for a cycle u_i, then p is a *u_i-blocks* or *block from u_i*. If w_p is contained in some v_i, then p is a *v_i-block* or a *block from v_i*. A *certified solution* is a pair (x, C), where x is a solution to (16) and C is a cancellation of the word w as in (18).

Observe that if $C \subseteq 2^{[1,m]}$ is a cancellation for $w = w_1 \cdots w_m$ then by maximality, for each simple cycle u_i, all u_i-blocks are contained in the same edge of C. We will also need the following two auxiliary lemmas.

Lemma 4.19 *Let C be a cancellation. If i, j are two distinct blocks from the same mixed cycle, then there is no edge $I \in C$ with $i, j \in I$.*

Proof Suppose there is such an $I \in C$. Furthermore, assume that i and j are chosen so that $|i - j|$ is minimal and $i < j$. Since $i, j \in I$, we have $w_i w_j \in A_0^+ \cup A_1^+$ by consistency of C. Hence, i and j cannot be neighbors. Therefore there is an $\ell \in [1,m]$ with $i < \ell < j$. This means there is a $J \in C$ with $\ell \in J$. By well-nestedness, $J \subseteq [i,j]$. Since every edge in C must contain at least two elements, we have $|J| \geq 2$ and thus a contradiction to the minimality of $|i - j|$. □

An edge $I \in C$ is called *standard* if $|I| = 2$ and the two blocks in I are from mixed cycles. Intuitively, the following lemma tells us that in a cancellation, most edges are standard.

Lemma 4.20 *Let C be a cancellation and u_i be a mixed cycle. Then there are at most $n + 3k + 1$ non-standard edges $I \in C$ containing a u_i-block.*

Proof Let $N \subseteq C$ be the set of all non-standard edges $I \in C$ that contain a u_i-block. Then, each edge $I \in N$ satisfies one of the following.

(i) I contains a block from some simple cycle. There are at most k such I.
(ii) I contains a block from some v_j, $j \in [0,k]$. Since $\|v_0\| + \cdots + \|v_k\| \leq n$, there are at most n such I.
(iii) I contains only blocks from mixed cycles and $|I| > 2$.

Let $M \subseteq C$ be the set of edges of type (iii). If we can show that $|M| \leq 2k+1$, then the lemma is proven. Consider the sets

$$M_- = \{I \in M \mid I \text{ contains a block from a mixed cycle } u_j,\ j < i\},$$
$$M_+ = \{I \in M \mid I \text{ contains a block from a mixed cycle } u_j,\ j > i\}.$$

We shall prove that $|M_- \cap M_+| \leq 1$ and that $|M_+ \setminus M_-| \leq k$. By symmetry, this also means $|M_- \setminus M_+| \leq k$ and thus $|M| = |M_- \cup M_+| \leq 2k+1$.

Suppose $I_1, I_2 \in M_- \cap M_+$, $I_1 \neq I_2$. Let $r \in I_1$ and $s \in I_2$ such that r and s are blocks from u_i, say with $r < s$. Since $I_1 \in M_+$, I_1 contains a block r' from a mixed cycle u_j, $j > i$. This means in particular $s < r'$. By well-nestedness, this implies $I_2 \subseteq [r, r']$, so that I_2 cannot contain a block from a mixed cycle u_ℓ with $\ell < i$, contradicting $I_2 \in M_-$. Thus, $|M_- \cap M_+| \leq 1$.

In order to prove $|M_+ \setminus M_-| \leq k$, we need another concept. For each $I \in M_+$, there is a maximal $j \in [1,k]$ such that u_j is a mixed cycle and I contains a block from u_j. Let $\mu(I) = j$. We will show $\mu(I_1) \neq \mu(I_2)$ for all $I_1, I_2 \in M_+ \setminus M_-$, $I_1 \neq I_2$. This clearly implies $|M_+ \setminus M_-| \leq k$.

Suppose $I_1, I_2 \in M_+ \setminus M_-$, $I_1 \neq I_2$, with $\mu(I_1) = \mu(I_2)$. Let $j = \mu(I_1) = \mu(I_2)$. Let r be a block from u_i contained in I_1 and let r' be a block from u_i contained in I_2. (Recall that those exist because $I_1, I_2 \in M$.) Without loss of generality, assume $r < r'$. Moreover, let s be a block from u_j contained in I_1 and let s' be a block from u_j contained in I_2. Thus, we have $r < r' < s'$.

However, we have $|I_1| > 2$, meaning I_1 contains a block p other than r and s. Since an edge cannot contain two blocks of one mixed cycle (Lemma 4.19), p has to belong to a mixed cycle u_t other than u_i and u_j. By the maximality of j, we have $i < t < j$. This implies, however, $r < r' < p < s'$, which contradicts well-nestedness. □

Mixed periods. From now on, for each $i \in [1,k]$, we use e_i to denote the i-th unit vector in $\mathbb{N}^k$, i.e. the vector with 1 in the i-th coordinate and 0 otherwise. A *mixed period* is a vector $\pi \in \mathbb{N}^k$ of the form $\|u_j\| \cdot e_i + \|u_i\| \cdot e_j$, where u_i and u_j are mixed cycles. Let $\mathbb{P} \subseteq \mathbb{N}^k$ be the set of mixed periods. Note that $|\mathbb{P}| \leq k^2$.

We will need a condition that guarantees that a given period $\pi \in \mathbb{P}$ can be added to a solution x to obtain another solution. Suppose we have two blocks p and q for which we know that if we insert a string f_1 to the left of w_p and a string f_2 to the right of w_q and $f_1 f_2$ cancels to 1 in G, then the whole word cancels to 1. Which string would we insert to the left of w_p and to the right of w_q if we build the solution $x + \pi$?

Suppose p is a u_i-block and q is a u_j-block. Moreover, let r be the first (left-most) u_i-block and let s be the last (right-most) u_j-block. If we add $\|u_j\| \cdot e_i$ to x, this inserts $\lambda^{p-r}(u_i^{\|u_j\|})$ to the left of w_p: Indeed, in the case $p = r$, we insert $u_i^{\|u_j\|}$; and when p moves one position to the right, the inserted string is rotated once to the left. Similarly, if we add $\|u_i\| \cdot e_j$ to x, we insert $\rho^{s-q}(u_j^{\|u_i\|})$ to the right of w_q: This is clear for $q = s$ and decrementing q means rotating the inserted string to the right. This motivates the following definition.

