KNAPSACK PROBLEMS IN FREE PRODUCTS OF GROUPS

ELIZAVETA FRENKEL, ANDREY NIKOLAEV, AND ALEXANDER USHAKOV

Abstract. We study complexity of knapsack and related algorithmic problems in free products of groups.

Keywords. Subset sum problem, knapsack problem, bounded subgroup membership problem, free products, hyperbolic groups, nilpotent groups.
2010 Mathematics Subject Classification. 03D15, 20F65, 20F10.

Contents

1. Introduction

1.1. **Motivation.** Blah-blah, discrete optimization in groups, blahbity-blah, yak-yak-yak.

1.2. **Results and open questions.** We establish connection between **SSP**, **KP**, **BSMP** and a more general discrete optimization problem formulated in terms of graphs labeled by group elements, which we termed the *acyclic graph problem* (**AGP**). We show that **AGP** is **P**-time solvable in all known groups where **SSP** is. Further, we show that free products preserve polynomial time **AGP**, i.e. $\mathbf{AGP}(G * H) \in \mathbf{P}$ if and only if $\mathbf{AGP}(G) \in \mathbf{P}$ and $\mathbf{AGP}(H) \in \mathbf{P}$. As a consequence, we obtain that **SSP**, **KP**, **BSMP** are polynomial time solvable in a wide class of groups. We also observe that the same is false for direct products, i.e. that $\mathbf{AGP}(G) \in \mathbf{P}$, $\mathbf{AGP}(H) \in \mathbf{P}$ does not imply $\mathbf{AGP}(G \times H) \in \mathbf{P}$.

The work of the third author was partially supported by NSF grant DMS-0914773.

2. Preliminaries

2.1. **Stating the problems.** In this paper we focus mostly on subset sum, knapsack, and submonoid membership problems and their variations (described below) in a given group G generated by a finite or countably infinite set $X \subseteq G$. We refer to all such problems as *knapsack-type* problems in groups.

Elements in G are given as words over the alphabet $X \cup X^{-1}$. We begin with three principal decision problems.

The subset sum problem $\mathbf{SSP}(G,X)$: Given $g_1, \ldots, g_k, g \in G$ decide if

$$g = g_1^{\varepsilon_1} \ldots g_k^{\varepsilon_k} \tag{1}$$

for some $\varepsilon_1, \ldots, \varepsilon_k \in \{0,1\}$.

Remark 2.1. As we show in Subsection 2.4 (Lemma 2.5), if X and Y are two *finite* generating sets for a group G, $\mathbf{SSP}(G,X)$ is in $\mathbf{P}$ if and only if $\mathbf{SSP}(G,Y)$ is in $\mathbf{P}$. However, if at least one of the sets X and Y is infinite, the same is false in general (see Example 2.2). With that in mind, we often write $\mathbf{SSP}(G)$ if a finite generating set is implied, or if the generating set is fixed explicitly. We also often write $\mathbf{SSP}$ instead of $\mathbf{SSP}(G)$ when we talk about the problem in general, or when the group G is clear from the context. The same applies to all problems we introduce here and in Section 2.3.

The knapsack problem $\mathbf{KP}(G,X)$: Given $g_1, \ldots, g_k, g \in G$ decide if

$$g =_G g_1^{\varepsilon_1} \ldots g_k^{\varepsilon_k} \tag{2}$$

for some non-negative integers $\varepsilon_1, \ldots, \varepsilon_k$.

There is also a variation of this problem, termed *integer knapsack problem* (**IKP**), when the coefficients ε_i are arbitrary integers. However, it is easy to see that **IKP** is **P**-time reducible to **KP** for any group G (see Section **??**).

The third problem is equivalent to **KP** in the classical (abelian) case, but in general it is a completely different problem that is of prime interest in algebra:

Submonoid membership problem $\mathbf{SMP}(G,X)$: Given elements $g_1, \ldots, g_k, g \in G$ decide if g belongs to the submonoid generated by $g_1, \ldots, g_k$ in G, i.e., if the following equality holds for some $g_{i_1}, \ldots, g_{i_s} \in \{g_1, \ldots, g_k\}, s \in \mathbb{N}$:

$$g = g_{i_1}, \ldots, g_{i_s}. \tag{3}$$

The restriction of **SMP** to the case when the set of generators $\{g_1, \ldots, g_n\}$ is closed under inversion (so the submonoid is actually a subgroup of G) is a well-known problem in group theory, called the *generalized word problem* (**GWP**) or the *uniform subgroup membership problem* in G. There is a huge bibliography on this subject, we mention some related results in Section **??**.

