Discrete Mathematics and Theoretical Computer Science . . . . . vol. 27:3 #8 (2025)

Parameterized complexity of factorization problems

Markus Lohrey Andreas Rosowski*

University of Siegen, Department ETI, Siegen, Germany

revisions 20th Feb. 2024, 14th Feb. 2025; accepted 18th Aug. 2025.

We study the parameterized complexity of the following factorization problem: given elements $\pi, \pi_1, \ldots, \pi_m$ of a monoid and a parameter k, can π be written as the product of at most (or exactly) k elements from $\pi_1, \ldots, \pi_m$? Also, two restricted variants of this problem (subset sum and knapsack) are considered. For these problems we show several new upper complexity bounds and fpt-equivalences for various classes of monoids. Finally, some new upper complexity bounds for variants of the parameterized change-making problems are shown.

Keywords: parameterized complexity, knapsack problems, change-making problems

1 Introduction

This paper deals with the parameterized complexity of factorization problems in monoids. In the most generic setting, we have a fixed class C of monoids. The input consists of a monoid $M \in \mathsf{C}$, elements $a, a_1, \ldots, a_m \in M$ and a positive integer k. The question is whether a can be written as a product of at most k elements from the list $a_1, \ldots, a_m$ (every a_i may occur several times in this product). In the parameterized setting, k is the parameter. This problem has been studied most intensively for the case, where C consists of the class of all symmetric groups S_n (S_n is the group of all permutations on the sets $\{1, 2, \ldots, n\}$). The non-parameterized variant of this problem has been introduced by Even and Goldreich under the name MGS (for minimum generator sequence) Even and Goldreich (1981), where the problem was shown to be NP-complete if k is given in unary notation. This results even holds for the case, when C is the class of abelian groups $\mathbb{Z}_2^n$ for $n \geq 1$. For the case, where the factorization length k is given in binary notation, Jerrum Jerrum (1985) proved that MGS is PSPACE-complete.

Let us now come to the parameterized factorization problem for permutations, where the parameter is the factorization length k. We denote this problem by $\mathsf{pF}[\mathsf{S}_*]$ (the $*$ indicates that the degree n of the symmetric group S_n is part of the input). It was shown by (Downey and Fellows, 1999, Chapter 11)[(i)] that $\mathsf{pF}[\mathsf{S}_*]$ is $\mathsf{W}[1]$-hard and hence unlikely to be fixed parameter tractable. On the other hand, no good upper bound is known. Interestingly, the parameterized factorization problem for transformation monoids

*Supported by the DFG research project LO 748/12-2.

(i) In Downey and Fellows (1999) the problem $\mathsf{pF}[\mathsf{S}_*]$ is called PERMUTATION GROUP FACTORIZATION.

ISSN 1365–8050

T_n (T_n is the monoid consisting of all mappings on the finite set $\{1, 2, \ldots, n\}$), which is denoted by $\mathsf{pF}[\mathsf{T}_*]$, is $\mathsf{W}[2]$-hard Cai et al. (1997). It was shown in Fernau and Bruchertseifer (2022) that $\mathsf{pF}[\mathsf{T}_*]$ is equivalent under fixed parameter reductions to the problem whether a given deterministic automaton has a synchronizing word of length at most k, where k is the parameter. The latter problem belongs to the intersection of the parameterized classes $\mathsf{A}[2]$, $\mathsf{W}[\mathsf{P}]$, and WNL (see Section 3 for a definition). In particular, $\mathsf{pF}[\mathsf{T}_*]$ also belongs to the classes $\mathsf{A}[2]$, $\mathsf{W}[\mathsf{P}]$, and WNL. Our first result improves on the membership in $\mathsf{A}[2]$ by showing that $\mathsf{pF}[\mathsf{T}_*]$ belongs to the class $\mathsf{W}^{\mathsf{func}}[2] \subseteq \mathsf{A}[2]$. The classes $\mathsf{W}^{\mathsf{func}}[2]$ and $\mathsf{W}[2]$ are both defined by parameterized model-checking problems for certain fragments of first-order logic (see Section 3 for details); the only difference is that $\mathsf{W}[2]$ restricts to formulas without function symbols, whereas $\mathsf{W}^{\mathsf{func}}[2]$ allows also function symbols.

We then proceed with showing that $\mathsf{pF}[\mathsf{S}_*]$ is equivalent with respect to fixed parameter reduction to two apparently simpler problems: the parameterized knapsack problem for symmetric groups $\mathsf{pKS}[\mathsf{S}_*]$ and the parameterized subset sum problem for symmetric groups $\mathsf{pSSS}[\mathsf{S}_*]$. The problem $\mathsf{pKS}[\mathsf{S}_*]$ is a restricted version of $\mathsf{pF}[\mathsf{S}_*]$ in the sense that only factorizations of the form $a = a_1^{x_1} a_1^{x_2} \cdots a_m^{x_m}$ with $x_1, \ldots, x_m \in \mathbb{N}$ and $\sum_{i=1}^{m} x_i \leq k$ are allowed. For $\mathsf{pSSS}[\mathsf{S}_*]$ one additionally requires $x_1, \ldots, x_m \in \{0, 1\}$. In the classical (unparameterized) setting, these problems are again NP-complete Lohrey et al. (2022).

When restricting the problems $\mathsf{pKS}[\mathsf{S}_*]$ and $\mathsf{pSSS}[\mathsf{S}_*]$ to cyclic permutation groups we will show that the resulting problems are equivalent to the parameterized variant of the classical subset sum problem for integers. This problem is known to be in $\mathsf{W}[3]$ Buss and Islam (2007) and $\mathsf{W}[1]$-hard Downey and Fellows (1995). Hence, the same complexity bounds hold for $\mathsf{pKS}[\mathsf{S}_*]$ and $\mathsf{pSSS}[\mathsf{S}_*]$ when restricted to cyclic permutation groups.

In the last section of the paper, we deal with the change-making problems introduced in Goebbels et al. (2017), which are variants of the classical knapsack and subset sum problems for integers, precise definitions can be found in Section 5. It was shown in Goebbels et al. (2017) that these change-making problems are $\mathsf{W}[1]$-hard and contained in XP (see Section 3 for a definition). We observe that by using the above mentioned result from Buss and Islam (2007) one can improve the upper bound to $\mathsf{W}[3]$. More interestingly, we also consider the approximative change making problems from Goebbels et al. (2017) that involve an objective function $f(x, y) = ax + by$ that has to be minimized. The approximative change making problems are in $\mathsf{W}[3]$ (if $b > 0$ or $a = 0$) or $\mathsf{para\text{-}NP}$-complete (if $b = 0$ and $a > 0$) (see Section 3 for a definition of $\mathsf{para\text{-}NP}$).

Related work. The knapsack problem and subset problem have been also studied intensively for finitely generated infinite groups; see Bassino et al. (2020) for a survey. In Goralčík and Koubek (1995) a variant of the factorization problem for transformation monoids has been considered, where it is asked whether for given transformations $a_1, \ldots, a_n \in T_n$ and $k \geq 1$ there is an idempotent transformation of a certain fixed rank (which is the size of the image) that can be written as a product of at most k transformations from the list $a_1, \ldots, a_n$.

2 General notations

We write $\mathrm{poly}(n)$ for an arbitrary polynomial in n. The composition fg of two functions $f, g : A \to A$ is evaluated from the left to right, i.e., $(fg)(a) = g(f(a))$. Therefore we also write af for $f(a)$. Then we have $a(fg) = afg$. In order to distinguish better between the argument a and the function f we sometimes also write $a \cdot f$ for af.

For a positive integer $N \in \mathbb{N}$ we denote by $\#(N)$ the number of bits of N. For integers $i \leq j$ we write $[i, j]$ for the interval $\{i, i+1, \ldots, j\}$. Let T_n be the set of all mappings on $[1, n]$ and S_n be the set of all permutations on $[1, n]$. With respect to composition of functions, T_n is a monoid (the *transformation monoid* on n elements) and S_n is a group (the *symmetric group* on n elements). A permutation group is a subgroup of S_n for some n.[ii] We will use standard notations for permutation groups; see e.g., Seress (2003). Permutations will be often written by their decomposition into disjoint cycles, called the *disjoint cycle decomposition*. A cycle of length two is a *transposition*. For a permutation π we denote by $\mathrm{ord}(\pi)$ the order of π, i.e., the smallest $k \geq 1$ such that π^k is the identity permutation.

Let G be a finite group. For a subset $A \subseteq G$ we denote with $\langle A \rangle$ the subgroup of G generated by the elements from A (i.e., the closure of A under the group multiplication). If $\langle A \rangle = G$ then A is called a generating set of G. For $k \geq 0$ we write $A^{\leq k}$ for the set of all products $a_1 a_2 \cdots a_l \in G$ with $l \leq k$ and $a_1, \ldots, a_l \in A$. For an element $g \in \langle A \rangle$ we denote with $|g|_A$ (the A-length of g) the smallest integer k such that $g \in A^{\leq k}$.

Several times we use the well-known Chinese remainder theorem in the following version:

Theorem 2.1 (see e.g. (Ireland and Rosen, 1990, Chapter 3, §4))**.** *Let $n_1, n_2, \ldots, n_l \geq 2$ be pairwise coprime integers and let $N = \prod_{1 \leq i \leq l} n_i$. Then there is a ring isomorphism*

$$h : \prod_{1 \leq i \leq l} \mathbb{Z}_{n_i} \to \mathbb{Z}_N.$$

Moreover, given the binary representations of $n_1, \ldots, n_l \in \mathbb{N}$ and $x_i \in \mathbb{Z}_{n_i}$ for $1 \leq i \leq l$, one can compute in polynomial time the binary representation of $h(x_1, \ldots, x_l) \in \mathbb{Z}_N$.[iii]

We will only use the fact that h is an isomorphism between the additive abelian groups $\prod_{1 \leq i \leq l} \mathbb{Z}_{n_i}$ and $\mathbb{Z}_N$.

3 Parameterized complexity theory

An excellent introduction into parameterized complexity theory is the monograph Flum and Grohe (2006). A *parameterized problem* is a subset $L \subseteq \Sigma^* \times \mathbb{N}$ for a finite alphabet Σ. For a pair $(u, k) \in \Sigma^* \times \mathbb{N}$ we call k the parameter. For a fixed number $d \in \mathbb{N}$, the d^{th} slice of L is $L_d = \{(u, d) \in L : u \in \Sigma^*\}$. The class FPT of *fixed parameter tractable problems* is the class of all parameterized problems $L \subseteq \Sigma^* \times \mathbb{N}$ for which there is an algorithm deciding L and running on input (u, k) in time $poly(|u|) \cdot f(k)$ for an arbitrary computable function $f : \mathbb{N} \to \mathbb{N}$. The class XP is the class of parameterized problems that can be solved by a *deterministic* algorithm in time $|u|^{f(k)} + f(k)$ for an arbitrary computable function $f : \mathbb{N} \to \mathbb{N}$ (Flum and Grohe, 2006, Definition 2.22). The class para-NP is the class of parameterized problems that can be solved by a *nondeterministic* algorithm in time $poly(|u|) \cdot f(k)$ for an arbitrary computable function $f : \mathbb{N} \to \mathbb{N}$ (Flum and Grohe, 2006, Definition 2.10). The class W[P] is the class of parameterized problems that can be solved by a *nondeterministic* algorithm in time $poly(|u|) \cdot f(k)$ for an arbitrary computable function $f : \mathbb{N} \to \mathbb{N}$ with the restriction that at most $\log(|u|) \cdot h(k)$ steps are

[ii] By Cayley's theorem, every finite group is isomorphic to some subgroup of S_n. When we talk about a permutation group G, we mean the abstract group G together with its action on the elements of $[1, n]$.

[iii] This follows from the standard proof of the Chinese remainder theorem; see (Ireland and Rosen, 1990, proof of Theorem 1 on p. 34).

allowed to be nondeterministic ones for an arbitrary computable function $h : \mathbb{N} \to \mathbb{N}$ (Flum and Grohe, 2006, Definition 3.1). By (Flum and Grohe, 2006, Proposition 3.2) we have

$$\mathsf{FPT} \subseteq \mathsf{W[P]} \subseteq \mathsf{XP} \cap \mathsf{para\text{-}NP}.$$

An *fpt-reduction* from a parameterized problem $L_1 \subseteq \Sigma_1^* \times \mathbb{N}$ to a parameterized problem $L_2 \subseteq \Sigma_2^* \times \mathbb{N}$ is a computable function $r : \Sigma_1^* \times \mathbb{N} \to \Sigma_2^* \times \mathbb{N}$ such that the following hold for all $u \in \Sigma_1^*$ and $k \in \mathbb{N}$:

- there are functions g, h such that $r(u,k) = (g(u,k), h(k))$,
- $r(u,k)$ can be computed in time $poly(|u|)f(k)$ for a computable function $f : \mathbb{N} \to \mathbb{N}$,
- $(u,k) \in L_1$ if and only if $r(u,k) \in L_2$.