Let (x, C) be a certified solution and let u_i and u_j be mixed cycles with $i < j$. Moreover, let $r \in [1, m]$ be the left-most u_i-block and let $s \in [1, m]$ be the right-most u_j-block. Then the mixed period $\pi = \|u_j\| \cdot e_i + \|u_i\| \cdot e_j$ is *compatible with* (x, C) if there are a u_i-block p and a u_j-block q such that

$$\{p, q\} \in C \text{ and } \lambda^{p-r}(u_i^{\|u_j\|})\rho^{s-q}(u_j^{\|u_i\|}) \text{ represents } 1 \text{ in } G. \tag{19}$$

With $\mathbb{P}(x, C)$, we denote the set of mixed periods that are compatible with (x, C). One might wonder why we require an edge $\{p, q\} \in C$. In order to guarantee that $\lambda^{p-r}(u_i^{\|u_j\|})$ and $\rho^{s-q}(u_j^{\|u_i\|})$ can cancel, it would be sufficient to merely forbid edges $I \in C$ that intersect $[p, q]$ and contain a block outside of $[p-1, q+1]$. However, this weaker condition can become false when we insert other mixed periods. Our stronger condition is preserved, which implies:

Lemma 4.21 *Let (x, C) be a certified solution. Then every $x' \in x + \mathbb{P}(x, C)^{\oplus}$ is a solution.*

Proof It suffices to show that if (x, C) is a certified solution and $\pi \in \mathbb{P}(x, C)$, then there is a certified solution (x', C') such that $x' = x + \pi$ and $\mathbb{P}(x, C) \subseteq \mathbb{P}(x', C')$. Suppose $\pi = \|u_j\| \cdot e_i + \|u_i\| \cdot e_j \in \mathbb{P}(x, C)$. Without loss of generality, assume $i < j$. Let $r \in [1, m]$ be the left-most u_i-block and $s \in [1, m]$ be the right-most u_j-block in w. Since $\pi \in \mathbb{P}(x, C)$, there is a u_i-block p and a u_j-block q such that (19) holds. As explained above, we can insert $\lambda^{p-r}(u_i^{\|u_j\|})$ on the left of w_p and $\rho^{s-q}(u_j^{\|u_i\|})$ on the right of w_q and thus obtain a word w' that corresponds to the vector $x' = x + \pi$.

Both inserted words consist of $\|u_j\| \cdot \|u_i\|$ many blocks and they cancel to 1, which means we can construct a cancellation C' from C as follows. Between the two sequences of inserted blocks, we add two-element edges so that the left-most inserted u_i-block is connected to the right-most inserted u_j-block, and so forth. The blocks that existed before are connected by edges as in C. It is clear that then, C' is a partition that is consistent, cancelling and maximal. Moreover, since there is an edge $\{p, q\}$, the new edges between the inserted blocks do not violate well-nestedness: If there were a crossing edge, then there would have been one that crosses $\{p, q\}$.

It remains to verify $\mathbb{P}(x, C) \subseteq \mathbb{P}(x', C')$. A mixed period $\pi' \in \mathbb{P}(x, C) \setminus \{\pi\}$ is clearly contained in $\mathbb{P}(x', C')$ too. Hence, it remains to show $\pi \in \mathbb{P}(x', C')$. This, however, follows from the fact that instead of the edge $\{p, q\}$ that witnesses compatibility of π with (x, C), we can use its counterpart in C'; let us call this edge $\{p', q'\}$: If r' is the left-most u_i-block in w' and s' is the right-most u_j-block in w', then $p' - r' = p - r + \|u_i\| \cdot \|u_j\|$ and $s' - q' = s - q + \|u_i\| \cdot \|u_j\|$. This implies $\lambda^{p-r}(u_i^{\|u_j\|}) = \lambda^{p'-r'}(u_i^{\|u_j\|})$ and $\rho^{s-q}(u_j^{\|u_i\|}) = \rho^{s'-q'}(u_j^{\|u_i\|})$ which implies that (19) holds for the edge $\{p', q'\}$. This completes the proof of the lemma. □

We shall need another auxiliary lemma.

Lemma 4.22 *Let C be a cancellation for w. Let u_i and u_j be distinct mixed cycles. Let $D \subseteq C$ be the set of standard edges $I \in C$ that contain one block from u_i and one block from u_j. Then the set*

$$B = \{p \in [1, m] \mid p \text{ is a } u_i\text{-block and } p \in I \text{ for some } I \in D\}$$

is an interval.

Proof We prove the case $i < j$, the other follows by symmetry. Suppose there are $r_1, r_2 \in B$ such that $r_1 < r_2$ and there is no $t \in B$ with $r_1 < t < r_2$.

Towards a contradiction, suppose $r_2 - r_1 > 1$. Since $r_1, r_2 \in B$, there are $I_1, I_2 \in D$ with $r_1 \in I_1$ and $r_2 \in I_2$. Let $I_1 = \{r_1, s_1\}$ and $I_2 = \{r_2, s_2\}$. Then s_1 and s_2 are u_j-blocks and by well-nestedness, we have $r_1 < r_2 < s_2 < s_1$. Since $r_2 - r_1 > 1$, there is a t with $r_1 < t < r_2$ and therefore some $J \in C$ with $t \in J$. Since $|J| \geq 2$, there has to be a $t' \in J$, $t' \neq t$. However, well-nestedness dictates that $t' \in [r_1, s_1] \setminus [r_2, s_2]$. Since J cannot contain another block from u_i (Lemma 4.19), we cannot have $t' \in [r_1, r_2]$, which only leaves $t' \in [s_2, s_1]$. Hence, t' is from u_j. By the same argument, any block $t'' \in J \setminus \{t, t'\}$ must be from u_i or u_j, contradicting Lemma 4.19. This means $|J| = 2$ and thus $t \in B$, in contradiction to the choice of r_1 and r_2. □

Let $M \subseteq [1, k]$ be the set of $i \in [1, k]$ such that u_i is a mixed cycle. We define a new norm on vectors $x \in \mathbb{N}^k$ by setting $\|x\|_{\mathsf{M}} = \max_{i \in M} x_i$.