As usual in complexity theory, it makes sense to consider the *bounded* versions of **KP** and **SMP**, at least they are always decidable in groups where the word problem is decidable. In this case the problem is to verify if the corresponding equalities (2) and (3) hold for a given g provided that the number of factors in these equalities is bounded by a natural number m which is given in the unary form, i.e., as the word 1^m. In particular, the bounded knapsack problem (**BKP**) for a group G asks to decide, when given $g_1, \ldots, g_k, g \in G$ and $1^m \in \mathbb{N}$, if the equality (2) holds for some $\varepsilon_i \in \{0, 1, \ldots, m\}$. This problem is **P**-time equivalent to **SSP** in G (see Section

2.4), so it suffices for our purposes to consider only **SSP** in groups. On the other hand, the bounded **SMP** in G is very interesting in its own right.

Bounded submonoid membership problem BSMP(G,X)**:** Given $g_1, \dots g_k, g \in G$ and $1^m \in \mathbb{N}$ (in unary) decide if g is equal in G to a product of the form $g = g_{i_1} \cdots g_{i_s}$, where $g_{i_1}, \dots, g_{i_s} \in \{g_1, \dots, g_k\}$ and $s \leq m$.

There are also interesting and important *search* variations of the decision problems above, when the task is to find an actual solution to equations (1), (2), or (3), provided that some solution exists (see Section **??** for more details on this). In most cases we solve both the decision and search variations of the problems above simultaneously, while establishing the time complexity upper bounds for the algorithms. However, as in the classical case, perhaps the most interesting variations of the search problems are their *optimization* versions. It seems these problems were never formally stated before for groups, so we discuss them in a bit more detail here, leaving a more thorough discussion for Section **??**.

The subset-sum optimization problem SSOP(G,X)**:** Given an instance $g_1, \dots, g_k, g \in G$ of **SSP**(G) find a solution, if it exists, $\varepsilon_1, \dots, \varepsilon_k \in \{0,1\}$ subject to the optimization condition that the sum $\sum_i \varepsilon_i$ is minimal. Otherwise, output *No solutions*.

The knapsack optimization problem KOP(G,X)**:** solve the equation (2) with the minimum possible number of factors.

In fact, in Section **??** we also discuss other variations of **KOP** in groups, which are even more direct generalization of the classical **KOP**. In this case when given $g_1, \dots, g_k, g \in G$ one has to find $\varepsilon_1, \dots, \varepsilon_k \in \mathbb{N}$ for which the product $g_1^{\varepsilon_1} \dots g_k^{\varepsilon_k}$ is as close to g (in the metric of the Cayley graph of G) as possible.

The submonoid membership optimization problem SMOP(G,X)**:** Given $g_1, \dots, g_k, g \in G$, express (if possible) g as a product

$$g =_G g_{i_1} \cdots g_{i_m} \tag{4}$$

with the minimum number of factors m.

The submonoid membership optimization problem plays an important part in geometric group theory. Indeed, in geometric language it asks to find a geodesic of a given element in a group (relative to a fixed finite generating set) or the distortion of a given element in a given finitely generated subgroup — both are crucial geometric tasks.

Sometimes (like in hyperbolic groups) the time complexity of the search **SMP** is not bounded from above by any computable function, in this case it makes sense to consider the optimization version of the bounded **SMP**, called **BSMOP**, in which one has to solve **BSMP**(G) with the minimal possible number of factors.

The formal description of the problems above depends on the given finite (or sometimes countable) generating set X of the group G. To this end if Π is any of the algorithmic problems above then by $\Pi(G,X)$ we denote this problem for the group G relative to the generating set X. It is not hard to show (see Section **??**) that for a given such Π, provided it is not an optimization problem, replacing one finite set of generators of G by another one ends up in a problem that is **P**-time reducible to the initial one. Therefore, the time complexity of such problems does not depend on a finite generating set, it is an invariant of the group G. In view of this, we omit X from the notation and denote such problems by $\Pi(G)$.

The typical groups we are interested in here are free, hyperbolic, abelian, nilpotent, or metabelian. In all these groups, and this is important, the word problem is decidable in **P**-time. We might also be interested in constructing some exotic examples of groups where the problems mentioned above have unexpected complexity.