We write $L_1 \leq^{\mathsf{fpt}} L_2$ if there is an *fpt-reduction* from L_1 to L_2. We write $L_1 \equiv^{\mathsf{fpt}} L_2$ for $L_1 \leq^{\mathsf{fpt}} L_2 \leq^{\mathsf{fpt}} L_1$. Moreover, for the parameterized problem L we define its fpt-closure $[L]^{\mathsf{fpt}}$ as the class of all parameterized problems K with $K \leq^{\mathsf{fpt}} L$.

In Guillemot (2011) the class WNL is defined as $[\mathsf{NTMC}]^{\mathsf{fpt}}$, where NTMC (for *nondeterministic Turing machine computation*) is the following parameterized problem:

Problem 3.1 (NTMC)**.**
Input: a nondeterministic Turing machine M, a unary encoded $q \in \mathbb{N}$ and $k \in \mathbb{N}$ (the parameter)
Question: Does M accept the empty string in q steps by examining at most k cells?

Of particular importance in parameterized complexity theory are model-checking problems for fragments of first-order logic. We rely on the definition of a structure given in Flum and Grohe (2006). We fix a countably infinite set of *variables*. A *signature* σ is a finite set of *relational symbols* and *function symbols*, where every relation symbol $r \in \sigma$ has an *arity* $\alpha(r) \in \mathbb{N} \setminus \{0\}$ and every function symbol $f \in \mathcal{R}$ has an arity $\alpha(f) \in \mathbb{N}$. Function symbols of arity 0 are also called *constant symbols*. A *σ-structure* (or just structure, if the signature is not important) is a tuple $\mathcal{A} = (A, (s^{\mathcal{A}})_{s \in \sigma})$, where A is a non-empty set (the universe of $\mathcal{A}$). For every relation symbol $r \in \sigma$, $r^{\mathcal{A}} \subseteq A^{\alpha(r)}$ is an $\alpha(r)$-ary relation on the universe. For every function symbol $f \in \sigma$, $f^{\sigma} : A^{\alpha(f)} \to A$ is an $\alpha(f)$-ary function on A. A *relational signature* only contains relational symbols and a *relational structure* is a σ-structure for a relational signature σ.

Terms over the signature σ are built in the usual way: every variable and constant symbol is a term and if $t_1, \ldots, t_n$ are terms and f is a function symbol with $\alpha(f) = n$ then also $f(t_1, \ldots, t_n)$ is a term. The size $|t|$ of a term t is inductively defined by $|x| = |a| = 1$ for a variable x and a constant symbol a and $|f(t_1, \ldots, t_n)| = 1 + \sum_{i=1}^{n} |t_i|$. Note that if the signature σ is relational then the only terms are variables.

First-order formulas over the signature σ are built up from atomic formulas of the form $r(t_1, \ldots, t_n)$ (where $r \in \sigma$ is a relational symbol, $\alpha(r) = n$, and $t_1, \ldots, t_n$ are terms) and $t_1 = t_2$ (for terms t_1, t_2) using boolean operators ($\neg$, $\wedge$ and $\vee$) and quantifiers ($\exists x$ and $\forall x$ for a variable x). A *relational first-order formula* is a first-order formula F such that every term that appears in F is a variable. A (first-order) *sentence* is a first-order formula without free variables, i.e., every occurrence of a variable x that appears in an atomic formula is within the scope of a quantifier $\exists x$ or $\forall x$. For a sentence F, we write $\mathcal{A} \models F$ if the formula F is true (in the usual sense) in the structure $\mathcal{A}$. For a first-order formula F we define the length $|F|$ of the formula inductively:

- $|t_1 = t_2| = 1 + |t_1| + |t_2|$ for terms t_1, t_2,

- $|r(t_1, \ldots, t_n) = 1 + \sum_{i=1}^{n} |t_i|$ for terms $t_1, \ldots, t_n$ and a relational symbol r,
- $|F \wedge G| = |F \vee G| = |F| + |G| + 1$,
- $|\neg F| = |\exists x F| = |\forall x F| = |F| + 1$.

For a class of first-order sentences C we define the parameterized problem $\mathsf{pMC}(\mathsf{C})$ as follows:

Problem 3.2 (parameterized model-checking problem for C, $\mathsf{pMC}(\mathsf{C})$ for short)**.**
Input: a sentence $F \in \mathsf{C}$ and a structure $\mathcal{A}$
Parameter: the length $|F|$ of the formula F
Question: Does $\mathcal{A} \models F$ hold?

We now define several well-known parameterized complexity classes by taking for C certain fragments of first-order logic. For $l \geq 1$ we define Σ_l as the class of relational first-order sentences of the form $\exists \overline{x}_1 \forall \overline{x}_2 \cdots Q_l \overline{x}_l \psi$, where ψ is a relational quantifier-free formula, $Q_l = \exists$ if l is odd, $Q_l = \forall$ if l is even, and every $\overline{x}_i$ is a tuple of variables of arbitrary length. If we require in addition that every tuple $\overline{x}_i$ with $2 \leq i \leq l$ has at most u variables, we obtain the class $\Sigma_{l,u}$. If we allow arbitrary terms in these classes, then we obtain Σ_l^{func} and $\Sigma_{l,u}^{\mathsf{func}}$, respectively.

We then define the following parameterized complexity classes, where $l \geq 1$:

$$
\begin{aligned}
\mathsf{W}[l] &= \left[\mathsf{pMC}(\Sigma_{l,1})\right]^{\mathsf{fpt}} && \text{(Flum and Grohe, 2006, Theorem 7.22)} \\
\mathsf{A}[l] &= \left[\mathsf{pMC}(\Sigma_l)\right]^{\mathsf{fpt}} && \text{(Flum and Grohe, 2006, Definition 5.7)} \\
\mathsf{W}^{\mathsf{func}}[l] &= \left[\mathsf{pMC}(\Sigma_{l,1}^{\mathsf{func}})\right]^{\mathsf{fpt}} && \text{(Chen et al., 2005, Definition 28)} \\
\mathsf{A}^{\mathsf{func}}[l] &= \left[\mathsf{pMC}(\Sigma_l^{\mathsf{func}})\right]^{\mathsf{fpt}}.
\end{aligned}
\tag{1}
$$

The following facts are known about these classes:

- $\mathsf{W}[1] = \mathsf{A}[1]$ (this follows directly from our definition).
- $\mathsf{W}[l] = \left[\mathsf{pMC}(\Sigma_{l,u})\right]^{\mathsf{fpt}}$ (Flum and Grohe, 2006, Theorem 7.22) and $\mathsf{W}^{\mathsf{func}}[l] = \left[\mathsf{pMC}(\Sigma_{l,u}^{\mathsf{func}})\right]^{\mathsf{fpt}}$ (Chen et al., 2005, Theorem 29) for every $u \geq 1$
- $\mathsf{W}[l] \subseteq \mathsf{W}^{\mathsf{func}}[l] \subseteq \mathsf{A}^{\mathsf{func}}[l] = \mathsf{A}[l]$ for all $l \geq 1$ (only the last equality is nontrivial; it is stated in (Flum and Grohe, 2006, p. 201)
- $\bigcup_{l \geq 1} \mathsf{W}[l] \subseteq \mathsf{WNL} \subseteq \mathsf{XP} \cap \mathsf{para\text{-}NP}$ (Fernau and Bruchertseifer, 2022, Theorem 1)
- $\bigcup_{l \geq 1} \mathsf{W}^{\mathsf{func}}[l] \subseteq \mathsf{W}[\mathsf{P}]$ (Flum and Grohe, 2006, Exercise 8.58)

Finally, let us remark that if in the problem NTMC (Problem 3.1) one takes q instead of k as the parameter, then the fpt-closure of the resulting problem is $\mathsf{W}[1]$; see Guillemot (2011).

4 Parameterized complexity of factorization problems

4.1 Generic variants of factorization problems

In the following the parameter will be always the natural number k. It therefore makes no difference whether the parameter k is given in binary or unary representation, since it enters the running time with an arbitrary function $f(k)$.

Let C be a class of monoids. Every monoid $M \in \mathsf{C}$ must have a finite representation that we define later for the classes considered in this paper. Also elements of M must have a finite representation. The parameterized factorization problem for the class C is the following problem:

Problem 4.1 (parameterized factorization problem for the class C, $\mathsf{pF}[\mathsf{C}]$ for short)**.**
Input: a monoid $M \in \mathsf{C}$, elements $a, a_1, \ldots, a_m \in M$ and $k \in \mathbb{N}$ (the parameter)
Question: Is $a \in \{a_1, \ldots, a_m\}^k$?

We also consider two variants of $\mathsf{pF}[\mathsf{C}]$ by restricting the order in which the elements $a_1, \ldots, a_m$ may occur.

Problem 4.2 (parameterized knapsack for C, $\mathsf{pKS}[\mathsf{C}]$ for short)**.**
Input: a monoid $M \in \mathsf{C}$, elements $a, a_1, \ldots, a_m \in M$ and $k \in \mathbb{N}$ (the parameter)
Question: Are there $x_1, \ldots, x_m \in \mathbb{N}$ with $\sum_{i=1}^m x_i = k$ and $a = a_1^{x_1} \cdots a_m^{x_m}$?

Problem 4.3 (parameterized subset sum for C, $\mathsf{pSSS}[\mathsf{C}]$ for short)**.**
Input: a monoid $M \in \mathsf{C}$, elements $a, a_1, \ldots, a_m \in M$ and $k \in \mathbb{N}$ (the parameter)
Question: Are there $x_1, \ldots, x_m \in \{0, 1\}$ with $\sum_{i=1}^m x_i = k$ and $a = a_1^{x_1} \cdots a_m^{x_m}$?

Note that when C only contains commutative monoids, the problems $\mathsf{pF}[\mathsf{C}]$ and $\mathsf{pKS}[\mathsf{C}]$ are obviously equivalent with respect to fpt-reductions. In this case, we only consider the problems $\mathsf{pKS}[\mathsf{C}]$ and $\mathsf{pSSS}[\mathsf{C}]$. If $\mathsf{C} = \{M\}$ consists of only one monoid, we write $\mathsf{pSSS}[M]$ instead of $\mathsf{pSSS}[\{M\}]$ and analogously for the other problems.

In this paper, the focus is on permutation groups and abelian groups.

Definition 4.4. We consider the following cases for the class C:

- $\mathsf{T}_* := \{T_n : n \geq 1\}$ is the class of all transformation monoids. The transformation monoid T_n is simply represented by the number n in *unary* representation and a function $f \in T_n$ is represented by the list $f(1), f(2), \ldots, f(n)$. The problem $\mathsf{pF}[\mathsf{T}_*]$ has been considered in Cai et al. (1997).
- $\mathsf{S}_* := \{S_n : n \geq 1\}$ is the class of all symmetric groups. For this case, we inherit the above input representation for T_*.
- AbPG is the class of abelian permutation groups. An abelian permutation group is represented by a list of pairwise commuting generators $\pi_1, \ldots, \pi_m \in S_n$ (for some n). Elements of $\langle \pi_1, \ldots, \pi_m \rangle$ are simply represented by permutations (by Furst et al. (1980) one can check in polynomial time, whether a given permutation belongs to $\langle \pi_1, \ldots, \pi_m \rangle$).
- CycPG is the class of cyclic permutation groups. We use the same input representation as for AbPG. In Appendix A we will show that given a list of permutations $\pi_1, \ldots, \pi_m$ one can check in polynomial time, whether the permutation group $\langle \pi_1, \ldots, \pi_m \rangle$ is cyclic. In the positive case one can compute in polynomial time a permutation π with $\langle \pi \rangle = \langle \pi_1, \ldots, \pi_m \rangle$.

- the singleton classes $\{\mathbb{Z}\}$ and $\{\mathbb{N}\}$: in these cases, there is of course no reason to choose an input representation for the monoid. Elements of $\mathbb{Z}$ or $\mathbb{N}$ are represented in binary representation.
- $\mathbb{Z}^* := \{\mathbb{Z}^n : n \geq 1\}$ and $\mathbb{N}^* := \{\mathbb{N}^n : n \geq 1\}$, where $\mathbb{Z}^n$ (respectively, $\mathbb{N}^n$) is represented by the unary encoding of n and group elements are represented by n-tuples of binary encodings.
- FAbG is the class of finite abelian groups, where the finite abelian group $G = \prod_{i=1}^{d} \mathbb{Z}_{n_i}$ is represented by the tuple $(n_1, n_2, \ldots, n_d)$ with each n_i given in binary representation. An element of G is given by a tuple $(a_1, a_2, \ldots, a_d)$ with each $a_i \in [0, n_i - 1]$ given in binary representation.
- $\mathsf{FCycG} := \{\mathbb{Z}_n : n \geq 2\}$ is the class of finite cyclic groups (with n again given in binary notation).