Lemma 4.23 *There is a polynomial q such that the following holds. For every certified solution (x, C) with $\|x\|_{\mathsf{M}} > q(n)$, there exists a mixed period $\pi \in \mathbb{P}(x, C)$ and a certified solution (x', C') such that $x' = x - \pi$ and $\mathbb{P}(x, C) \subseteq \mathbb{P}(x', C')$.*

Proof We show that the lemma holds if $q(n) \geq (n + 3k + 1) + kn^2$. (Recall that $k \leq n$.) Let (x, C) be a certified solution with $\|x\|_{\mathsf{M}} > q(n)$. Then there is a mixed cycle u_i such that $x_i > q(n)$ and hence $u_i^{x_i}$ consists of more than $q(n)$ blocks. Let $D \subseteq C$ be the set of all edges $I \in C$ that contain a block from u_i. Since an edge can contain at most one block per mixed cycle (Lemma 4.19), we have $|D| > q(n)$. Hence, Lemma 4.20 tells us that D contains more than kn^2 standard edges. Hence, there exists a mixed cycle u_j such that the set $E \subseteq D$ of standard edges $I \in D$ that consist of one block from u_i and one block from u_j satisfies $|E| > n^2$. If B_i (resp., B_j) denotes the set of blocks from u_i (resp., u_j) contained in some edge $I \in E$, then each of the sets B_i and B_j has to be an interval (Lemma 4.22) of size more than n^2.

We only deal with the case $i < j$, the case $i > j$ can be done similarly. Let us take a subinterval $[p', p]$ of B_i such that $p - p' = \|u_i\| \cdot \|u_j\| \leq n^2$. By well-nestedness and since B_j is an interval, the neighbors (with respect to the edges from E) of $[p', p]$ form an interval $[q, q'] \subseteq B_j$ as well, and we have $p - p' = q' - q = \|u_i\| \cdot \|u_j\|$. Moreover, we have an edge $\{p - \ell, q + \ell\} \in E$ for each $\ell \in [0, p - p']$. In particular, $w_{p'} w_{p'+1} \cdots w_{p-1} w_{q+1} \cdots w_{q'}$ represents 1 in G.

Let r be the left-most u_i-block and let s be the right-most u_j-block. Then, as shown before the definition of compatibility (p. 45), we have

$$\lambda^{p-r}(u_i^{\|u_j\|}) = w_{p'}w_{p'+1}\cdots w_{p-1}, \qquad \rho^{s-q}(u_j^{\|u_i\|}) = w_{q+1}w_{q+1}\cdots w_{q'}.$$

Therefore, $\lambda^{p-r}(u_i^{\|u_j\|})\rho^{s-q}(u_j^{\|u_i\|})$ represents 1 in G and $\{p,q\}$ witnesses compatibility of $\pi = \|u_j\| \cdot e_i + \|u_i\| \cdot e_j$ with (x, C). Hence, $\pi \in \mathbb{P}(x, C)$.

Let $x' = x - \pi$. We remove the factors $w_{p'}\cdots w_{p-1}$ and $w_{q+1}\cdots w_{q'}$ from w. Then, the remaining blocks spell $w' = v_0 u_1^{x'_1} v_1 \cdots u_k^{x'_k} v_k$. Indeed, recall that removing from a word y^t any factor of length $\ell \cdot |y|$ will result in the word $y^{t-\ell}$. Moreover, let C' be the set of edges that agree with C on the remaining blocks. By the choice of the removed blocks, it is clear that C' is a cancellation for w'. Hence, (x', C') is a certified solution.

It remains to verify $\mathbb{P}(x, C) \subseteq \mathbb{P}(x', C')$. First note that for every mixed cycle u_ℓ, all u_ℓ-blocks that remain in w' change their position relative to the left-most and the right-most u_ℓ-block by a difference that is divisible by $\|u_\ell\|$ (if $i \neq \ell \neq j$ then these relative positions do not change at all). Note that the expression $\lambda^{p-r}(u_i^{\|u_j\|})$ is not altered when $p - r$ changes by a difference divisible by $\|u_i\|$, and an analogous fact holds for $\rho^{s-q}(u_j^{\|u_i\|})$. Hence, the edge in C' that corresponds to the C-edge $\{p, q\}$ is a witness for $\pi \in \mathbb{P}(x', C')$. Moreover, for all other mixed periods $\pi' \in \mathbb{P}(x, C) \setminus \{\pi\}$ that are witnessed by an edge $\{t, u\} \in C$, the blocks t and u do not belong to $[p', p-1] \cup [q+1, q']$. Therefore, the corresponding edge in C' exists and serves as a witness for $\pi' \in \mathbb{P}(x', C')$. □

Lemma 4.24 *There exists a polynomial q such that the following holds. For every solution $x \in \mathbb{N}^k$, there exists a certified solution (x', C') such that $\|x'\|_{\mathsf{M}} \leq q(n)$ and $x \in x' + \mathbb{P}(x', C')^{\oplus}$.*

Proof Let q be the polynomial provided by Lemma 4.23. Since x is a solution, there is a certified solution (x, C). Repeated application of Lemma 4.23 yields certified solutions $(x_0, C_0), \ldots, (x_m, C_m)$ and mixed periods $\pi_1, \ldots, \pi_m$ such that $(x_0, C_0) = (x, C)$, $\pi_i \in \mathbb{P}(x_{i-1}, C_{i-1}) \subseteq \mathbb{P}(x_i, C_i)$, $x_i = x_{i-1} - \pi_i$, and $\|x_m\|_{\mathsf{M}} \leq q(n)$. In particular, $\mathbb{P}(x_m, C_m)$ contains each π_i and hence

$$x = x_m + \pi_1 + \cdots + \pi_m \in x_m + \mathbb{P}(x_m, C_m)^{\oplus}.$$

Thus, $(x', C') = (x_m, C_m)$ is the desired certified solution. □

We are now ready to prove Proposition 4.17 and thus Theorem 4.10.

Proof of Proposition 4.17. Suppose that p_0 and p_1 are the polynomials guaranteed by the knapsack tameness of G_0 and G_1, respectively. Recall that $S \subseteq \mathbb{N}^k$ is the set of solutions to (16). We prove that there exists a polynomial p such that for every $x \in S$ there is a semilinear set $S' \subseteq \mathbb{N}^k$ of magnitude at most $p(n)$ such that $x \in S' \subseteq S$. This clearly implies that S has magnitude at most $p(n)$. First, we apply Lemma 4.24. It yields a polynomial q and a

certified solution (x', C') with $\|x'\|_{\mathsf{M}} \leq q(n)$ such that $x \in x' + \mathbb{P}(x', C')^{\oplus}$. Let $w' = v_0 u_1^{x'_1} v_1 \cdots u_k^{x'_k} v_k$ and consider w' decomposed into blocks as we did above with w.