2.2. **Algorithmic set-up.** Since the knapsack-type problems were not previously studied in non-commutative setting it is worthwhile to say a few words on how we present the data, models of computations, size functions, etc. (we refer to the book [2] for more details). Our model of computation is RAM (random access machines).

To make the statements of the problems (from Section 2.1) a bit more precise consider the following. If a generating set $X = \{x_1, \ldots, x_n\}$ of a group G is finite, then the *size* of the word $g = x_1 \ldots x_k$ is its length $|g| = k$ and the size of the tuple $g_1, \ldots, g_k, g$ from G is the total sum of the lengths $|g_1| + \ldots + |g_k| + |g|$.

If the generating set X of G is infinite, then the size of a letter $x \in X$ is not necessarily equal to 1, it depends on how we represent elements of X. In what follows we always assume that there is an efficient injective function $\nu : X \to \{0,1\}^*$ which encodes the elements in X such that for every $u \in \{0,1\}^*$ one can algorithmically recognize if $u \in \nu(X)$, or not. In this case for $x \in X$ define:

$$\text{size}(x) = |\nu(x)|$$

and for a word $w = x_1 \ldots x_n$ with $x_i \in X$ define:

$$\text{size}(w) = \text{size}(x_1) + \ldots + \text{size}(x_n).$$

Similar to the above the size of a tuple $(g_1, \ldots, g_k, g)$ is:

$$\text{size}(g_1, \ldots, g_k, g) = \text{size}(g_1) + \ldots + \text{size}(g_k) + \text{size}(g).$$

One can go a bit further and identify elements $x \in X$ with their images $\nu(x) \in \{0,1\}^*$, and words $w = x_1 \ldots x_n \in X^*$ with the words $\nu(x_1) \ldots \nu(x_n) \in \{0,1\}^*$. This gives a homomorphism of monoids $\nu^* : X^* \to \{0,1\}^*$. If in addition ν is such that for any $x, y \in X$ the word $\nu(x)$ is not a prefix of $\nu(y)$ (this is easy to arrange), then:

- ν^* is injective,
- $\nu^*(X^*)$ and $\nu^*(X)$ are algorithmically recognizable in $\{0,1\}^*$,
- and for every word $v \in \nu^*(X^*)$ one can find the word $w \in X^*$ such that $\nu^*(w) = v$.

From now on we always assume that a generating set comes equipped with a function ν, termed *encoding*, satisfying all the properties mentioned above. In fact, almost always all our generating sets X are finite, and in those rare occasions when X is infinite we describe ν precisely.

In general, we view decision problems as pairs (I, D), where I is the space of instances of the problem equipped with a size function size : $I \to \mathbb{N}$ and a set $D \subseteq I$ of affirmative (positive) instances of the problem. Of course, the set I should be constructible and size function should be computable. In all our examples the set I consists either of tuples of words $(g_1, \ldots, g_k, g)$ in the alphabet Σ_X for some (perhaps, infinite) set of generators X of a group G, or, in the case of **BKP** or **BSMP**, tuples of the type $(g_1, \ldots, g_k, g, 1^m)$ where 1^m is a natural number m given in unary. The problem (I, D) is decidable if there is an algorithm $\mathcal{A}$ that for any $x \in I$ decides whether x is in D or not ($\mathcal{A}$ answers "Yes" or "No"). The problem is in class **P** if there is a decision algorithm $\mathcal{A}$ with polynomial time function with respect to the size of the instances in I, i.e., there is a polynomial $p(n)$ such that for any $x \in I$ the algorithm $\mathcal{A}$ starts on x, halts in at most $p(\text{size}(x))$ steps, and gives a correct answer "Yes" or "No". Similarly, we define problems in linear or quadratic time, and non-deterministic polynomial time **NP**.

Recall that a decision problem (I_1, D_1) is **P**-*time reducible* to a problem (I_2, D_2) if there is a **P**-time computable function $f : I_1 \to I_2$ such that for any $u \in I_1$ one has $u \in D_1 \iff f(u) \in D_2$. Such reductions are usually called either *many-to-one* **P**-time reductions or Karp reductions. Since we do not use any other reductions we omit "many-to-one" from the name and call them **P**-time reductions. Similarly, one can introduce linear or quadratic time reductions, etc. We say that two problem are **P**-time equivalent if each of them **P**-time reduces to the other.