The classes except for $\mathsf{C} = \mathsf{T}_*$ and $\mathsf{C} = \mathsf{S}_*$ are called the *commutative classes* in the rest of the paper. For those classes we do not consider $\mathsf{pF}[\mathsf{C}]$ since it is equivalent to $\mathsf{pKS}[\mathsf{C}]$ with respect to fpt-reductions.

Remark 4.5. *Note that as abstract classes of groups,* CycPG *and* FCycG *(respectively,* AbPG *and* FAbG*) are the same, but there is a difference in the succinctness of descriptions. In general, if $\gamma_1, \ldots, \gamma_d$ are the pairwise disjoint cycles of a permutation $\pi \in S_n$, and $\ell_i \in [1, n]$ is the length of γ_i, then $\mathrm{ord}(\pi) = \mathrm{lcm}(\ell_1, \ldots, \ell_d)$. In particular, every prime power p^e that divides $\mathrm{ord}(\pi)$ is bounded by n. Therefore the cyclic permutation group $\langle\pi\rangle \leq S_n$ is isomorphic to $\mathbb{Z}_m$, where all prime powers in m are bounded by n (recall that the latter is given in unary representation in our setting). Hence, if for instance, $\mathbb{Z}_{2^m} \cong \langle\pi\rangle$ for $\pi \in S_n$, then $n \geq 2^m$ and the encoding of π (a list of n numbers from $[1, n]$) needs at least $2^m \cdot m$ bits.*

Remark 4.6. *For an abelian permutation group $\langle\pi_1, \ldots, \pi_m\rangle$ we represent an element $\pi \in \langle\pi_1, \ldots, \pi_m\rangle$ as a permutation. Alternatively, we could represent π by binary encoded exponents $k_1, \ldots, k_m \in \mathbb{N}$ such that $\pi = \pi_1^{k_1} \cdots \pi_m^{k_m}$. These two representations are equivalent in the sense that one can change in polynomial time between the two representations. One direction is clear: given binary encoded numbers $k_1, \ldots, k_m$ one can compute by iterated squaring the permutation $\pi_1^{k_1} \cdots \pi_m^{k_m}$. On the other hand, given a permutation $\pi \in \langle\pi_1, \ldots, \pi_m\rangle$, the algorithm from Furst et al. (1980) allows to compute in polynomial time a straight-line program over $\pi_1, \ldots, \pi_m$ for π. This is a context-free grammar that produces a single word w over the letters $\pi_1, \ldots, \pi_m$ that evaluates to π. One can then easily compute in polynomial time the number of occurrences of every π_i in w.*

In some papers, one finds the alternative of $\mathsf{pF}[\mathsf{C}]$, where the condition $a \in \{a_1, \ldots, a_m\}^k$ is replaced by $a \in \{a_1, \ldots, a_m\}^{\leq k}$. Similarly, in $\mathsf{pKS}[\mathsf{C}]$ and $\mathsf{pSSS}[\mathsf{C}]$ one could replace the condition $\sum_{i=1}^{m} x_i = k$ by $\sum_{i=1}^{m} x_i \leq k$. Let us denote these problems with $\mathsf{pF}_{\leq}[\mathsf{C}]$, $\mathsf{pKS}_{\leq}[\mathsf{C}]$, and $\mathsf{pSSS}_{\leq}[\mathsf{C}]$, respectively.

For the subset sum problems one also finds the variants, where the a_i are assumed to be pairwise different. These variants will be denoted with $\mathsf{pSSS}^{\neq}[\mathsf{C}]$ and $\mathsf{pSSS}^{\neq}_{\leq}[\mathsf{C}]$. In the case where C consists of commutative monoids, one can define these problems also as follows: given a monoid $M \in \mathsf{C}$, a finite subset $A \subseteq M$, an element $a \in M$ and the parameter k, the question is whether there is a subset $S \subseteq A$ with $|S| = k$ (respectively, $|S| \leq k$) and $\sum_{b \in A} b = a$. The problem $\mathsf{pSSS}^{\neq}[\mathbb{Z}]$ was studied in Buss and Islam (2007), whereas $\mathsf{pSSS}[\mathbb{Z}]$ is the variant studied in Downey and Fellows (1995). We will later show that the above variants of subset sum are all equivalent with respect to fpt-reductions for the classes C from Definition 4.4. By replicating group elements in the input list one easily shows the following:

Lemma 4.7. *For every class* C *we have* $\mathsf{pF}[\mathsf{C}] \leq^{\mathsf{fpt}} \mathsf{pKS}[\mathsf{C}] \leq^{\mathsf{fpt}} \mathsf{pSSS}[\mathsf{C}]$ *and* $\mathsf{pF}_{\leq}[\mathsf{C}] \leq^{\mathsf{fpt}} \mathsf{pKS}_{\leq}[\mathsf{C}] \leq^{\mathsf{fpt}} \mathsf{pSSS}_{\leq}[\mathsf{C}]$.

Proof: We only show $\mathsf{pF}[\mathsf{C}] \leq^{\mathsf{fpt}} \mathsf{pKS}[\mathsf{C}] \leq^{\mathsf{fpt}} \mathsf{pSSS}[\mathsf{C}]$; the second statement can be shown in the same way. For $\mathsf{pF}[\mathsf{C}] \leq^{\mathsf{fpt}} \mathsf{pKS}[\mathsf{C}]$ consider input data $M \in \mathsf{C}$, $a, a_1, \ldots, a_m \in M$ and $k \in \mathbb{N}$ for $\mathsf{pF}[\mathsf{C}]$. We clearly have $a \in \{a_1, \ldots, a_m\}^k$ if and only if there exist $x_{i,j} \in \mathbb{N}$ ($i \in [1,k]$, $j \in [1,m]$) such that

$$a = \prod_{i=1}^{k} \prod_{j=1}^{m} a_j^{x_{i,j}} \text{ and } \sum_{i=1}^{k} \sum_{j=1}^{m} x_{i,j} = k. \tag{2}$$

To show that $\mathsf{pKS}[\mathsf{C}] \leq^{\mathsf{fpt}} \mathsf{pSSS}[\mathsf{C}]$ note that there exist $x_1, \ldots, x_m \in \mathbb{N}$ such that

$$a = a_1^{x_1} a_2^{x_2} \cdots a_m^{x_m} \text{ and } \sum_{i=1}^{m} x_i = k$$

if and only if there exist $x_{i,j} \in \{0,1\}$ ($i \in [1,m]$, $j \in [1,k]$) such that

$$a = a_1^{x_{1,1}} \cdots a_1^{x_{1,k}} a_2^{x_{2,1}} \cdots a_2^{x_{2,k}} \cdots a_m^{x_{m,1}} \cdots a_m^{x_{m,k}} \text{ and } \sum_{i=1}^{m} \sum_{j=1}^{k} x_{i,j} = k.$$

□

4.2 Factorization in transformation monoids

In Cai et al. (1997) it was shown that $\mathsf{pF}[\mathsf{T}_*]$ is $\mathsf{W}[2]$-hard. The complexity of $\mathsf{pF}[\mathsf{T}_*]$ was related in Fernau and Bruchertseifer (2022) to another parameterized problem. Consider a deterministic finite automaton $A = (Q, \Sigma, q_0, \delta, F)$ (DFA for short), where Q is the finite set of states, Σ is the finite input alphabet, $q_0 \in Q$ is the initial state, $F \subseteq Q$ is the set of final states and $\delta : Q \times \Sigma \to Q$ is the transition function. A *synchronizing word* $w \in \Sigma^*$ for A has the property that there exists a state $q \in Q$ such that for every state $p \in Q$ we have $\hat{\delta}(p, w) = q$, where $\hat{\delta} : Q \times \Sigma^* \to Q$ is the extension of δ to words (such a synchronizing word does not necessarily exist). The following problem has been introduced in Fernau et al. (2015); see also Fernau and Bruchertseifer (2022).

Problem 4.8 (p-DFA-SW Fernau and Bruchertseifer (2022))**.**
Input: DFA A, $k \in \mathbb{N}$ (the parameter)
Question: Is there a synchronizing word w for A with $|w| \leq k$?

It is known that $\mathsf{p\text{-}DFA\text{-}SW} \equiv^{\mathsf{fpt}} \mathsf{pF}[\mathsf{T}_*]$ Fernau and Bruchertseifer (2022) and that p-DFA-SW belongs to the intersection of the classes $\mathsf{A}[2]$, $\mathsf{W}[\mathsf{P}]$, and WNL Fernau et al. (2015). The complexity class $\mathsf{W}[\text{SYNC}]$ was defined in Fernau and Bruchertseifer (2022) as $\mathsf{W}[\text{SYNC}] = [\mathsf{p\text{-}DFA\text{-}SW}]^{\mathsf{fpt}}$. Therefore, $\mathsf{pF}[\mathsf{T}_*]$ is $\mathsf{W}[\text{SYNC}]$-complete and belongs to the intersection of the classes $\mathsf{A}[2]$, $\mathsf{W}[\mathsf{P}]$, and WNL. Note that $\mathsf{W}^{\mathsf{func}}[2] \subseteq \mathsf{A}^{\mathsf{func}}[2] = \mathsf{A}[2]$. Therefore, the following result improves the membership of $\mathsf{pF}[\mathsf{T}_*]$ in $\mathsf{A}[2]$ (and makes the statement $\mathsf{pF}[\mathsf{T}_*] \in \mathsf{W}[\mathsf{P}]$ obsolete since $\mathsf{W}^{\mathsf{func}}[2] \subseteq \mathsf{W}[\mathsf{P}]$).

Theorem 4.9. $\mathsf{pF}[\mathsf{T}_*]$ *belongs to* $\mathsf{W}^{\mathsf{func}}[2]$.

Proof: We present an fpt-reduction from $\mathsf{pF}[\mathsf{T}_*]$ to $\mathsf{pMC}(\Sigma_{2,1}^{\mathsf{func}})$. Consider an input instance (f, B, k) for $\mathsf{pF}[\mathsf{T}_*]$, where $f \in T_n$, $B \subseteq T_n$ and $k \in \mathbb{N}$. W.l.o.g. we can assume that B contains the identity function id with $\mathrm{id}(a) = a$ for all $a \in [1,n]$. Then, (f, B, k) is a positive instance of $\mathsf{pF}[\mathsf{T}_*]$ if and only if there are

$f_1, \ldots, f_k \in B$ such that $af = af_1 \cdots f_k$ for all $a \in [1, n]$. Let $\sigma = \{P, Q, h\}$ be a signature in which P is a unary relation symbol, Q is a binary relation symbol, and h is a binary function symbol. We define the σ-structure $\mathcal{A}$ as follows:

- $A = [1, n] \uplus B$ is the universe,
- $P^{\mathcal{A}} = B$,
- $Q^{\mathcal{A}} = \{(a, af) : a \in [1, n]\} \cup (B \times B)$ and
- the interpretation of the function symbol h is defined as follows, where $g \in B$ is arbitrary:

$$h^{\mathcal{A}}(a, p) = \begin{cases} ap & \text{if } a \in [1, n] \text{ and } p \in B, \\ g & \text{if } a \in B \text{ or } p \in [1, n]. \end{cases}$$

Let φ_k be the following sentence:

$$\varphi_k = \exists x_1 \cdots \exists x_k \bigwedge_{i=1}^{k} P(x_i) \wedge \forall z \; Q(z, h(\ldots h(h(z, x_1), x_2), \ldots, x_k)).$$

Then, (f, B, k) is a positive instance if and only if $\mathcal{A} \models \varphi_k$. □

Since $\mathsf{pF}[\mathsf{T}_*]$ is $\mathsf{W}[\text{SYNC}]$-complete Fernau and Bruchertseifer (2022), we obtain:

Corollary 4.10. $\mathsf{W}[\text{SYNC}] \subseteq \mathsf{W}^{\mathsf{func}}[2]$.

4.3 Factorization in symmetric groups

We now show that the various variants of factorization problems for symmetric groups are equivalent with respect to fpt-reductions:

Theorem 4.11. $\mathsf{pF}[\mathsf{S}_*] \equiv^{\mathsf{fpt}} \mathsf{pKS}[\mathsf{S}_*] \equiv^{\mathsf{fpt}} \mathsf{pSSS}[\mathsf{S}_*]$

Proof: By Lemma 4.7 it suffices to show $\mathsf{pSSS}[\mathsf{S}_*] \leq^{\mathsf{fpt}} \mathsf{pF}[\mathsf{S}_*]$. Let $\pi, \pi_1, \ldots, \pi_m \in S_n$ and $k \in \mathbb{N}$ be the input data for $\mathsf{pSSS}[\mathsf{S}_*]$. W.l.o.g. we can assume that $k \geq 4$. For $i < j$ we use $[\![i, j]\!]$ to denote the cycle $(i, i+1 \ldots, j)$. Let $n_0 = n + k + 3$ and $\beta = [\![n+1, n_0 - 1]\!]$, which is a cycle of length $k + 2$. For $l \in [1, m+1]$ we define the cycle $\gamma_l = [\![n_0 + (l-1)k, n_0 + lk]\!]$. Note that consecutive cycles γ_l and γ_{l+1} intersect in the point $n_0 + lk$.