Let us briefly describe the idea of the remaining steps to construct S'. The semilinear set $x' + \mathbb{P}(x', C')$ already satisfies $x \in x' + \mathbb{P}(x', C') \subseteq S$. The only entries in the semilinear representation for $x' + \mathbb{P}(x', C')$ that are not polynomially bounded yet are the coordinates of x' that correspond to simple cycles. Therefore, we consider the set $T \subseteq [1, k]$ of all $i \in [1, k]$ for which the cycle u_i is simple. In order to reduce the entries at these coordinates as well, we partition T according to which edge of C' the blocks from a cycle u_i, $i \in T$, belong to. (Recall that by maximality of C', all the blocks of a simple cycle belong to the same edge.) Then, all blocks that belong to the same edge (i) belong to G_s for some $s \in \{0, 1\}$ and (ii) yield 1 in G_s. Therefore, these blocks form a solution to a knapsack instance over G_s, to which we can apply knapsack tameness of G_s.

Let us make this idea precise. Since C' is maximal, for each $i \in T$, all u_i-blocks are contained in one edge $I_i \in C'$. Note that it is allowed that one edge contains the blocks of multiple simple cycles. We partition T into sets $T = T_1 \uplus \cdots \uplus T_t$ so that $i \in T$ and $j \in T$ belong to the same part if and only if the u_i-blocks and the u_j-blocks belong to the same edge of C, i.e. $I_i = I_j$.

For a moment, let us fix an $\ell \in [1, t]$ and let $I \in C'$ be the edge containing all u_i-blocks for all the $i \in T_\ell$. Moreover, let $T_\ell = \{i_1, \ldots, i_r\}$. The words $\bar{v}_j$ for $j \in [0, r]$ will collect those blocks that belong to I but are not u_{i_s}-blocks for any $s \in [1, r]$. Formally:

1. $\bar{v}_0$ consists of all blocks that belong to I that are to the left of all u_{i_1}-blocks.
2. Similarly, $\bar{v}_r$ is the concatenation of all blocks belonging to I that are to the right of all u_{i_r}-blocks.
3. Finally, for $j \in [1, r-1]$, $\bar{v}_j$ consists of all blocks that belong to I and are to the right of all u_{i_j}-blocks and to the left of all $u_{i_{j+1}}$-blocks.

By consistency of C', for some $s \in \{0, 1\}$, all the words $\bar{v}_j$ for $j \in [0, r]$ and the words u_{i_j} for $j \in [1, r]$ belong to A_s^* and thus represent elements of G_s. Since G_s is knapsack tame, the set

$$S_\ell = \{z \in \mathbb{N}^k \mid \bar{v}_0 u_{i_1}^{z_{i_1}} \bar{v}_1 u_{i_2}^{z_{i_2}} \bar{v}_2 \cdots u_{i_r}^{z_{i_r}} \bar{v}_r \text{ represents 1 in } G_s,\ \ z_j = 0 \text{ for } j \notin T_\ell\}$$

has magnitude at most $p_s(n)$. Consider the vector $y \in \mathbb{N}^k$ with $y_i = 0$ for $i \in T$ and $y_i = x'_i$ for $i \in [1, k] \setminus T$ (i.e. when u_i is a mixed cycle). We claim that $S' = y + S_1 + \cdots S_t + \mathbb{P}(x', C')^{\oplus}$ has magnitude at most $q(n) + p_0(n) + p_1(n) + n$ and satisfies $x \in S' \subseteq S$.

First, since y and the members of $S_1, \ldots, S_t$ are non-zero on pairwise disjoint coordinates, the magnitude of $y + S_1 + \cdots + S_t$ is the maximum of $\|y\|_\infty$ and the maximal magnitude of $S_1, \ldots, S_t$. Hence, it is bounded by $q(n) + p_0(n) + p_1(n)$. The summand $\mathbb{P}(x', C')^{\oplus}$ contributes only periods, and their magnitude is bounded by n (recall that they are mixed periods). Thus, the magnitude of S' is at most $p(n) = q(n) + p_0(n) + p_1(n) + n$.

The cancelling property of (x', C') tells us that $x' - y$ is contained in the sum $S_1 + \cdots + S_t$. By the choice of (x', C'), we have $x \in x' + \mathbb{P}(x', C')^{\oplus}$. Together, this means $x \in S'$. Hence, it remains to show $S' \subseteq S$. To this end, consider a vector $x'' \in y + S_1 + \cdots + S_t$. It differs from x' only in the exponents at simple cycles. Therefore, we can apply essentially the same cancellation to x'' as to x': we just need to adjust the edges containing the blocks of simple cycles. It is therefore clear that the resulting cancellation C'' has the same compatible mixed periods as C': $\mathbb{P}(x'', C'') = \mathbb{P}(x', C')$. Thus, by Lemma 4.21, we have $x'' + \mathbb{P}(x', C')^{\oplus} \subseteq S$. This proves $S' = y + S_1 + \cdots + S_t + \mathbb{P}(x', C')^{\oplus} \subseteq S$ and hence Proposition 4.17. □

4.2.5 Proof of Theorem 4.6

Let us first consider knapsack. According to Proposition 4.7, it suffices to provide a logspace reduction from the knapsack problem over G to the membership problem for acyclic NFA over G. Suppose we have an instance

$$h_0 g_1^{x_1} h_1 \cdots g_k^{x_k} h_k = 1$$

of the knapsack problem over G of size n. Moreover, let h_i be represented by $v_i \in A^*$ for each $i \in [0, k]$ and let g_i be represented by $u_i \in A^*$ for $i \in [1, k]$.

By Theorem 4.10, there is a polynomial p such that the above instance has a solution if and only if it has a solution $x \in \mathbb{N}^k$ with $\|x\|_\infty \leq p(n)$. We construct an acyclic NFA $\mathcal{A} = (Q, A, \Delta, q_0, q_f)$ as follows. It has the state set $Q = [0, k+1] \times [0, p(n)]$ and the following transitions. From $(0, 0)$, there is one transition labeled v_0 to $(1, 0)$. For each $i \in [1, k]$ and $j \in [0, p(n) - 1]$, there are two transitions from (i, j) to $(i, j+1)$; one labeled by u_i and one labeled by ε. Furthermore, there is a transition from $(i, p(n))$ to $(i+1, 0)$ labeled v_i for each $i \in [1, k]$. The initial state is $q_0 = (0, 0)$ and the final state is $q_f = (k + 1, 0)$.