Now we define an optimization problem as a tuple $(I, J, F, \mu, \text{extr})$, where I is the set of instances, J is a set of solutions, $F : I \to P(J)$ is a function that for each instance $u \in I$ associate a subset $F(u) \subseteq J$ of all feasible solutions for an instance u, $\mu(u, v)$ is a non-negative real function which for $u \in I, v \in F(u)$ measures the cost of solution v for an instance u, extr is either min or max. This optimization problem, given $u \in I$, asks to find $v \in F(u)$ such that $\mu(u, v) = \text{extr}\{\mu(u, v') \mid v' \in F(u)\}$. Given two optimization problems $P_i = (I_i, J_i, F_i, \mu_i, \text{extr}_i)$, $i = 1, 2$, we say that P_1 is **P**-*time reducible* to P_2 if there is a **P**-time computable functions $f : I_1 \to I_2$, $f_u : F_1(u) \to F_2(f(u)), u \in I_1$, such that $v \in F_1(u) \iff f_u(v) \in F_2(f(u))$ and $\mu_1(u, v) = \text{extr}_1\{\mu_1(u, v') \mid v' \in F_1(u)\} \iff \mu_2(f(u), f_u(v)) = \text{extr}_2\{\mu_2(f(u), v') \mid v' \in F_2(f(u))\}$. In our consequent considerations, the functions f_u are apparent from the set up and we do not mention them in our arguments. We say that two optimization problem are **P**-time equivalent if each of them **P**-time reduces to the other.

2.3. **More on the formulation of the problems.** In this section we continue the discussion from the introduction on different variations of the problems **SSP**, **KP**, **SMP** in groups.

There are two ways to state search variations of the problems: the first one, as described in the introduction, considers only feasible instances of the problem, i.e. we assume that a solution to the instance exists; the second one is stronger, in this case it is required to solve the decision problem and simultaneously find a solution (if it exists) for a given instance. The former requires only a partial algorithm, while the latter asks for a total one. The weaker version of the problems **SSP**(G), **KP**(G), **SMP**(G) is always decidable in groups G with decidable word problem, while the stronger one might be undecidable (for instance, **SMP** in hyperbolic groups). In this paper we consider the stronger version of the search problems.

We mentioned in the introduction that the knapsack optimization problem (**KOP**) may have different formulations in the non-commutative groups. Now we explain what we meant.

Recall first, that perhaps the most typical version of the classical **KOP** asks, when given positive integers $a_1, \ldots, a_k, a$ to find $\varepsilon_1, \ldots, \varepsilon_k \in \mathbb{N}$ such that the sum $\varepsilon_1 a_1 + \ldots + \varepsilon_k a_k$ is less or equal to a but maximal possible under this restriction. One can generalize this to non-commutative groups as follows.

KOP1(G, X)**:** Given $g_1, \ldots, g_k, g \in G$ find $\varepsilon_1, \ldots, \varepsilon_k \in \mathbb{N} \cup \{0\}$ with the least possible distance between g and $g_1^{\varepsilon_1} \ldots g_k^{\varepsilon_k}$ in the Cayley graph $Cay(G, X)$.

This formulation allows solutions with the "total weight" higher than the capacity of the knapsack. To define precisely when a given solution fits in geometrically in the knapsack we need the following. For elements $g, h, u \in G$ we say that u *belongs to the segment* $[g, h]$ if there is a geodesic path in $Cay(G, X)$ from g to h that contains u. Now we can formulate the problem.

KOP2(G, X)**:** Given $g_1, \ldots, g_k, g \in G$ find $\varepsilon_1, \ldots, \varepsilon_k \in \mathbb{N} \cup \{0\}$ such that $g_1^{\varepsilon_1} \ldots g_k^{\varepsilon_k}$ belongs to the segment $[1, g]$ and the distance between g and $g_1^{\varepsilon_1} \ldots g_k^{\varepsilon_k}$ in the Cayley graph $Cay(G, X)$ is the least possible.

We formulate similar generalizations for the subset sum problem.

SSOP1(G, X)**:** Given $g_1, \ldots, g_k, g \in G$ find $\varepsilon_1, \ldots, \varepsilon_k \in \{0, 1\}$ such that the distance between g and $g_1^{\varepsilon_1} \ldots g_k^{\varepsilon_k}$ in the Cayley graph $Cay(G, X)$ is the least possible.