We define an instance of $\mathsf{pF}[\mathsf{S}_*]$. The group generators are

$$\tau_{i,j} \;=\; \pi_j \beta \prod_{l=i}^{j} \gamma_l \quad \text{for } j \in [1, m], i \in [1, j] \text{ and} \tag{3}$$

$$\tau_{i,m+1} \;=\; \beta \prod_{l=i}^{m+1} \gamma_l \quad \text{for } i \in [1, m+1]. \tag{4}$$

Finally we define the permutation τ to be factored by

$$\tau = \pi\beta^{k+1}\prod_{l=1}^{m+1}\gamma_l. \tag{5}$$

For the following, it is important that

- $\pi, \pi_1, \ldots, \pi_m$ act trivially on all points that are not in $[1, n]$,
- β acts trivially on all points that are not in $[n+1, n_0-1]$, and
- $\gamma_1, \ldots, \gamma_{m+1}$ act trivially on all points that are not in $[n_0, n_0+(m+1)k]$.

Moreover, the three intervals $[1, n]$, $[n+1, n_0-1]$ and $[n_0, n_0+(m+1)k]$ are pairwise disjoint.

We now show that there exist $x_1, \ldots, x_m \in \{0,1\}$ with $\sum_{i=1}^m x_i = k$ and $\pi = \pi_1^{x_1}\cdots\pi_m^{x_m}$ if and only if $\tau \in \{\tau_{i,j} : 1 \le i \le j \le m+1\}^{\le k+1}$. We start with the easier direction from left to right.

Suppose that there exist $x_1, \ldots, x_m \in \{0,1\}$ with $\sum_{i=1}^m x_i = k$ and $\pi = \pi_1^{x_1}\cdots\pi_m^{x_m}$. In other words, we can write $\pi = \pi_{i_1}\cdots\pi_{i_k}$ with strictly increasing indices $1 \le i_1 < \cdots < i_k \le m$. Note that since the indices are strictly increasing we have $i_d \ge i_{d-1}+1$ for $2 \le d \le k$ and $m+1 \ge i_k+1$. Thus, the permutations $\tau_{i_{d-1}+1,i_d}$ and $\tau_{i_k+1,m+1}$ are defined. We obtain the following (below, the reader finds justifications for the equalities):

$$\begin{aligned}
\tau_{1,i_1}\cdot\left(\prod_{d=2}^{k}\tau_{i_{d-1}+1,i_d}\right)\cdot\tau_{i_k+1,m+1} &\overset{\text{(a)}}{=} \pi_{i_1}\beta\prod_{l=1}^{i_1}\gamma_l\cdot\prod_{d=2}^{k}\left(\pi_{i_d}\beta\prod_{l=i_{d-1}+1}^{i_d}\gamma_l\right)\cdot\beta\prod_{l=i_k+1}^{m+1}\gamma_l \\
&\overset{\text{(b)}}{=} \pi_{i_1}\cdots\pi_{i_k}\beta^{k+1}\cdot\prod_{l=1}^{i_1}\gamma_l\cdot\prod_{d=2}^{k}\prod_{l=i_{d-1}+1}^{i_d}\gamma_l\cdot\prod_{l=i_k+1}^{m+1}\gamma_l \\
&\overset{\text{(c)}}{=} \pi_{i_1}\cdots\pi_{i_k}\beta^{k+1}\prod_{l=1}^{m+1}\gamma_l \overset{\text{(d)}}{=} \tau.
\end{aligned}$$

Equality (a) follows directly from the definition of the $\pi_{i,j}$ in (3) and (4). For equality (b) note that for all $i, a \in [1, m]$ and $b \in [a+1, m+1]$ the three permutations π_i, β and $\prod_{l=a}^{b}\gamma_l$ pairwise commute since these permutations act non-trivially on pairwise disjoint domains. Equality (c) is trivial and (d) finally follows from (5). By the above calculation we obtain $\tau \in \{\tau_{i,j} : 1 \le i \le j \le m+1\}^{\le k+1}$.

Now suppose $\tau \in \{\tau_{i,j} : 1 \le i \le j \le m+1\}^{\le k+1}$. Then τ is generated by a sequence of $h \le k+1$ permutations. Hence there are numbers $j_1, \ldots, j_h \in [1, m+1]$ and for every $d \in [1, h]$ numbers $i_d \in [1, j_d]$ such that

$$\tau = \prod_{d=1}^{h}\tau_{i_d,j_d}. \tag{6}$$

By projecting onto the domain $[n+1, n_0-1]$ of β, this implies $\beta^{k+1} = \beta^h$ (note that every $\tau_{i,j}$ contains exactly one copy of β and τ contains β^{k+1}). Since β is a cycle of length $k+2$ we obtain $h = k+1$. Moreover, projecting (6) onto the union of the domains of the γ_l (which is $[n_0, n_0+(m+1)k]$) yields

$$\prod_{l=1}^{m+1}\gamma_l = \prod_{d=1}^{k+1}\prod_{l=i_d}^{j_d}\gamma_l. \tag{7}$$

This equation implies two crucial constraints, namely that every cycle γ_l ($l \in [1, m+1]$) occurs in the right-hand side of (7) exactly once (Claim 4.12) and the unique occurrence of the cycle γ_r appears to the left of the unique occurrence of the cycle γ_s whenever $r < s$ (Claim 4.13). From these constraints we can then deduce that the projection of the factorization (6) onto the domain $[1, n]$ of the permutations $\pi, \pi_1, \ldots, \pi_m$ yields an $\mathsf{pSSS}[\mathsf{S}_*]$-solution for the input instance $\pi, \pi_1, \ldots, \pi_m$.

Claim 4.12. *If equation* (7) *holds then for all $l \in [1, m+1]$ the cycle γ_l occurs in the right-hand side exactly once.*

Proof of Claim 4.12. If a cycle γ_l does not appear in the right-hand side of (7), then the point $n_0 + (l-1)k + 1$ is not moved by the right-hand side of (7) but it is moved by the left-hand side (recall that $k \geq 4$). Hence, every cycle γ_l ($l \in [1, m+1]$) must appear in the right-hand side.

Now suppose that a cycle γ_s for some $s \in [1, m+1]$ occurs more than once in the right-hand side. We have

$$(n_0 + (s-1)k + 1) \cdot \prod_{l=1}^{m+1} \gamma_l = n_0 + (s-1)k + 2.$$

In the following we consider the first two occurences of γ_s in the right-hand side of (7) and suppose that these occurrences are among the first $e \geq 2$ cycles γ_l and any further occurrence of γ_s comes after the first e cycles. Then we have

$$(n_0 + (s-1)k + 1) \cdot \prod_{d=1}^{k+1} \prod_{l=i_d}^{j_d} \gamma_l = (n_0 + (s-1)k + 3) \cdot \prod_{d=e+1}^{k+1} \prod_{l=i_d}^{j_d} \gamma_l.$$

Here we use the fact that every cycle γ_l has length $k+1$ and $k \geq 4$. From (7) we therefore obtain

$$(n_0 + (s-1)k + 3) \cdot \prod_{d=e+1}^{k+1} \prod_{l=i_d}^{j_d} \gamma_l = n_0 + (s-1)k + 2. \tag{8}$$

Since for every cycle γ_r with $r \neq s$ and every $z \in [1, k-1]$ we have

$$(n_0 + (s-1)k + z) \cdot \gamma_r = n_0 + (s-1)k + z,$$

equation (8) can only hold if the product $\prod_{d=e+1}^{k+1} \prod_{l=i_d}^{j_d} \gamma_l$ contains k further occurrences of γ_s (since γ_s is a cycle of length $k+1$, we have $(n_0+(s-1)k+3)\cdot\gamma_s^k = (n_0+(s-1)k+3)\cdot\gamma_s^{-1} = n_0+(s-1)k+2$). However, the product $\prod_{d=e+1}^{k+1} \prod_{l=i_d}^{j_d} \gamma_l$ contains at most $k+1-e \leq k-1$ further occurrences of γ_s. By this it follows

$$(n_0 + (s-1)k + 1) \cdot \prod_{d=1}^{k+1} \prod_{l=i_d}^{j_d} \gamma_l \neq n_0 + (s-1)k + 2,$$

which is a contradiction. This proves Claim 4.12. □

Claim 4.13. *If equation* (7) *holds and $r < s$ then γ_r appears before γ_s in the right-hand side of* (7)*.*

Proof of Claim 4.13. By Claim 4.12 we can assume that every cycle γ_l is contained at the right-hand side exactly once. Assume that $r < s$ and γ_s appears before γ_r. We will deduce a contradiction. We can

assume that the difference $s - r$ is minimal in the sense that there is no y such that $r < y < s$ and γ_s appears before γ_y or γ_y appears before γ_r. In this case we must have $s = r + 1$. Otherwise we have $s - r \geq 2$. If we then consider the cycle γ_{r+1}, we obtain a contradiction to the minimality of $s - r$ in case γ_s appears before γ_{r+1} as well as in case γ_{r+1} appears before γ_s (since then also γ_{r+1} appears before γ_r).

Let $r_1, \ldots, r_{m+1} \in \{1, \ldots, m+1\}$ with $r_i \neq r_j$ for $i \neq j$ be the sequence of numbers such that

$$\prod_{d=1}^{k+1} \prod_{l=i_d}^{j_d} \gamma_l = \prod_{l=1}^{m+1} \gamma_{r_l}$$

(this should be seen as an identity between words over the letters γ_l). Thus, we have

$$\prod_{l=1}^{m+1} \gamma_{r_l} = \prod_{l=1}^{m+1} \gamma_l. \tag{9}$$

Since $s = r + 1$, there are $p < q$ such that $r_p = r + 1$ and $r_q = r$. We obtain the following (justifications for the equalities can be found below):

$$(n_0 + rk - 1) \cdot \prod_{l=1}^{m+1} \gamma_{r_l} \overset{\text{(a)}}{=} (n_0 + rk - 1) \cdot \prod_{l=q}^{m+1} \gamma_{r_l} \overset{\text{(b)}}{=} (n_0 + rk) \cdot \prod_{l=q+1}^{m+1} \gamma_{r_l} \overset{\text{(c)}}{=} n_0 + rk\,.$$

Equation (a) holds since the point $n_0 + rk - 1$ is not moved by the cycles γ_i with $i \neq r$. Since $\gamma_r = \gamma_{r_q}$ it follows that $n_0 + rk - 1$ is not moved by $\prod_{l=1}^{q-1} \gamma_{r_l}$ (recall that every γ_i appears exactly once in the product). Equation (b) holds since $(n_0 + rk - 1) \cdot \gamma_{r_q} = (n_0 + rk - 1) \cdot \gamma_r = n_0 + rk$. Finally, (c) holds since $n_0 + rk$ is only moved by $\gamma_r = \gamma_{r_q}$ and $\gamma_{r+1} = \gamma_{r_p}$ and, since $p < q$, these two cycles do not appear in the product $\prod_{l=q+1}^{m+1} \gamma_{r_l}$.

On the other hand, using similar arguments, we obtain

$$\begin{aligned}(n_0 + rk - 1) \cdot \prod_{l=1}^{m+1} \gamma_l &= (n_0 + rk - 1) \cdot \prod_{l=r}^{m+1} \gamma_l = (n_0 + rk) \cdot \prod_{l=r+1}^{m+1} \gamma_l \\ &= (n_0 + rk + 1) \cdot \prod_{l=r+2}^{m+1} \gamma_l = n_0 + rk + 1\,.\end{aligned}$$

This contradicts (9) and shows Claim 4.13. □

From Claim 4.12 it follows that in equation (7) the intervals $[i_d, j_d]$ for $d \in [1, k+1]$ form a partition of $[1, m+1]$. Moreover, by Claim 4.13 we must have $j_d < i_{d+1}$ for all $d \in [1, k]$. This implies $i_1 = 1$, $i_{d+1} = j_d + 1$ for all $d \in [1, k]$ and $j_{k+1} = m + 1$. Recall that every generator $\tau_{i,j}$ for $1 \leq i \leq j \leq m$ includes the permutation $\pi_j \in S_n$ acting on $[1, n]$. Hence, by projecting (6) onto $[1, n]$ we get $\pi = \pi_{j_1} \pi_{j_2} \cdots \pi_{j_k}$ with $j_1 < j_2 < \cdots < j_k$. □

4.4 Factorization in the commutative setting

4.4.1 Simple lemmas

In this section, we collect a few simple lemmas that will be needed for the proof of Theorem 4.20.

Clearly, the $\mathbb{N}$-variants of our factorization problems reduce to the corresponding $\mathbb{Z}$-variants. Vice versa, by adding a large enough element from $\mathbb{N}$ (respectively, $\mathbb{N}^*$) to all input elements, one gets:

Lemma 4.14. *For all* $\mathsf{X} \in \{\mathsf{KS}, \mathsf{SSS}, \mathsf{SSS}^{\neq}\}$ *we have* $\mathsf{pX}[\mathbb{Z}] \leq^{\mathsf{fpt}} \mathsf{pX}[\mathbb{N}]$ *and* $\mathsf{pX}[\mathbb{Z}^*] \leq^{\mathsf{fpt}} \mathsf{pX}[\mathbb{N}^*]$.