It is clear that $\mathcal{A}$ accepts a word that represents 1 if and only if the exponent equation has a solution. Finally, the reduction can clearly be carried out in logarithmic space.

For subset sum the same reduction as above works but the polynomial bound on solutions is for free.

4.3 LogCFL-completeness

In this section we complement Theorem 4.6 with a lower bound.

Theorem 4.25 *If (A, I) is a transitive forest and not a complete graph, then knapsack and subset sum for $\mathbb{G}(A, I)$ are* LogCFL*-complete.*

By Theorem 4.6 it suffices to show that knapsack and subset sum for $\mathbb{G}(A, I)$ are LogCFL-hard if (A, I) is not a complete graph. If (A, I) is not complete, then (A, I) contains two non-adjacent vertices and thus $\mathbb{G}(A, I)$ contains an isomorphic copy of F_2, the free group of rank two. Hence, we will show that knapsack and subset sum for F_2 are LogCFL-hard:

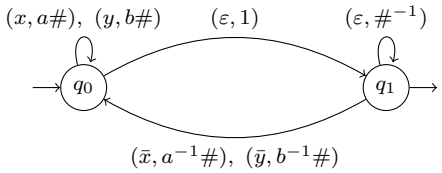

Fig. 2 Transducer used in the proof of Lemma 4.27

Proposition 4.26 *For F_2, knapsack and subset sum are* LogCFL*-hard.*

Let $\{a, b\}$ be a generating set for F_2. Let $\theta\colon \{a, b, a^{-1}, b^{-1}\}^* \to F_2$ be the morphism that maps a word w to the group element represented by w.

A *valence automaton* over a group G is a tuple $\mathcal{A} = (Q, \Sigma, \Delta, q_0, q_f)$ where Q, Σ, q_0, q_f are as in a finite automaton and Δ is a finite subset of $Q \times \Sigma^* \times G \times Q$. The *language accepted by* $\mathcal{A}$ is denoted $L(\mathcal{A})$ and consists of all words $w_1 \cdots w_n$ such that there is a computation

$$p_0 \xrightarrow{w_1, g_1} p_1 \to \cdots \to p_{n-1} \xrightarrow{w_n, g_n} p_n$$

such that $(p_{i-1}, w_i, g_i, p_i) \in \Delta$ for $i \in [1, n]$, $p_0 = q_0$, $p_n = q_f$, and $g_1 \cdots g_n = 1$ in G. We call this computation also an *accepting run* of $\mathcal{A}$ for w (of length n). Note that we allow ε-transitions of the form $(p, \varepsilon, g, q) \in \Delta$. This implies that an accepting run for a word w can be of length greater than $|w|$.

An analysis of a proof (in this case [30]) of the Chomsky-Schützenberger theorem yields:

Lemma 4.27 *For every language $L \subseteq \Sigma^*$ the following statements are equivalent:*

(i) *L is context-free.*
(ii) *There is a valence automaton $\mathcal{A}$ over F_2 such that $L = L(\mathcal{A})$.*
(iii) *There is a valence automaton $\mathcal{A}$ over F_2 and a constant $c \in \mathbb{N}$ such that $L = L(\mathcal{A})$ and for every $w \in L$ there exists an accepting run of $\mathcal{A}$ for w of length at most $c \cdot |w|$.*

Proof The equivalence of (i) and (ii) is well known (see [30]) and the implication from (iii) to (ii) is trivial. We show that (i) implies (iii). For this, we shall use the concept of rational transductions. If Σ and Γ are alphabets, subsets $T \subseteq \Gamma^* \times \Sigma^*$ are called *transductions*. Given a language $L \subseteq \Sigma^*$ and a transduction $T \subseteq \Gamma^* \times \Sigma^*$, we define

$$TL = \{u \in \Gamma^* \mid (u, v) \in T \text{ for some } v \in L\}.$$

A *finite-state transducer* is a tuple $\mathcal{A} = (Q, \Sigma, \Gamma, \Delta, q_0, q_f)$, where Q is a finite set of *states*, Σ is its *input alphabet*, Γ is its *output alphabet*, Δ is a finite subset of $Q \times \Gamma^* \times \Sigma^* \times Q$, $q_0 \in Q$ is its *initial state*, and $q_f \in Q$ is its *final state*.

The elements of Δ are called *transitions*. We say that a pair $(u, v) \in \Gamma^* \times \Sigma^*$ is *accepted by* $\mathcal{A}$ if there is a sequence

$$(p_0, u_1, v_1, p_1), (p_1, u_2, v_2, p_2), \ldots, (p_{n-1}, u_n, v_n, p_n)$$

of transitions where $n \geq 1$, $p_0 = q_0$, $p_n = q_f$, $u = u_1 \cdots u_n$, and $v = v_1 \cdots v_n$. The set of all pairs $(u, v) \in \Gamma^* \times \Sigma^*$ that are accepted by $\mathcal{A}$ is denoted by $T(\mathcal{A})$. A transduction $T \subseteq \Gamma^* \times \Sigma^*$ is called *rational* if there is a finite-state transducer $\mathcal{A}$ with $T(\mathcal{A}) = T$.

Let $W_2 \subseteq \{a, b, a^{-1}, b^{-1}\}^*$ be the word problem of F_2, i.e.

$$W_2 = \{w \in \{a, b, a^{-1}, b^{-1}\}^* \mid \theta(w) = 1\}.$$

For languages $K \subseteq \Gamma^*$ and $L \subseteq \Sigma^*$, we write $K \rightsquigarrow L$ if there is a rational transduction $T \subseteq \Gamma^* \times \Sigma^*$ and a constant c such that $K = TL$ and for each $u \in K$, there is a $v \in L$ with $|v| \leq c|u|$ and $(u, v) \in T$. Observe that the relation $\rightsquigarrow$ is transitive, meaning that it suffices to show $L \rightsquigarrow W_2$ for every context-free language L.

Let D_2 be the one-sided Dyck language over two pairs of parentheses, in other words: D_2 is the smallest language $D_2 \subseteq \{x, \bar{x}, y, \bar{y}\}^*$ such that $\varepsilon \in D_2$ and whenever $uv \in D_2$, we also have $uwv \in D_2$ for $w \in \{x\bar{x}, y\bar{y}\}$.

It is easy to see that $L \rightsquigarrow D_2$ for every context-free language L. Indeed, an ε-free pushdown automaton (which exists for every context-free language [22]) for L can be converted into a transducer witnessing $L \rightsquigarrow D_2$. Therefore, it remains to show that $D_2 \rightsquigarrow W_2$.