SSOP2(G, X)**:** Given $g_1, \ldots, g_k, g \in G$ find $\varepsilon_1, \ldots, \varepsilon_k \in \{0, 1\}$ such that the $g_1^{\varepsilon_1} \ldots g_k^{\varepsilon_k}$ belongs to the segment $[1, g]$ and the distance between g and $g_1^{\varepsilon_1} \ldots g_k^{\varepsilon_k}$ in the Cayley graph $Cay(G, X)$ is the least possible.

One can also consider optimization problems relative to a given non-trivial "weight" function $c : G \to \mathbb{R}$. For example, instead of optimizing $m \to \min$ in (4), one can ask to optimize $\sum c(g_{i_j}) \to \min$. Notice that the optimization problems above correspond to the case when the weight function c is a constant function $c = 1$ on G.

2.4. **Examples and basic facts.** The classical **SSP** is the following algorithmic question. Given $a_1, \ldots, a_k \in \mathbb{Z}$ and $M \in \mathbb{Z}$ decide if

$$M = \varepsilon_1 a_1 + \ldots + \varepsilon_k a_k$$

for some $\varepsilon_1, \ldots, \varepsilon_k \in \{0, 1\}$. It is well known (see [1, 3, 4]) that if the numbers in **SSP** are given in binary, then the problem is **NP**-complete, but if they are given in unary, then the problem is in **P**. The examples below show how these two variations of **SSP** appear naturally in the group theory context.

Example 2.2. Three variations of subset sum problem for $\mathbb{Z}$:

- **SSP**$(\mathbb{Z}, \{1\})$ is linear-time equivalent to the classical **SSP** in which numbers are given in unary. In particular, **SSP**$(\mathbb{Z}, \{1\})$ is in **P**.
- For $n \in \mathbb{N} \cup \{0\}$ put $x_n = 2^n$. The set $X = \{x_n \mid n \in \mathbb{N} \cup \{0\}\}$ obviously generates $\mathbb{Z}$. Fix an encoding $\nu : X^{\pm 1} \to \{0, 1\}^*$ for $X^{\pm 1}$ defined by
$$\begin{cases} x_i & \overset{\nu}{\mapsto} & 0101(00)^i 11, \\ -x_i & \overset{\nu}{\mapsto} & 0100(00)^i 11. \end{cases}$$
Then **SSP**$(\mathbb{Z}, X)$ is **P**-time equivalent to its classical version where the numbers are given in binary form. In particular, **SSP**$(\mathbb{Z}, X)$ is **NP**-complete.
- Let $X = \{2^n \mid n \in \mathbb{N} \cup \{0\}\}$ and the number 2^n is represented by the word $01(00)^{2^n} 11$ (unary representation). Then **SSP**$(\mathbb{Z}, X)$ is in **P**.
- One can easily define **SSP** and **KP** in arbitrary algebras A over a field. These problems are equivalent to **SSP** and **KP** in the additive group A^+ of A. $\square$

The first example is of no surprise, of course, since, by definition, we treat words representing elements of the group as in unary. The second one shows that there might be a huge difference in complexity of **SSP**(G, X) for finite and infinite generating sets X. The third one indicates that if X is infinite then it really matters how we represent the elements of X.

Definition 2.3. Let G and H be groups generated by countable sets X and Y with encodings ν and μ, respectively. A homomorphism $\varphi : G \to H$ is called $\mathbf{P}$-time computable relative to $(X, \nu), (Y, \mu)$ if there exists an algorithm that given a word $\nu(u) \in \nu(\Sigma_X^*)$ computes in polynomial time (in the size of the word $\nu(u)$) a word $\mu(v) \in \mu(\Sigma_Y^*)$ representing the element $v = \varphi(u) \in H$.

Example 2.4. Let G_i be a group generated by a set X_i with encoding ν_i, $i = 1, 2$. If X_1 is finite then any homomorphism $\varphi : G_1 \to G_2$ is $\mathbf{P}$-time computable relative to $(X_1, \nu_1), (X_2, \nu_2)$.