Proof: Consider input vectors $a, a_1, \ldots, a_m \in \mathbb{Z}^n$ and let $k \geq 1$ be the parameter. All integers are given in binary representation. We then compute a vector $b \in \mathbb{N}^n$ such that $a + kb, a_1 + b, \ldots, a_m + b \in \mathbb{N}^n$ and take the new input elements $a + kb, a_1 + b, \ldots, a_m + b$. □

The next lemma is shown by concatenating the binary representations of d integers $c_1, \ldots, c_d$ to a single binary representation with enough zeros inbetween in order to avoid unwanted carries.

Lemma 4.15. *For all* $\mathsf{X} \in \{\mathsf{KS}, \mathsf{KS}_{\leq}, \mathsf{SSS}, \mathsf{SSS}_{\leq}, \mathsf{SSS}^{\neq}, \mathsf{SSS}^{\neq}_{\leq}\}$ *we have* $\mathsf{pX}[\mathbb{N}^*] \leq^{\mathsf{fpt}} \mathsf{pX}[\mathbb{N}]$.

Proof: We only show $\mathsf{pSSS}^{\neq}[\mathbb{N}^*] \leq^{\mathsf{fpt}} \mathsf{pSSS}^{\neq}[\mathbb{N}]$, the reduction also works for the other cases. Let $A \subseteq \mathbb{N}^d$, $a \in \mathbb{N}^d$ and $k \geq 1$ be the input data for $\mathsf{pSSS}^{\neq}[\mathbb{N}^*]$. Every number is given in binary representation.

The idea is then to append the binary representations of d integers $c_1, \ldots, c_d$ into a single binary representation with enough zeros inbetween in order to avoid unwanted carries. Let e be the maximum entry in a vector from A and let $m = \#(ke)$. Note that $\#(ke)$ denotes the number of bits of ke. Consider a tuple $c = (c_1, \ldots, c_d) \in [0, e]^d$ and let $(b_{i,1} \cdots b_{i,m_i})$ be the binary representation of the number c_i with $b_{i,1}$ the least significant bit. Note that $b_{i,m_i} = 1$ and $m_i \leq m$. We then encode c by the number $\mathsf{code}(c)$ with the binary representation

$$(b_{1,1} \cdots b_{1,m_1} \underbrace{00 \cdots 0}_{m - m_1 \text{ many}} b_{2,1} \cdots b_{2,m_2} \underbrace{00 \cdots 0}_{m - m_2 \text{ many}} \cdots b_{d,1} \cdots b_{d,m_d}).$$

Let $\mathsf{code}(A) = \{\mathsf{code}(b) : b \in A\}$. Then, there is a subset $S \subseteq A$ of size k such that $\sum_{b \in S} b = a$ if and only if there is a subset $S \subseteq \mathsf{code}(A)$ of size k such that $\sum_{b \in S} b = \mathsf{code}(a)$. □

In Vaidyanathan (2019) it was shown that $\mathsf{pF}[\mathsf{S}_*]$ and $\mathsf{pF}_{\leq}[\mathsf{S}_*]$ are equivalent with respect to fpt-reductions. One direction generalizes easily to the following statement (simply add the neutral element 1 sufficiently many times to the input list):

Lemma 4.16. *For all classes* C *from Definition 4.4 and* $\mathsf{X} \in \{\mathsf{F}, \mathsf{KS}, \mathsf{SSS}\}$ *we have* $\mathsf{pX}_{\leq}[\mathsf{C}] \leq \mathsf{pX}[\mathsf{C}]$.

Proof: For $\mathsf{X} \in \{\mathsf{F}, \mathsf{KS}\}$ it suffices to add the neutral element $1 \in M$ to the input sequence $a_1, \ldots, a_m$. For $\mathsf{pSSS}_{\leq}[\mathsf{C}] \leq^{\mathsf{fpt}} \mathsf{pSSS}[\mathsf{C}]$, we have to add k copies of 1 to the input sequence. □

Lemma 4.17. *If* $\mathsf{X} \in \{\mathsf{F}, \mathsf{KS}, \mathsf{SSS}, \mathsf{SSS}^{\neq}\}$ *and* C *is one of the classes from Definition 4.4, then* $\mathsf{pX}[\mathsf{C}] \leq^{\mathsf{fpt}} \mathsf{pX}_{\leq}[\mathsf{C}]$.

Proof: Let us first explain the idea for $\mathsf{pF}[\mathsf{S}_*] \leq^{\mathsf{fpt}} \mathsf{pF}_{\leq}[\mathsf{S}_*]$ and consider the symmetric group S_n and the elements $\pi, \pi_1, \ldots, \pi_m \in S_n$. The idea is to work in the group $S_n \times \mathbb{Z}_{k+1} \leq S_{n+k+1}$. For this take for τ a cycle of length $k+1$ on the set $[n+1, n+k+1]$ and view $\pi, \pi_1, \ldots, \pi_m$ as elements of S_{n+k+1}. Then

we have $\pi \in \{\pi_1, \ldots, \pi_m\}^k$ if and only if $\pi\tau^k \in \{\pi_1\tau, \ldots, \pi_m\tau\}^{\leq k}$. The same idea works (with small modifications) for the classes T_*, CycPG, AbPG, FAbG, $\mathbb{Z}^*$, and $\mathbb{N}^*$.

For CycPG we have to take a cycle of length $p > \max(n, k)$ for a prime p instead of the cycle τ of length $k+1$. Since n is given in unary notation, such a prime can be found in polynomial time. Then, for every $\pi \in S_n$, the group $\langle\pi\rangle \times \langle\tau\rangle$ is again cyclic and the cycle τ has length at least $k+1$.

For $\mathsf{C} = \mathsf{FCycG}$ we explain the idea for $\mathsf{pKS}[\mathsf{FCycG}] \leq^{\mathsf{fpt}} \mathsf{pKS}_{\leq}[\mathsf{FCycG}]$. Let $z, z_1, \ldots, z_m \in \mathbb{Z}_n$ and $k \in \mathbb{N}$ be the input of $\mathsf{pKS}[\mathsf{FCycG}]$. We will work in the group $\mathbb{Z}_n \times \mathbb{Z}_{nk+1}$. There are numbers $x_1, \ldots, x_m \in \mathbb{N}$ such that

$$\sum_{i=1}^{m} x_i z_i \equiv z \bmod n \text{ with } \sum_{i=1}^{m} x_i = k$$

if and only if

$$\sum_{i=1}^{m} x_i (z_i, 1) = (z, k) \text{ with } \sum_{i=1}^{m} x_i \leq k$$

in which the computations in the first coordinate are modulo n and in the second modulo $(nk+1)$. Moreover, since n and $nk+1$ are coprime, we have $\mathbb{Z}_n \times \mathbb{Z}_{nk+1} \cong \mathbb{Z}_{n(nk+1)}$ and the latter is a cyclic group. An isomorphism $h : \mathbb{Z}_n \times \mathbb{Z}_{nk+1} \to \mathbb{Z}_{n(nk+1)}$ is obtained from Theorem 2.1, which also shows that the values $h(z_1, 1), \ldots, h(z_m, 1), h(z, k) \in \mathbb{Z}_{n(nk+1)}$ can be computed in polynomial time.

Analogously we obtain $\mathsf{pX}[\mathsf{FCycG}] \leq^{\mathsf{fpt}} \mathsf{pX}_{\leq}[\mathsf{FCycG}]$ for all $\mathsf{X} \in \{\mathsf{F}, \mathsf{SSS}, \mathsf{SSS}^{\neq}\}$. Finally, for $\mathbb{Z}$ we first reduce $\mathsf{pX}[\mathbb{Z}]$ to $\mathsf{pX}[\mathbb{N}]$ using Lemma 4.14. The latter is then reduced to $\mathsf{pX}_{\leq}[\mathbb{N} \times \mathbb{N}]$ using the above construction. Finally $\mathsf{pX}_{\leq}[\mathbb{N} \times \mathbb{N}]$ can be reduced to $\mathsf{pX}_{\leq}[\mathbb{N}]$ by Lemma 4.15. □

Lemma 4.18. *We have* $\mathsf{pSSS}[\mathsf{C}] \leq^{\mathsf{fpt}} \mathsf{pSSS}^{\neq}[\mathsf{C}]$ *for each of the classes from Definition 4.4 .*

Proof: We first show $\mathsf{pSSS}[\mathsf{S}_*] \leq^{\mathsf{fpt}} \mathsf{pSSS}^{\neq}[\mathsf{S}_*]$. Consider input data $\pi, \pi_1, \ldots, \pi_m \in S_n$ and k for $\mathsf{pSSS}[\mathsf{S}_*]$. We will work with the group $\mathbb{Z}_2^{2m+1} \times \mathbb{Z}_{2k+2} \times S_n \leq S_{4m+2k+n+4}$.

Consider $i \in [1, m]$. We replace the element π_i in the sequence $\pi_1, \ldots, \pi_m$ by the group element

$$\rho_i = (e_{2i}^{2m+1}, 1, \pi_i) \in \mathbb{Z}_2^{2m+1} \times \mathbb{Z}_{2k+2} \times S_n,$$

where $e_{2i}^{2m+1} \in \mathbb{Z}_2^{2m+1}$ is the $2m+1$ dimensional unit vector with a 1 at the $(2i)^{\text{th}}$ coordinate (all other entries are zero). Note that the ρ_i are pairwise different. We then add to the sequence all filling elements

$$(0, \ldots, 0, 1, \ldots, 1, 0, \ldots, 0, \mathsf{id}) \in \mathbb{Z}_2^{2m+1} \times \mathbb{Z}_{2k+2} \times S_n, \tag{10}$$

where the 1's form a contiguous non-empty block that starts at some position $2i+1$ and ends at some position $2j+1$ with $0 \leq i \leq j \leq m$. With id we denote the identity permutation. Let $\rho_{m+1}, \ldots, \rho_{m+\ell}$ be a list of all filling elements (the precise order does not matter, since the filling elements pairwise commute). Note that the elements in the list $\rho_1, \ldots, \rho_{m+\ell}$ are pairwise different.

Finally, let $\rho = (1, 1, \ldots, 1, k, \pi) \in \mathbb{Z}_2^{2m+1} \times \mathbb{Z}_{2k+2} \times S_n$. We claim that there is a subset $I \subseteq [1, m]$ such that $|I| = k$ and $\prod_{i \in I} \pi_i = \pi$ if and only if there is a subset $J \subseteq [1, m+\ell]$ with $|J| = 2k+1$ and $\prod_{j \in J} \rho_j = \rho$.[(iv)]

[(iv)] When writing $\prod_{i \in I} \pi_i$ we assume that the product goes over the set I in ascending order and similarly for other products.

First assume that there is $J \subseteq [1, m+\ell]$ with $|J| = 2k+1$ and $\prod_{j\in J} \rho_j = \rho$. The k in the second last position of ρ forces J to contain exactly k indices from $[1, m]$. Let $I = J \cap [1, m]$ so that $|I| = k$. Projecting on the last coordinate yields $\prod_{i\in I} \pi_i = \pi$.

For the other direction let $I \subseteq [1, m]$ with $|I| = k$ and $\prod_{i\in I} \pi_i = \pi$. Consider now the group element $\prod_{i\in I} \rho_i$. Its last entry is π and the second last entry is k. Moreover, the first $2m+1$ entries of $\prod_{i\in I} \rho_i$ contain exactly k ones (all other entries are zero) and two ones are separated by at least one zero. The zero gaps can be filled with exactly $k+1$ filling elements of the form (10). Hence, there is a subset $J \subseteq [m+1, m+\ell]$ such that $\prod_{i\in I\cup J} \rho_i = \rho$.

The same reduction can be used in a similar way for the other classes from Definition 4.4. For $\mathsf{C} = \mathsf{FCycG}$ assume that we have a $\mathsf{pSSS}[\mathsf{FCycG}]$-instance over the cyclic group $\mathbb{Z}_n$ with n given in binary notation. Let $p_1 < p_2 < \cdots < p_{2m+2}$ be the first $2m+2$ primes that are larger than $2k+1$ and do not divide n. Since n has at most $\log_2(n)$ different prime divisors we need to check at most the first $2k+1+\lfloor\log_2(n)\rfloor + 2m+2$ primes. Hence, we can find these primes in polynomial time. Then we can work in the group $\prod_{i=1}^{2m+2} \mathbb{Z}_{p_i} \times \mathbb{Z}_n \cong \mathbb{Z}_N$ for $N = n \cdot \prod_{i=1}^{2m+2} p_i$ which is again a cyclic group. Note that the binary encoding of N can be computed in polynomial time. Moreover we can compute the corresponding numbers in $\mathbb{Z}_N$ in polynomial time.