Let F_3 be the free group of rank 3 and let $\{a, b, \#\}$ be a free generating set for F_3. As above, let

$$W_3 = \{w \in \{a, b, \#, a^{-1}, b^{-1}, \#^{-1}\}^* \mid w \text{ represents 1 in } F_3\}$$

be the word problem of F_3. Since F_3 can be embedded into F_2 [41, Proposition 3.1], we clearly have $W_3 \rightsquigarrow W_2$. It therefore suffices to show $D_2 \rightsquigarrow W_3$.

For this, we use a construction of Kambites [30]. He proves that if $\mathcal{A}$ is the transducer in Figure 4.27 and $T = T(\mathcal{A})$, then $D_2 = TW_3$. Thus, for every $u \in D_2$, we have $(u, v) \in T$ for some $v \in W_3$. An inspection of $\mathcal{A}$ yields that $|v| = 2|u| + |v|_{\#^{-1}}$ and $|v|_{\#} = |u|$. Since $v \in W_3$, we have $|v|_{\#^{-1}} = |v|_{\#}$ and thus $|v| = 3|u|$. Hence, the transduction T witnesses $D_2 \rightsquigarrow W_3$. We have thus shown $L \rightsquigarrow D_2 \rightsquigarrow W_3 \rightsquigarrow W_2$ and hence the lemma. □

Given w, it is easy to convert the valence automaton $\mathcal{A}$ from Lemma 4.27 into an acyclic automaton that exhausts all computations of $\mathcal{A}$ of length $c \cdot |w|$. This yields the following.

Proposition 4.28 *For F_2, the membership problem for acyclic NFA is* LogCFL*-hard.*

Proof Fix a context-free language $L \subseteq \Sigma^*$ with a LogCFL-complete membership problem; such languages exist [21]. Fix a valence automaton $\mathcal{A} = (Q, \Sigma, \Delta, q_0, q_f)$ over F_2 and a constant $c \in \mathbb{N}$ such that the statement of

Lemma 4.27(iii) holds for L, $\mathcal{A}$, and c. Consider a word $w \in \Sigma^*$. From w we construct an acyclic automaton $\mathcal{B}$ over the input alphabet $\{a, b, a^{-1}, b^{-1}\}$ such that $1 \in \theta(L(\mathcal{B}))$ if and only if $w \in L$. Let $m = |w|$, $w = a_1 a_2 \cdots a_m$ and $n = c \cdot m$. The set of states of $\mathcal{B}$ is $[0, m] \times [0, n] \times Q$. The transitions of $\mathcal{B}$ are defined as follows:

- $(i-1, j-1, p) \xrightarrow{x} (i, j, q)$ if $(p, a_i, x, q) \in \Delta$ for all $i \in [1, m]$, $j \in [1, n]$, and $x \in \{a, b, a^{-1}, b^{-1}\}^*$
- $(i, j-1, p) \xrightarrow{x} (i, j, q)$ if $(p, \varepsilon, x, q) \in \Delta$ for all $i \in [0, m]$, $j \in [1, n]$, and $x \in \{a, b, a^{-1}, b^{-1}\}^*$

The initial state of $\mathcal{B}$ is $(0, 0, q_0)$ and all states (m, j, q_f) with $j \in [0, n]$ are final in $\mathcal{B}$. It is then straightforward to show that $1 \in \theta(L(\mathcal{B}))$ if and only if $w \in L$. The intuitive idea is that in a state of $\mathcal{B}$ we store in the first component the current position in the word w. In this way we enforce the simulation of a run of $\mathcal{A}$ on input w. In the second component of the state we store the total number of simulated $\mathcal{A}$-transitions. In this way we make $\mathcal{B}$ acyclic. Finally, the third state component of $\mathcal{B}$ stores the current $\mathcal{A}$-state. □

Proof of Proposition 4.26. Let $\mathcal{A} = (Q, \{a, b, a^{-1}, b^{-1}\}, \Delta, q_0, q_f)$ be an acyclic automaton. We construct words $w, w_1, \ldots, w_m \in \{a, b, a^{-1}, b^{-1}\}$ such that the following three statements are equivalent:

(i) $1 \in \theta(L(\mathcal{A}))$.
(ii) $\theta(w) \in \theta(w_1^* w_2^* \cdots w_m^*)$.
(iii) $\theta(w) \in \theta(w_1^{e_1} w_2^{e_2} \cdots w_m^{e_m})$ for some $e_1, e_2, \ldots, e_m \in \{0, 1\}$.

W.l.o.g. assume that $Q = \{1, \ldots, n\}$, where 1 is the initial state and n is the unique final state of $\mathcal{A}$.

Let $\alpha_i = a^i b a^{-i}$ for $i \in [1, n+2]$. It is well known that the α_i generate a free subgroup of rank $n+2$ in F_2 [41, Proposition 3.1]. Define the embedding $\varphi\colon F_2 \to F_2$ by $\varphi(a) = \alpha_{n+1}$ and $\varphi(b) = \alpha_{n+2}$. For a transition $t = (p, w, q) \in \Delta$ let $\tilde{t} = \alpha_p \varphi(w) \alpha_q^{-1}$. Let $\Delta = \{t_1, \ldots, t_m\}$ such that $t_i = (p, a, q)$ and $t_j = (q, b, r)$ implies $i < j$. Since $\mathcal{A}$ is acyclic, such an enumeration must exist. Together with the fact that the α_i generate a free group, it follows that the following three statements are equivalent:

(i) $1 \in \theta(L(\mathcal{A}))$.
(ii) $\theta(\alpha_1 \alpha_n^{-1}) \in \theta(\tilde{t}_1^* \tilde{t}_2^* \cdots \tilde{t}_m^*)$.
(iii) $\theta(\alpha_1 \alpha_n^{-1}) \in \theta(\tilde{t}_1^{e_1} \tilde{t}_2^{e_2} \cdots \tilde{t}_m^{e_m})$ for some $e_1, e_2, \ldots, e_m \in \{0, 1\}$.

This shows the proposition. □

4.4 TC^0-completeness

We finally show that subset sum and knapsack for free abelian groups $\mathbb{Z}^m$ are complete for the circuit complexity class TC^0. Note that $\mathbb{Z}^m$ is isomorphic to the graph group $\mathbb{G}(A, I)$ where (A, I) is the complete graph on m nodes. The proof of the following result is a simple combination of known results from [17, 46].