To formulate the following results put

$$\begin{aligned} \mathcal{DP} &= \{\mathbf{SSP}, \mathbf{KP}, \mathbf{SMP}, \mathbf{BKP}, \mathbf{BSMP}\}, \\ \mathcal{OP} &= \{\mathbf{SSOP}, \mathbf{SSOP}1, \mathbf{SSOP}2, \mathbf{KOP}, \mathbf{KOP}1, \mathbf{KOP}2, \mathbf{SMOP}, \mathbf{BSMOP}\}, \\ \mathcal{P} &= \mathcal{DP} \cup \mathcal{OP}. \end{aligned}$$

Lemma 2.5. *Let G_i be a group generated by a set X_i with an encoding ν_i, $i = 1, 2$. If $\phi : G_1 \to G_2$ is a $\mathbf{P}$-time computable embedding relative to $(X_1, \nu_1), (X_2, \nu_2)$ then $\mathbf{\Pi}(G_1, X_1)$ is $\mathbf{P}$-time reducible to $\mathbf{\Pi}(G_2, X_2)$ for any problem $\mathbf{\Pi} \in \mathcal{P}$.*

Proof. Straightforward. □

In view of Example 2.4 we have the following result.

Proposition 2.6. If X and Y are finite generating sets for a group G, then $\mathbf{\Pi}(G, X)$ is $\mathbf{P}$-time equivalent to $\mathbf{\Pi}(G, Y)$ for any problem $\mathbf{\Pi} \in \mathcal{P}$.

Proposition 2.7. Let G be a group and X a generating set for G. Then the word problem ($\mathbf{WP}$) for G is $\mathbf{P}$-time reducible to $\mathbf{\Pi}(G, X)$ for any problem $\mathbf{\Pi} \in \mathcal{P}$.

Proof. Let $w = w(X)$. Then $w = 1$ in G if and only if $1^\varepsilon = w$ in G for some $\varepsilon \in \{0, 1\}$, i.e., if and only if the instance $1, w$ of $\mathbf{SSP}(G)$ is positive. Likewise for other problems from $\mathcal{P}$. □

Corollary 2.8. Let G be a group with a generating set X. Then:

1) $\mathbf{SSP}(G, X)$ (or $\mathbf{BKP}(G, X)$, or $\mathbf{BSMP}(G, X)$) is decidable if and only if the word problem for G is decidable.
2) If the word problem for G is $\mathbf{NP}$-hard, then $\mathbf{\Pi}(G, X)$ is $\mathbf{NP}$-hard too for any $\mathbf{\Pi} \in \mathcal{DP}$.

This corollary shows that from $\mathbf{SSP}$ viewpoint groups with polynomial time decidable word problem are the most interesting.

The following result shows how decision version of $\mathbf{SSP}(G)$ gives a search algorithm to find an actual sequence of ε_i's that is a particular solution for a given instance of $\mathbf{SSP}(G)$.

Proposition 2.9. For any group G the search $\mathbf{SSP}(G)$ is $\mathbf{P}$-time Turing reducible to the decision $\mathbf{SSP}(G)$. In particular, if $\mathbf{SSP}(G)$ is in $\mathbf{P}$ then search $\mathbf{SSP}(G)$ is also in $\mathbf{P}$.

Proof. The argument is rather known, so we just give a quick outline to show that it works in the non-commutative case too. Let $w_1, \ldots, w_k, w$ be a given instance of $\mathbf{SSP}(G)$ that has a solution in G. To find a solution $\varepsilon_1, \ldots, \varepsilon_k \in \{0, 1\}$ for this instance consider the following algorithm.

- Solve the decision problem for $(w_2, \ldots, w_k), w$ in G. If the answer is positive, then put $\varepsilon_1 = 0$. Otherwise put $\varepsilon_1 = 1$ and replace w with $w_1^{-1}w$.
- Continue inductively and find the whole sequence $\varepsilon_1, \ldots, \varepsilon_k$.

□

Proposition 2.10. For any group G the following hold:

1) $\mathbf{BKP}(G)$ is $\mathbf{P}$-time reducible to $\mathbf{SSP}(G)$;
2) $\mathbf{BSMP}(G)$, as well as its optimization variation, is $\mathbf{P}$-time reducible to $\mathbf{SSOP}(G)$.

Proof. Given an instance $1^m, w_1, \ldots, w_k, w$ of $\mathbf{BKP}(G)$ we consider a sequence

$$w_1, \ldots, w_1, w_2, \ldots, w_2, \ldots, w_k, \ldots, w_k, w \in G, \tag{5}$$

where each segment $w_i, \ldots, w_i$ has precisely m words w_i. Obviously, the initial instance of $\mathbf{BKP}(G)$ has a solution in G if and only if $\mathbf{SSP}(G)$ has a solution in G for the sequence (5). This establishes a $\mathbf{P}$-time reduction of $\mathbf{BKP}(G)$ to $\mathbf{SSP}(G)$.