For $\mathsf{C} = \mathsf{CycPG}$ we replace the cyclic permutation group $G \leq S_n$ by $\prod_{i=1}^{2m+2} \mathbb{Z}_{p_i} \times G$, where $p_1 < p_2 < \cdots < p_{2m+2}$ are primes with $p_1 \geq \max\{n+1, 2k+2\}$. This ensures that $\prod_{i=1}^{2m+2} \mathbb{Z}_{p_i} \times G \leq S_{n'}$ is again a cyclic permutation group with $n' = \sum_{i=1}^{2m+2} p_i + n$.

Finally for $\mathsf{C} = \mathbb{Z}$ and $\mathsf{C} = \mathbb{N}$ we obtain with Lemmas 4.14 and 4.15 the following reduction: $\mathsf{pSSS}[\mathbb{N}] \leq^{\mathsf{fpt}} \mathsf{pSSS}[\mathbb{Z}] \leq^{\mathsf{fpt}} \mathsf{pSSS}[\mathbb{Z}^*] \leq^{\mathsf{fpt}} \mathsf{pSSS}^{\neq}[\mathbb{Z}^*] \leq^{\mathsf{fpt}} \mathsf{pSSS}^{\neq}[\mathbb{N}^*] \leq^{\mathsf{fpt}} \mathsf{pSSS}^{\neq}[\mathbb{N}] \leq^{\mathsf{fpt}} \mathsf{pSSS}^{\neq}[\mathbb{Z}]$. □

Corollary 4.19. *Let* C *be one of the classes from Definition 4.4. Then the problems* $\mathsf{pSSS}[\mathsf{C}]$, $\mathsf{pSSS}^{\neq}[\mathsf{C}]$, $\mathsf{pSSS}_{\leq}[\mathsf{C}]$, $\mathsf{pSSS}^{\neq}_{\leq}[\mathsf{C}]$ *are equivalent with respect to fpt-reductions.*

Proof: By Lemmas 4.16 and 4.17 we have

$$\mathsf{pSSS}[\mathsf{C}] \equiv^{\mathsf{fpt}} \mathsf{pSSS}_{\leq}[\mathsf{C}].$$

By Lemma 4.18 we have

$$\mathsf{pSSS}[\mathsf{C}] \equiv^{\mathsf{fpt}} \mathsf{pSSS}^{\neq}[\mathsf{C}]$$

($\mathsf{pSSS}^{\neq}[\mathsf{C}] \leq^{\mathsf{fpt}} \mathsf{pSSS}[\mathsf{C}]$ is trivial). Moreover, we have

$$\mathsf{pSSS}^{\neq}_{\leq}[\mathsf{C}] \leq^{\mathsf{fpt}} \mathsf{pSSS}_{\leq}[\mathsf{C}] \leq^{\mathsf{fpt}} \mathsf{pSSS}[\mathsf{C}] \leq^{\mathsf{fpt}} \mathsf{pSSS}^{\neq}[\mathsf{C}] \leq^{\mathsf{fpt}} \mathsf{pSSS}^{\neq}_{\leq}[\mathsf{C}],$$

where the final reduction follows from Lemma 4.17. □

For the commutative classes from Definition 4.4 with the exception of FAbG and AbPG, we can show the equivalence (with respect to fpt-reductions) for all our factorization problems:

Theorem 4.20. *All problems* $\mathsf{pX}[\mathsf{C}]$ *for*

- $\mathsf{X} \in \{\mathsf{KS}, \mathsf{KS}_{\leq}, \mathsf{SSS}, \mathsf{SSS}_{\leq}, \mathsf{SSS}^{\neq}, \mathsf{SSS}^{\neq}_{\leq}\}$ *and*
- $\mathsf{C} \in \{\mathsf{CycPG}, \mathsf{FCycG}, \mathbb{N}, \mathbb{Z}, \mathbb{N}^*, \mathbb{Z}^*\}$

are equivalent with respect to fpt-reductions. Moreover all these problems are $\mathsf{W}[1]$*-hard and contained in* $\mathsf{W}[3]$.

We obtain Theorem 4.20 from Theorem 4.26 that will be shown in the next section. The statements concerning $\mathsf{W}[1]$ and $\mathsf{W}[3]$ from Theorem 4.20 follow from Downey and Fellows (1995) (where $\mathsf{W}[1]$-hardness of $\mathsf{pSSS}[\mathbb{Z}]$ was shown) and Buss and Islam (2007) (where membership of $\mathsf{pSSS}^{\neq}[\mathbb{Z}]$ in $\mathsf{W}[3]$ was shown).

4.4.2 Commutative subset sum problems

Lemma 4.21. *For all* $\mathsf{X} \in \{\mathsf{KS}, \mathsf{KS}_{\leq}, \mathsf{SSS}, \mathsf{SSS}_{\leq}, \mathsf{SSS}^{\neq}, \mathsf{SSS}^{\neq}_{\leq}\}$ *we have*

$$\mathsf{pX}[\mathsf{AbPG}] \leq^{\mathsf{fpt}} \mathsf{pX}[\mathsf{FAbG}] \quad \textit{and} \quad \mathsf{pX}[\mathsf{CycPG}] \leq^{\mathsf{fpt}} \mathsf{pX}[\mathsf{FCycG}].$$

This result is an immediate consequence of the following result from Iliopoulos (1988); McKenzie and Cook (1987):

Theorem 4.22 (c.f. Iliopoulos (1988); McKenzie and Cook (1987))**.** *From a list* $\pi_1, \ldots, \pi_m, \pi \in S_n$ *of pairwise commuting permutations one can compute in polynomial time the following:*

- *unary encoded numbers* $n_1, \ldots, n_d$ *such that* $\langle \pi_1, \ldots, \pi_m \rangle \cong \prod_{j=1}^{d} \mathbb{Z}_{n_j}$ *and*
- *an isomorphism* $h : \langle \pi_1, \ldots, \pi_m \rangle \to \prod_{i=1}^{d} \mathbb{Z}_{n_i}$ *that is represented by* $h(\pi_1), \ldots, h(\pi_m)$.

For the proof of Theorem 4.20 we will only need the case of Theorem 4.22 for cyclic permutation groups. For this case, we provide a self-contained proof in Appendix A.

Lemma 4.23. $\mathsf{pSSS}[\mathsf{FCycG}] \leq^{\mathsf{fpt}} \mathsf{pSSS}[\mathbb{Z}^*]$

Proof: Let $\mathbb{Z}_n$ be the input group with n given in binary representation and let $a, a_1, \ldots, a_m \in [0, n-1]$ be the input elements.

We have to check whether there exists a subset $I \in [1, m]$ such that $|I| = k$ and

$$\sum_{i \in I} a_i \equiv a \mod n. \tag{11}$$

For every $I \in [1, m]$ of size k, (11) is equivalent to

$$\exists q \in [0, k] : \sum_{i \in I} a_i - q \cdot n = a. \tag{12}$$

We define $a' = (k, a)$, $a'_i = (1, a_i)$ and $b = (0, -n)$. These are elements of $\mathbb{Z}^2$. Let us consider the sequence $(a'_1, a'_2 \ldots, a'_m, b, \ldots, b)$ of length $m + k$, where b appears exactly k times. Write this sequence as $(c_1, c_2, \ldots, c_{m+k})$. Then there is $I \in [1, m]$ such that $|I| = k$ and (12) holds, if and only if there is a subset $J \subseteq [1, m + k]$ such that $|J| \leq 2k$ and

$$\sum_{j \in J} c_j = a'. \tag{13}$$

Finally, by adding the zero vector to the sequence $(c_1, c_2, \ldots, c_{m+k})$, we can replace the condition $|J| \leq 2k$ by the condition $|J| = 2k$. □

Lemma 4.24. $\mathsf{pSSS}[\mathbb{N}] \leq^{\mathsf{fpt}} \mathsf{pSSS}[\mathsf{CycPG}]$

Proof: The result is an easy consequence of the Chinese remainder theorem (Theorem 2.1). To see this, let $a, a_1, \ldots, a_m \in \mathbb{N}$ and $k \in \mathbb{N}$ be the input data for $\mathsf{pSSS}[\mathbb{N}]$. Let $e = \max\{a_1, \ldots, a_m\}$ and d be the smallest number such that

$$\prod_{i=1}^{d} p_i > ke,$$

where $p_1 < \cdots < p_d$ are the first d primes. Let $N = \prod_{i\in[1,d]} p_i$ so that $ke < N$. We have $d \in \mathcal{O}(\log k + \log e)$ and $p_d \in \mathcal{O}(d \log d)$. The list $p_1, \ldots, p_d$ with every p_i in *unary* encoding can be computed in time $\mathrm{poly}(\log k, \log e)$.

Note that our $\mathsf{pSSS}[\mathbb{N}]$ instance has no solution if $a > ke$. In this case the reduction returns some negative $\mathsf{pSSS}[\mathsf{CycPG}]$-instance. Hence we assume that $a \leq ke < N$ in the following.

The group we are working with is $G = \prod_{i=1}^{d} \mathbb{Z}_{p_i}$. Since the primes are chosen pairwise different, Theorem 2.1 implies that the group G is isomorphic to the cyclic group $\mathbb{Z}_N$. The group G can be embedded into S_n for $n = \sum_{i\in[1,d]} p_i$ by identifying the direct factors $\mathbb{Z}_{p_i}$ of G with disjoint cycles of length p_i ($i \in [1, d]$). For $i \in [1, m]$ and $j \in [1, d]$ let

$$z_{i,j} = a_i \bmod p_j \in [0, p_j - 1] \text{ and } z_j = a \bmod p_j \in [0, p_j - 1].$$

Then, for every subset $I \subseteq [1, m]$ of size k we have $\sum_{i\in I} a_i = a$ if and only if $\sum_{i\in I} a_i \equiv a \mod N$ if and only if $\sum_{i\in I} z_{i,j} \equiv z_j \mod p_j$ for all $j \in [1, d]$. Here we use the Chinese remainder theorem for the second equivalence. □

Corollary 4.25. *All problems* $\mathsf{pX}[\mathsf{C}]$ *for*

- $\mathsf{X} \in \{\mathsf{SSS}, \mathsf{SSS}_{\leq}, \mathsf{SSS}^{\neq}, \mathsf{SSS}^{\neq}_{\leq}\}$ *and*
- $\mathsf{C} \in \{\mathsf{CycPG}, \mathsf{FCycG}, \mathbb{N}, \mathbb{Z}, \mathbb{N}^*, \mathbb{Z}^*\}$

are equivalent with respect to fpt-reductions.

Proof: We have

$$\begin{array}{lll}
\mathsf{pSSS}[\mathsf{CycPG}] & \leq^{\mathsf{fpt}} \mathsf{pSSS}[\mathsf{FCycG}] & \text{(Lemma 4.21)} \\
& \leq^{\mathsf{fpt}} \mathsf{pSSS}[\mathbb{Z}^*] & \text{(Lemma 4.23)} \\
& \leq^{\mathsf{fpt}} \mathsf{pSSS}[\mathbb{N}^*] & \text{(Lemma 4.14)} \\
& \leq^{\mathsf{fpt}} \mathsf{pSSS}[\mathbb{N}] & \text{(Lemma 4.15)} \\
& \leq^{\mathsf{fpt}} \mathsf{pSSS}[\mathsf{CycPG}]. & \text{(Lemma 4.24)}
\end{array}$$

Finally from Corollary 4.19 we obtain for every $\mathsf{C} \in \{\mathsf{CycPG}, \mathsf{FCycG}, \mathbb{N}, \mathbb{Z}, \mathbb{N}^*, \mathbb{Z}^*\}$ the equivalence of the problems $\mathsf{pSSS}[\mathsf{C}], \mathsf{pSSS}^{\neq}[\mathsf{C}], \mathsf{pSSS}_{\leq}[\mathsf{C}], \mathsf{pSSS}^{\neq}_{\leq}[\mathsf{C}]$ with respect to fpt-reductions. □

The missing piece in the proof of Theorem 4.20 is the following:

Theorem 4.26. *If* C *is one of the commutative classes from Definition 4.4 then*

$$\mathsf{pSSS}[\mathsf{C}] \leq^{\mathsf{fpt}} \mathsf{pKS}_{\leq}[\mathsf{C}] \leq^{\mathsf{fpt}} \mathsf{pKS}[\mathsf{C}] \leq^{\mathsf{fpt}} \mathsf{pSSS}[\mathsf{C}].$$

Proof: By Lemmas 4.7 and 4.16 the statement $\mathsf{pKS}_{\leq}[\mathsf{C}] \leq^{\mathsf{fpt}} \mathsf{pKS}[\mathsf{C}] \leq^{\mathsf{fpt}} \mathsf{pSSS}[\mathsf{C}]$ holds for every class C. It remains to show $\mathsf{pSSS}[\mathsf{C}] \leq^{\mathsf{fpt}} \mathsf{pKS}_{\leq}[\mathsf{C}]$. The idea is similar to the proof of Lemma 4.18. It suffices to show $\mathsf{pSSS}^{\neq}[\mathsf{C}] \leq^{\mathsf{fpt}} \mathsf{pKS}_{\leq}[\mathsf{C}]$. We first prove the theorem for $\mathsf{C} = \mathbb{Z}^*$.