Theorem 4.29 *For every fixed $m \geq 1$, knapsack and subset sum for the free abelian group $\mathbb{Z}^m$ are complete for TC^0. Hence, knapsack and subset sum for $\mathbb{G}(A, I)$ are complete for TC^0 if (A, I) is a non-empty complete graph.*

Proof Hardness for TC^0 follows from the well-known fact that the word problem for $\mathbb{Z}$ is TC^0-complete: The word problem for $\mathbb{Z}$ is exactly the membership problem for the language $\mathrm{Eq} = \{w \in \{a, b\}^* \mid |w|_a = |w|_b\}$ (take $b = a^{-1}$). The canonical TC^0-complete language is $\mathrm{Maj} = \{w \in \{a, b\}^* \mid |w|_a \geq |w|_b\}$ [50], which is equivalent (with respect to AC^0-Turing reductions) to Eq: (i) $w \in \mathrm{Eq}$ if and only if $w \in \mathrm{Maj}$ and $w' \in \mathrm{Maj}$, where w' is obtained by swapping the letters a and b in w, and (ii) $w \in \mathrm{Maj}$ if and only if $\bigwedge_{i=0}^{|w|} wb^i \in \mathrm{Eq}$.

Let us now show that knapsack for $\mathbb{Z}^m$ belongs to TC^0. Let $A = \{a_1, \ldots, a_m\}$ be the generating set for $\mathbb{Z}^m$. Given a word $w \in (A \cup A^{-1})^*$ we can compute the vector $(b_1, \ldots, b_m) \in \mathbb{Z}^m$ with $b_i := |w|_{a_i} - |w|_{a_i^{-1}}$ represented in unary notation in TC^0 (counting the number of occurrences of a symbol in a string and subtraction can be done in TC^0). Hence, we can transform in TC^0 an instance of knapsack for $\mathbb{Z}^m$ into a system of equations $Ax = b$, where $A \in \mathbb{Z}^{m \times n}$ is an integer matrix with unary encoded entries, $b \in \mathbb{Z}^m$ is an integer vector with unary encoded entries, and x is a vector of n variables ranging over $\mathbb{N}$. Let $t = n(ma)^{2m+1}$, where a is the maximal absolute value of an entry in $(A \mid b)$. By [46] the system $Ax = b$ has a solution if and only if it has a solution with all entries of x from the interval $[0, t]$. Since m is a constant, the unary encoding of the number t can be computed in TC^0 (iterated multiplication can be done in TC^0). However, the question whether the system $Ax = b$ has a solution from $[0, t]^n$ is an instance of the m-integer-linear-programming problem from [17], which was shown to be in TC^0 in [17]. For subset sum for $\mathbb{Z}^m$ one can use the same argument with $t = 1$. □

5 Open problems

The following two open problems were mentioned earlier:

- Is the subset sum problem for the graph group $\mathbb{G}(\mathsf{P4})$ NP-hard? The problem belongs to NP.
- Is the class of polynomially bounded knapsack groups (i.e. those where every solvable knapsack instance has a solution where all components are bounded polynomially in the size of the knapsack instance) closed under direct products with $\mathbb{Z}$? See the remarks at the beginning of section 4.2.2.

An important class of groups with open decidability status of the knapsack problem is that of braid groups.

In [33], it is shown that knapsack is decidable for every co-context-free group. A group is *co-context-free* if the set of all words that do not represent the group identity is a context-free language. The algorithm from [33] has an exponential running time and it is open whether for every co-context-free group the knapsack problem belongs to NP.

References

1. I. J. Aalbersberg and H. J. Hoogeboom. Characterizations of the decidability of some problems for regular trace languages. *Mathematical Systems Theory*, 22:1–19, 1989.
2. I. Agol. The virtual Haken conjecture. With an appendix by Agol, Daniel Groves, and Jason Manning. *Documenta Mathematica*, 18:1045–1087, 2013.
3. S. Arora and B. Barak. *Computational Complexity — A Modern Approach*. Cambridge University Press, 2009.
4. L. Babai, R. Beals, J. Cai, G. Ivanyos, and E. M. Luks. Multiplicative equations over commuting matrices. In *Proceedings of the 7th Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 1996*, pages 498–507. ACM/SIAM, 1996.
5. A. Bertoni, G. Mauri, and N. Sabadini. Membership problems for regular and context free trace languages. *Information and Computation*, 82:135–150, 1989.
6. M. Bestvina and N. Brady. Morse theory and finiteness properties of groups. *Inventiones Mathematicae*, 129(3):445–470, 1997.
7. J.-C. Birget, A. Y. Ol′shanskii, E. Rips, and M. V. Sapir. Isoperimetric functions of groups and computational complexity of the word problem. *Annals of Mathematics. Second Series*, 156(2):467–518, 2002.
8. A. Björner and F. Brenti. *Combinatorics of Coxeter Groups*, volume 231 of *Graduate Texts in Mathematics*. Springer, New York, 2005.
9. M. Charikar, E. Lehman, A. Lehman, D. Liu, R. Panigrahy, M. Prabhakaran, A. Sahai, and A. Shelat. The smallest grammar problem. *IEEE Transactions on Information Theory*, 51(7):2554–2576, 2005.
10. J. Crisp and B. Wiest. Embeddings of graph braid and surface groups in right-angled Artin groups and braid groups. *Algebraic & Geometric Topology*, 4:439–472, 2004.
11. V. Diekert. *Combinatorics on Traces*, volume 454 of *Lecture Notes in Computer Science*. Springer, 1990.
12. V. Diekert and J. Kausch. Logspace computations in graph products. *Journal of Symbolic Computation*, 75:94–109, 2016.
13. V. Diekert and M. Lohrey. Word equations over graph products. *International Journal of Algebra and Computation*, 18(3):493–533, 2008.
14. V. Diekert and A. Muscholl. Solvability of equations in graph groups is decidable. *International Journal of Algebra and Computation*, 16(6):1047–1069, 2006.
15. V. Diekert, A. G. Myasnikov, and A. Weiß. Conjugacy in Baumslag's group, generic case complexity, and division in power circuiats. In *Symposium of the 11th Latin American Symposium, LATIN 2014*, volume 8392 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 2014.
16. V. Diekert and G. Rozenberg, editors. *The Book of Traces*. World Scientific, 1995.
17. M. Elberfeld, A. Jakoby, and T. Tantau. Algorithmic meta theorems for circuit classes of constant and logarithmic depth. *Electronic Colloquium on Computational Complexity (ECCC)*, 18:128, 2011.
18. E. Frenkel, A. Nikolaev, and A. Ushakov. Knapsack problems in products of groups. *Journal of Symbolic Computation*, 74:96–108, 2016.
19. J. von zur Gathen and M. Sieveking. A bound on solutions of linear integer equalities and inequalities. *Proceedings of the American Mathematical Society*, 72(1):155–158, 1978.
20. R. Ghrist and V. Peterson. The geometry and topology of reconfiguration. *Advances in Applied Mathematics*, 38(3):302–323, 2007.
21. S. Greibach. The hardest context-free language. *SIAM Journal on Computing*, 2(4):304–310, 1973.
22. S. A. Greibach. A new normal-form theorem for context-free phrase structure grammars. *Journal of the Association for Computing Machinery*, 12(1):42–52, 1965.
23. C. Haase. *On the complexity of model checking counter automata*. PhD thesis, University of Oxford, St Catherine's College, 2011.
24. F. Haglund and D. T. Wise. Coxeter groups are virtually special. *Advances in Mathematics*, 224(5):1890–1903, 2010.
25. W. Hesse, E. Allender, and D. A. M. Barrington. Uniform constant-depth threshold circuits for division and iterated multiplication. *Journal of Computer and System Sciences*, 65:695–716, 2002.