To reduce $\mathbf{BSMP}(G)$ to $\mathbf{SSOP}(G)$ for a given instance $1^m, w_1, \ldots, w_k, w$ of $\mathbf{BSMP}(G)$ consider a sequence

$$w_1, \ldots, w_k, w_1, \ldots, w_k, \ldots, w_1, \ldots, w_k, w \in G, \tag{6}$$

where each segment $w_1, \ldots, w_k$ occurs precisely m times. Obviously, any solution of $\mathbf{BSMP}$ for a given instance gives a solution of $\mathbf{SSP}(G)$ for the sequence (6) and vice versa. Hence, solving $\mathbf{SSOP}(G)$ for the sequence (6) also solves $\mathbf{BSMP}(G)$ and $\mathbf{BSMOP}(G)$ for the initial instance. This gives a polynomial time reduction of $\mathbf{BSMP}(G)$ and $\mathbf{BSMOP}(G)$ to $\mathbf{SSOP}(G)$.

Finally, note that replacing $w_1, \ldots, w_k$ with $w_1, w_1^{-1}, \ldots, w_k, w_k^{-1}$ gives a polynomial time reduction of $\mathbf{IKP}(G)$ to $\mathbf{KP}(G)$.

□

3. Acyclic graph problem

The acyclic graph problem $\mathbf{AGP}(G, X)$: Given $g \in G$ and an acyclic oriented graph Γ labeled by words in $X \cup X^{-1}$, decide whether there is an oriented path in Γ labeled by a word w such that $w = g$ in G.

Proposition 3.1. $\mathbf{SSP}(G)$, $\mathbf{KP}(G)$, $\mathbf{BSMP}(G)$ are $\mathbf{P}$-time reducible to $\mathbf{AGP}(G)$.

Proof. Draw graphs. □

Proposition 3.2. $\mathbf{AGP}(G) \in \mathbf{P}$ for every nilpotent group G.

Proposition 3.3. $\mathbf{AGP}(G) \in \mathbf{P}$ for every hyperbolic group G.

Proposition 3.4. There exist groups G, H such that $\mathbf{AGP}(G), \mathbf{AGP}(H) \in \mathbf{P}$, but $\mathbf{AGP}(G \times H)$ is $\mathbf{NP}$-complete.

Proof. $\mathbf{BSMP}(F_2 \times F_2)$. □

4. **AGP** in free products

Theorem 4.1. *If G, H are groups such that $\mathbf{AGP}(G), \mathbf{AGP}(H) \in \mathbf{P}$ then $\mathbf{AGP}(G * H) \in \mathbf{P}$.*

Proof. Draw ε-edges in the graph. □

Corollary 4.2. SSP, **KP**, **BSMP**, **AGP** are polynomial time decidable in free products of nilpotent and hyperbolic groups in any finite amount.

5. Knapsack problem in free products

Proposition 5.1. If G, H are groups such that $\mathbf{KP}(G), \mathbf{KP}(H) \in \mathbf{P}$, then $\mathbf{KP}(G * H)$ is $\mathbf{P}$-time reducible to $\mathbf{BKP}(G * H)$.

References

[1] M. Garey and J. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman, 1979.

[2] A. G. Miasnikov, V. Shpilrain, and A. Ushakov. *Non-Commutative Cryptography and Complexity of Group-Theoretic Problems.* Mathematical Surveys and Monographs. AMS, 2011.

[3] C. Papadimitriou. *Computational Complexity.* Addison-Wesley, 1994.

[4] C. Papadimitriou and K. Steiglitz. *Combinatorial optimization: Algorithms and Complexity.* Dover Publications, 1998.

Elizaveta Frenkel, Moscow State University, GSP-1, Leninskie gory, 119991, Moscow, Russia
E-mail address: lizzy.frenkel@gmail.com

Andrey Nikolaev, Stevens Institute of Technology, Hoboken, NJ, 07030 USA
E-mail address: anikolae@stevens.edu

Alexander Ushakov, Stevens Institute of Technology, Hoboken, NJ, 07030 USA
E-mail address: aushakov@stevens.edu