Let $A \subseteq \mathbb{Z}^n$ be finite, $a \in \mathbb{Z}^n$ and k be the parameter. Let $A = \{a_1, \ldots, a_m\}$. We then move to the group $\mathbb{Z}^{m+n+1}$. For $i \in [1, m]$ let $e_i \in \mathbb{Z}^m$ be the i^{th} unit vector, i.e., the vector with m entries, where the i^{th} entry is 1 and all other entries are zero. We define $b_i = (1, e_i, a_i) \in \mathbb{Z}^{m+n+1}$ and $b = (k, 1^m, a) \in \mathbb{Z}^{m+n+1}$. We then add all filling vectors $(0, 0^i, 1^j, 0^l, 0^n)$, where $i + j + l = m$. Consider the set $B = \{b_1, \ldots, b_m\} \cup F$, where F is the set of all filling vectors.

We claim that there is a subset $S \subseteq A$ of size k with $\sum_{x \in S} x = a$ if and only if $b \in B^{\leq 2k+1}$.[v] First assume that $\sum_{x \in S} x = a$ with $|S| = k$. Let $S = \{a_{i_1}, \ldots, a_{i_k}\}$ with $i_1 < i_2 < \cdots < i_k$. Then $\sum_{j=1}^{k} b_{i_j} = (k, v, a)$, where $v \in \{0, 1\}^m$ has exactly k ones. Hence, by adding at most $k + 1$ filling vectors to $\sum_{j=1}^{k} b_{i_j}$, we obtain $b = (k, 1^m, a)$.

For the other direction, assume that $b \in B^{\leq 2k+1}$ and let $b = \sum_{i=1}^{l} c_i$ with $c_i \in B$ and $l \leq 2k + 1$. Inspecting the first coordinate shows that the sum $\sum_{i=1}^{l} c_i$ contains exactly k vectors from $\{b_1, \ldots, b_m\}$. Moreover, since the entries at positions $2, \ldots, m+1$ in b are 1, these k vectors must be pairwise different. By projecting onto the last coordinate we obtain a subset $S \subseteq A$ of size k with $\sum_{x \in S} x = a$.

The above idea also works for $\mathsf{C} \in \{\mathsf{CycPG}, \mathsf{AbPG}, \mathsf{FAbG}, \mathbb{Z}\}$. For $\mathsf{C} = \mathbb{Z}$ we can use the above reduction together with $\mathsf{pKS}_{\leq}[\mathbb{Z}^*] \leq^{\mathsf{fpt}} \mathsf{pKS}_{\leq}[\mathbb{Z}]$, which is obtained as follows:

$$\begin{array}{lll} \mathsf{pKS}_{\leq}[\mathbb{Z}^*] & \leq^{\mathsf{fpt}} \mathsf{pKS}[\mathbb{Z}^*] & \text{(Lemma 4.16)} \\ & \leq^{\mathsf{fpt}} \mathsf{pKS}[\mathbb{N}^*] & \text{(Lemma 4.14)} \\ & \leq^{\mathsf{fpt}} \mathsf{pKS}[\mathbb{N}] & \text{(Lemma 4.15)} \\ & \leq^{\mathsf{fpt}} \mathsf{pKS}_{\leq}[\mathbb{N}]. & \text{(Lemma 4.17)} \end{array}$$

Finally, for $\mathsf{C} = \mathsf{FCycG}$ we obtain

$$\mathsf{pSSS}[\mathsf{FCycG}] \equiv^{\mathsf{fpt}} \mathsf{pSSS}[\mathsf{CycPG}] \leq^{\mathsf{fpt}} \mathsf{pKS}_{\leq}[\mathsf{CycPG}] \leq^{\mathsf{fpt}} \mathsf{pKS}_{\leq}[\mathsf{FCycG}]$$

with Corollary 4.25 and Lemma 4.21. □

4.4.3 Abelian groups

Clearly, the problems $\mathsf{pKS}[\mathsf{AbPG}]$ and $\mathsf{pSSS}[\mathsf{AbPG}]$ are $\mathsf{W}[1]$-hard; this follows from the corresponding results for $\mathsf{pF}[\mathsf{CycPG}]$ and $\mathsf{pSSS}[\mathsf{CycPG}]$. The best upper bound that we can show is $\mathsf{W}[5]$:

Theorem 4.27. *The problems* $\mathsf{pSSS}[\mathsf{FAbG}]$ *and* $\mathsf{pKS}[\mathsf{FAbG}]$ *belong to* $\mathsf{W}[5]$.

---

[v] Note that when writing $b \in B^{\leq 2k+1}$ we use the multiplicative notation for the abelian group $\mathbb{Z}^{m+n+1}$.

Proof: It suffices to show $\mathsf{pSSS[FAbG]} \in \mathsf{W}[5]$ because by Lemma 4.7 we have $\mathsf{pKS[FAbG]} \leq^{\mathsf{fpt}} \mathsf{pSSS[FAbG]}$. Let $t_i = (t_{i,1}, \ldots, t_{i,d}) \in \prod_{j=1}^{d} \mathbb{Z}_{n_j}$ for $i \in [1,m]$, $u = (u_1, \ldots, u_d) \in \prod_{j=1}^{d} \mathbb{Z}_{n_j}$ and $k \in \mathbb{N}$ be the input for $\mathsf{pSSS[FAbG]}$. We can assume that $t_{i,j}, u_j \in [0, n_j - 1]$ for all $i \in [1,m]$, $j \in [1,d]$.

First we give a reduction to an equivalent problem, where we avoid computations modulo n_j. We have

$$\exists x_1, \ldots, x_m \in \{0,1\} : \sum_{i=1}^{m} x_i = k \text{ and } \forall j \in [1,d] : \sum_{i=1}^{m} x_i t_{i,j} \equiv u_j \bmod n_j$$

if and only if

$$\exists x_1, \ldots, x_m \in \{0,1\} : \sum_{i=1}^{m} x_i = k \text{ and } \forall j \in [1,d]\ \exists y \in [0, k-1] : \sum_{i=1}^{m} x_i t_{i,j} = u_j + y \cdot n_j.$$

The values $u_{y,j} := u_j + y \cdot n_j \in \mathbb{N}$ for $j \in [1,d]$ and $y \in [0, k-1]$ can be precomputed in FPT. Now the input for our new problem consists of the tuples $t_1, \ldots, t_m$ as defined above, the parameter $k \in \mathbb{N}$, and a matrix

$$\begin{bmatrix} u_{0,1} & \cdots & u_{0,d} \\ \vdots & \ddots & \vdots \\ u_{k-1,1} & \cdots & u_{k-1,d-1} \end{bmatrix} \in \mathbb{N}^{k \times d}.$$

This representation allows us to do all computations in $\mathbb{N}$.

Now we present an algorithm for the above problem. Most parts of the algorithm are identical to the $\mathsf{W}[3]$-algorithm of Buss and Islam (2007) for $\mathsf{pSSS}^{\neq}[\mathbb{Z}]$. It is straightforward to transform this algorithm into a $\Sigma_{5,2}$-formula that only depends on k. As in Buss and Islam (2007) we also represent all numbers in base r with $r = 2^{\lceil \log k \rceil} \in [k, 2k]$ by taking blocks of $\lceil \log k \rceil$ bits in the binary representations of the numbers. Let us write

$$(b_0, b_1, \ldots, b_p)_r = \sum_{i=0}^{p} b_i r^i$$

for $b_0, \ldots, b_p \in [0, r-1]$. Let $t_{i,j,p}$ (respectively, $u_{y,j,p}$) be the p^{th} base-r digit of the number $t_{i,j}$ (respectively, $u_{y,j}$). We start with $p = 0$. Assume that $m_j + 1$ is the maximal number of digits in the base-r expansions of the numbers $t_{i,j}$, $u_{y,j}$ for $i \in [1,m]$, $y \in [0, k-1]$. The algorithm consists of the following steps:

(a) Existentially guess k tuples $t_{i_1}, \ldots, t_{i_k}$ $(i_1 < i_2 < \cdots < i_k)$.

(b) Universally guess a coordinate $j \in [1,d]$.

(c) Existentially guess a $y \in [0, k-1]$. The remaining goal is to check whether $u_{y,j} = t_{i_1,j} + \cdots + t_{i_k,j}$. We define the following base-r digits, where $p \in [0, m_j]$:

$$\begin{aligned} s_p &= (t_{i_1,j,p} + \cdots + t_{i_k,j,p}) \bmod r \in [0, r-1], \\ s_{m_j+1} &= 0, \\ c_0 &= 0, \end{aligned}$$

$$c_{p+1} \quad = \quad (t_{i_1,j,p} + \cdots + t_{i_k,j,p}) \operatorname{div} r \in [0, r-1].$$

Note that $t_{i_1,j,p} + \cdots + t_{i_k,j,p} \leq k(r-1) \leq r(r-1) = r^2 - r$ and

$$t_{i_1,j} + \cdots + t_{i_k,j} = \sum_{p=0}^{m_j+1} (s_p r^p + c_p r^p) = \underbrace{(s_0, \ldots, s_{m_j+1})_r}_{v} + \underbrace{(c_0, \ldots, c_{m_j+1})_r}_{w}.$$

The new goal is to check whether $v + w = u_{y,j}$. Note that when adding v and w, the maximal carry that can arrive at a certain position is 1 (since $2(r-1)+1 = (r-1,1)_r$).

(d) Universally select a position $p \in [0, m_j + 1]$.

(e) Compute $\delta = (u_{y,j,p} - s_p - c_p) \bmod r$. Reject if δ is neither 0 nor 1.

(f) If $\delta = 0$ then do the following (we verify that no carry arrives at position p):

(i) Universally select a position $p' \in [0, p-1]$ that generates a carry (i.e., $s_{p'} + c_{p'} \geq r$).

(ii) Existentially select a position $q \in [p'+1, p-1]$ such that a carry from position $q-1$ is not propagated through position q (i.e., $s_q + c_q < r-1$).

(g) If $\delta = 1$ then do the following (we verify that a carry arrives at position p):

(ii) Universally select a position $p' \in [-1, p-1]$.

(iiii) If $p' = -1$ or the digits at position p' do not propagate a carry (i.e., $s_{p'} + c_{p'} < r-1$) then existentially select a position $q \in [p'+1, p-1]$ such that a carry is generated at position q (i.e., $s_q + c_q \geq r$).

It is straightforward (but tedious) to translate the above algorithm into a $\Sigma_{5,2}$-sentence that only depends on the parameter k. The existential (respectively, universal) guesses correspond to existential (respectively, universal) quantifiers in the formula. The constructed formula is model-checked in a structure that is constructed from the input numbers. For each of the following elements we add an element to the universe of the structure:

(1) every $i \in [1, m]$, where i represents the tuple t_i,

(2) every dimension $j \in [1, d]$,

(3) every $y \in [0, k-1]$,

(4) every position p that exists in one of the base-r expansions of the numbers $t_{i,j}$ and $u_{y,j}$, and

(5) all $\alpha \in [0, r-1]$ that stand for the base-r digits.

Moreover, we need the following predicates:

- four unary predicates I, J, Y, and P that hold exactly for the elements from (1), (2), (3), and (4), respectively,

- a 4-ary relation dig_t that contains the tuple (i, j, p, α) if $\alpha \in [0, r-1]$ is the digit at position p in the number $t_{i,j}$,
- a 4-ary relation dig_u that contains the tuple (y, j, p, α) if $\alpha \in [0, r-1]$ is the digit at position p in the number $u_{y,j}$,
- the natural linear order $<_P$ on the positions from (4), and
- the natural linear order $<_r$ on $[0, r-1]$.

Note that the sets from (1)–(5) do not have to be disjoint.