26. T. Hsu and D. T. Wise. On linear and residual properties of graph products. *Michigan Mathematical Journal*, 46(2):251–259, 1999.
27. O. H. Ibarra and S. Moran. Probabilistic algorithms for deciding equivalence of straight-line programs. *Journal of the Association for Computing Machinery*, 30(1):217–228, 1983.
28. R. Impagliazzo and A. Wigderson. P = BPP if E requires exponential circuits: Derandomizing the XOR Lemma. In *Proceedings of the 29th Annual ACM Symposium on the Theory of Computing, STOC 1997*, pages 220–229. ACM Press, 1997.
29. B. Jenner. Knapsack problems for NL. *Information Processing Letters*, 54(3):169–174, 1995.
30. M. Kambites. Formal languages and groups as memory. *Communications in Algebra*, 37:193–208, 2009.
31. R. M. Karp. Reducibility among combinatorial problems. In R. E. Miller and J. W. Thatcher, editors, *Complexity of Computer Computations*, pages 85–103. Plenum Press, New York, 1972.
32. D. König and M. Lohrey. Evaluating matrix circuits. In *Proceedings of the 21st International Conference on Computing and Combinatorics, COCOON 2015*, volume 9198 of *Lecture Notes in Computer Science*, pages 235–248. Springer, 2015.
33. D. König, M. Lohrey, and G. Zetzsche. Knapsack and subset sum problems in nilpotent, polycyclic, and co-context-free groups. In *Algebra and Computer Science*, volume 677 of *Contemporary Mathematics*. AMS, 2016.
34. D. Kuske and M. Lohrey. Logical aspects of Cayley-graphs: the monoid case. *International Journal of Algebra and Computation*, 16(2):307–340, 2006.
35. M. Lohrey. Algorithmics on SLP-compressed strings: A survey. *Groups Complexity Cryptology*, 4(2):241–299, 2012.
36. M. Lohrey. *The Compressed Word Problem for Groups*. SpringerBriefs in Mathematics. Springer, 2014.
37. M. Lohrey and S. Schleimer. Efficient computation in groups via compression. In *Proceedings of Computer Science in Russia, CSR 2007*, volume 4649 of *Lecture Notes in Computer Science*, pages 249–258. Springer, 2007.
38. M. Lohrey and B. Steinberg. The submonoid and rational subset membership problems for graph groups. *Journal of Algebra*, 320(2):728–755, 2008.
39. M. Lohrey and G. Zetzsche. Knapsack in graph groups, HNN-extensions and amalgamated products. In *Proceedings of the 33rd International Symposium on Theoretical Aspects of Computer Science (STACS 2016)*, volume 47 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 50:1–50:14, Dagstuhl, Germany, 2016. Schloss Dagstuhl–Leibniz-Zentrum für Informatik.
40. M. Lohrey and G. Zetzsche. The complexity of knapsack in graph groups, 2017. In *Proceedings of the 34th International Symposium on Theoretical Aspects of Computer Science (STACS 2017)*, volume 66 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 52:1–52:14, Dagstuhl, Germany, 2017. Schloss Dagstuhl–Leibniz-Zentrum für Informatik.
41. R. C. Lyndon and P. E. Schupp. *Combinatorial Group Theory*. Springer, 1977.
42. K. A. Mihaĭlova. The occurrence problem for direct products of groups. *Math. USSR Sbornik*, 70:241–251, 1966. English translation.
43. A. Muscholl and D. Peled. Message sequence graphs and decision problems on Mazurkiewicz traces. In *Proceedings of the 24th International Symposium on Mathematical Foundations of Computer Science, MFCS 1999*, number 1672 in Lecture Notes in Computer Science, pages 81–91. Springer, 1999.
44. A. Myasnikov, A. Nikolaev, and A. Ushakov. Knapsack problems in groups. *Mathematics of Computation*, 84:987–1016, 2015.
45. A. Nikolaev and A. Ushakov. Subset sum problem in polycyclic groups. *Journal of Symbolic Computation*, 84:84–94, 2018.
46. C. H. Papadimitriou. On the complexity of integer programming. *Journal of the Association for Computing Machinery*, 28(4):765–768, 1981.
47. L. Pottier. Minimal solutions of linear Diophantine systems: bounds and algorithms. In *Proceedings of the 4th International Conference on Rewriting Techniques and Applications, RTA 1991*, volume 488 of *Lecture Notes in Computer Science*, pages 162–173. Springer-Verlag, 1991.

48. I. H. Sudborough. On the tape complexity of deterministic context–free languages. *Journal of the Association for Computing Machinery*, 25(3):405–414, 1978.
49. A. W. To. Unary finite automata vs. arithmetic progressions. *Information Processing Letters*, 109(17):1010–1014, 2009.
50. H. Vollmer. *Introduction to Circuit Complexity*. Springer, 1999.
51. D. T. Wise. Research announcement: the structure of groups with a quasiconvex hierarchy. *Electronic Research Announcements in Mathematical Sciences*, 16:44–55, 2009.
52. E. S. Wolk. A note on the "The comparability graph of a tree". *Proceedings of the American Mathematical Society*, 16:17–20, 1965.