The $\Sigma_{5,2}$-formula starts with the prefix

$$\exists i_1 \exists i_2 \cdots \exists i_k : \bigwedge_{l=1}^{k} I(i_l) \wedge \forall j : J(j) \to \exists y : Y(y) \wedge \forall p : P(p) \to \cdots . \tag{14}$$

These quantifiers correspond to the guesses in steps (a)–(d) of the above algorithm. For the remaining part of the formula we need for every $a \in [0, r-1]$ formulas $R_a(i_1, \ldots, i_k, j, p)$ and $Q_a(i_1, \ldots, i_k, j, p)$, which express $(t_{i_1,j,p} + \cdots + t_{i_k,j,p}) \bmod r = a$ and $(t_{i_1,j,p} + \cdots + t_{i_k,j,p}) \operatorname{div} r = a$, respectively. Here, we use the variables quantified in (14). Define the finite set

$$A_a = \{(a_1, \ldots, a_k) \in [0, r-1]^k : (a_1 + \cdots + a_k) \bmod r = a\} \subseteq [0, r-1]^k.$$

It can be precomputed from k and its size only depends on k. Then we define the quantifier-free formula $R_a(i_1, \ldots, i_k, j, p)$ by

$$\bigvee_{(a_1,\ldots,a_k) \in A_a} \bigwedge_{1 \le l \le k} \mathsf{dig}_t(i_l, j, p, a_l).$$

The formula $Q_a(i_1, \ldots, i_k, j, p)$ can be obtained analogously. Note that these formulas only depend on the parameter k. With the formulas $R_a(i_1, \ldots, i_k, j, p)$ and $Q_a(i_1, \ldots, i_k, j, p)$ we can access the numbers $s_p, c_p \in [0, r-1]$ from step (c) of the algorithm. Using them, one can construct analogously to $S_a(i_1, \ldots, i_k, j, p)$ quantifier-free formulas expressing conditions like $(u_{y,j,p} - s_p - c_p) \bmod r = 0$ and $(u_{y,j,p} - s_p - c_p) \bmod r = 1$ (see steps (e)–(g) in the algorithm). We leave the further details to the reader. □

5 Change making problems

Note that in the following proofs for a positive integer $N \in \mathbb{N}$ we use $\#(N)$ to denote the number of bits of N.

The problem $\mathsf{pKS}_{\le}[\mathbb{N}]$ (respectively, $\mathsf{pSSS}^{\neq}_{\le}[\mathbb{N}]$) was studied in Goebbels et al. (2017) under the name p-unbounded-change-making (respectively, p-0-1-change-making). A third variant from Goebbels et al. (2017) is:

Problem 5.1 (p-bounded-change-making Goebbels et al. (2017))**.**
Input: binary encoded numbers $c, c_1, b_1 \ldots, c_m, b_m \in \mathbb{N}$ with $c_i \neq c_j$ for $i \neq j$ and $k \in \mathbb{N}$ (the parameter)
Question: Are there $x_i \in [0, b_i]$ $(i \in [1, m])$ with $\sum_{i=1}^{m} x_i c_i = c$ and $\sum_{i=1}^{m} x_i \le k$?

It was shown in Goebbels et al. (2017) that these change-making problems are W[1]-hard and contained in XP. For p-unbounded-change-making and p-0-1-change-making we obtain membership in W[3] from Theorem 4.20. Also p-bounded-change-making belongs to W[3]. For this, notice that $\mathsf{pKS}_{\leq}[\mathbb{N}] \leq^{\mathsf{fpt}}$ p-bounded-change-making: if we set $b_i = k$ for all $i \in [1, m]$ in p-bounded-change-making, we obtain $\mathsf{pKS}_{\leq}[\mathbb{N}]$. Finally, also p-bounded-change-making $\leq^{\mathsf{fpt}} \mathsf{pSSS}^{\neq}_{\leq}[\mathbb{N}]$ holds: because of the restriction $\sum_{i=1}^{m} x_i \leq k$ in p-bounded-change-making, we can assume that $b_i \leq k$ for all $i \in [1, m]$. We obtain an instance of $\mathsf{pSSS}_{\leq}[\mathbb{N}]$ by duplicating every c_i in a p-bounded-change-making-instance k times.

Theorem 5.2. *The problem* p-bounded-change-making *is equivalent with respect to fpt-reductions to the problems from Theorem 4.20 and therefore belongs to* W[3].

In Goebbels et al. (2017), approximate versions of the change-making problems, where a linear objective function is minimized, were defined:

Problem 5.3 (p-unbounded-change-approx Goebbels et al. (2017))**.**
Input: binary encoded numbers $c, c_1, \ldots, c_m \in \mathbb{N}$ with $c_i \neq c_j$ for $i \neq j$, a linear function $f(x, y) = ax + by$ given by $a, b \in \mathbb{N}$ and $k \in \mathbb{N}$ (the parameter)
Question: Are there $x_1, \ldots, x_m \in \mathbb{N}$ with $f(\sum_{i=1}^{m} x_i c_i - c, \sum_{i=1}^{m} x_i) \leq k$ and $\sum_{i=1}^{m} x_i c_i \geq c$?

Problem 5.4 (p-bounded-change-approx Goebbels et al. (2017))**.**
Input: binary encoded numbers $c, c_1, b_1 \ldots, c_m, b_m \in \mathbb{N}$ with $c_i \neq c_j$ for $i \neq j$, a linear function $f(x, y) = ax + by$ given by $a, b \in \mathbb{N}$ and $k \in \mathbb{N}$ (the parameter)
Question: Are there $x_1, \ldots, x_m \in [0, b_i]$ with $f(\sum_{i=1}^{m} x_i c_i - c, \sum_{i=1}^{m} x_i) \leq k$ and $\sum_{i=1}^{m} x_i c_i \geq c$?

Finally, one obtains the problem p-0-1-change-approx by fixing b_i to 1 for all $i \in [1, m]$ in p-bounded-change-approx Goebbels et al. (2017). Also for the above approximation variants, W[1]-hardness and containment in XP was shown in Goebbels et al. (2017). With our results we can improve the upper bounds:

Theorem 5.5. *The problems* $\mathsf{pSSS}[\mathbb{N}]$, p-unbounded-change-approx, p-bounded-change-approx *and* p-0-1-change-approx *are equivalent under fpt-reductions when restricted to functions $f(x, y) = ax + by$ with $(a, b) \in \mathbb{N} \times \mathbb{N} \setminus \{(a, 0) : a \geq 1\}$. Therefore the problems belong to* W[3].

Proof: We prove the theorem in the following order:

(i) $\mathsf{pKS}_{\leq}[\mathbb{N}] \leq^{\mathsf{fpt}}$ p-unbounded-change-approx,

(ii) p-unbounded-change-approx $\leq^{\mathsf{fpt}}$ p-bounded-change-approx,

(iii) $\mathsf{pSSS}^{\neq}_{\leq}[\mathbb{N}] \leq^{\mathsf{fpt}}$ p-0-1-change-approx,

(iv) p-0-1-change-approx $\leq^{\mathsf{fpt}}$ p-bounded-change-approx,

(v) p-bounded-change-approx $\leq^{\mathsf{fpt}} \mathsf{pSSS}_{\leq}[\mathbb{Z}]$.

Since (iv) is trivial and (iii) can be proven analogously to (i) we only show the reductions for (i), (ii) and (v). For (i) notice that an instance of $\mathsf{pKS}_{\leq}[\mathbb{N}]$ can be transformed into an equivalent instance of p-unbounded-change-approx by taking the linear function $f(x, y) = (k + 1)x + y$.

For (ii) let $c, c_1, \ldots, c_m, f(x,y) = ax+by$, and k be the input data for p-unbounded-change-approx. By setting $b_i = k$ for $i \in [1,m]$ we obtain an equivalent instance of p-bounded-change-approx if $b > 0$. If $b = 0$ then also $a = 0$ and hence the condition $f(\sum_{i=1}^m x_i c_i - c, \sum_{i=1}^m x_i) \leq k$ is always satisfied. The p-unbounded-change-approx-instance is then positive if and only if $c = 0$ or $c_i > 0$ for at least one $i \in [1,m]$.

It remains to show (v). Let $c, c_1, b_1, \ldots, c_m, b_m, f(x,y) = ax + by$, and k be the input data for p-bounded-change-approx. If $a = 0 = b$ the instance is positive if and only if $\sum_{i=1}^m b_i c_i \geq c$.

Suppose now that $a = 0$ and $b \geq 1$ and w.l.o.g. $c_1 > c_2 > \cdots > c_m$. We show that it can be checked in polynomial time whether the p-bounded-change-approx-instance is positive. The latter holds if and only if there exist $x_i \in [0, b_i]$ ($i \in [1,m]$) with $\sum_{i=1}^m x_i c_i \geq c$ and $\sum_{i=1}^m x_i \leq \lfloor k/b \rfloor$. We can w.l.o.g. assume that $b_i > 0$ for all $i \in [1,m]$. If $\sum_{i=1}^m b_i \leq \lfloor \frac{k}{b} \rfloor$ then it suffices to check whether $\sum_{i=1}^m b_i c_i \geq c$. If $\sum_{i=1}^m b_i > \lfloor \frac{k}{b} \rfloor$ then we compute the unique $l \in [0, m-1]$ and $b'_{l+1} < b_{l+1}$ such that $\sum_{i=1}^l b_i + b'_{l+1} = \lfloor \frac{k}{b} \rfloor$. It then suffices to check whether $\sum_{i=1}^l b_i c_i + b'_{l+1} c_{l+1} \geq c$.

The case that remains is $a, b \geq 1$. We define the following integers:

$$\begin{aligned} d &= c \cdot 2^{\#(ka)+\#(kb)} & d_i &= b - 1 + c_i \cdot 2^{\#(ka)+\#(kb)} \text{ for } i \in [1,m] \\ d_0 &= -1 & d_{m+1} &= (1-a)2^{\#(kb)} - 2^{\#(ka)+\#(kb)} \\ & & d_{m+2} &= 2^{\#(kb)}. \end{aligned}$$

Moreover, we set $k = b_0 = b_{m+1} = b_{m+2}$.

Claim 5.6. *The p-bounded-change-approx-instance $c, c_1, b_1, \ldots, c_m, b_m, f(x,y) = ax + by$, k is positive if and only if there exist $x_i \in [0, b_i]$ $(0 \leq i \leq m+2)$ such that*

$$\sum_{i=0}^{m+2} x_i d_i = d \quad \textit{and} \quad \sum_{i=0}^{m+2} x_i \leq k.$$

Proof of Claim 5.6. First suppose the p-bounded-change-approx-instance is positive. Then there exist $x_i \in [0, b_i]$ with

$$\sum_{i=1}^m x_i c_i \geq c \quad \text{and} \quad a\Big(\sum_{i=1}^m x_i c_i - c\Big) + b \sum_{i=1}^m x_i \leq k.$$

By choosing $x_0 = (b-1)\sum_{i=1}^m x_i$ and $x_{m+1} = \sum_{i=1}^m x_i c_i - c$ and $x_{m+2} = (a-1)(\sum_{i=1}^m x_i c_i - c)$ we obtain

$$\begin{aligned} \sum_{i=0}^{m+2} x_i d_i &= (b-1)\big(\sum_{i=1}^m x_i\big) d_0 + \sum_{i=1}^m x_i d_i + \big(\sum_{i=1}^m x_i c_i - c\big) d_{m+1} + (a-1)\big(\sum_{i=1}^m x_i c_i - c\big) d_{m+2} \\ &= -(b-1)\sum_{i=1}^m x_i + \sum_{i=1}^m x_i (b-1) + \sum_{i=1}^m x_i c_i \cdot 2^{\#(ka)+\#(kb)} \\ &\quad + \big(\sum_{i=1}^m x_i c_i - c\big)(1-a) \cdot 2^{\#(kb)} - \big(\sum_{i=1}^m x_i c_i - c\big) \cdot 2^{\#(ka)+\#(kb)} \end{aligned}$$

$$
\begin{aligned}
&+ (a-1)\big(\sum_{i=1}^{m} x_i c_i - c\big) \cdot 2^{\#(kb)} \\
&= \sum_{i=1}^{m} x_i c_i \cdot 2^{\#(ka)+\#(kb)} - \big(\sum_{i=1}^{m} x_i c_i - c\big) \cdot 2^{\#(ka)+\#(kb)} \\
&= c \cdot 2^{\#(ka)+\#(kb)} = d
\end{aligned}
$$

and

$$
\begin{aligned}
\sum_{i=0}^{m+2} x_i &= (b-1)\sum_{i=1}^{m} x_i + \sum_{i=1}^{m} x_i + \sum_{i=1}^{m} x_i c_i - c + (a-1)\big(\sum_{i=1}^{m} x_i c_i - c\big) \\
&= a\big(\sum_{i=1}^{m} x_i c_i - c\big) + b\sum_{i=1}^{m} x_i \leq k.
\end{aligned}
$$

Therefore we have $x_0, x_{m+1}, x_{m+2} \in [0,k]$. Thus $x_i \in [0,b_i]$ for all $i \in [0,m+2]$.

For the other direction in Claim 5.6 assume that $x_i \in [0,b_i]$ are such that

$$
\sum_{i=0}^{m+2} x_i d_i = d \quad \text{and} \quad \sum_{i=0}^{m+2} x_i \leq k. \tag{15}
$$

We bring the left equation in (15) in a form where all numbers are positive:

$$
\sum_{i=1}^{m} x_i d_i + x_{m+2} d_{m+2} = d + x_0(-d_0) + x_{m+1}(-d_{m+1}),
$$

or, after plugging in the values of the d_i and rearranging,

$$
\begin{aligned}
&\sum_{i=1}^{m} x_i(b-1) + \quad x_{m+2}\, 2^{\#(kb)} \;+\; \Big(\sum_{i=1}^{m} x_i c_i\Big)\, 2^{\#(ka)+\#(kb)} \\
&= \quad x_0 + x_{m+1}(a-1)\, 2^{\#(kb)} \;+\; (c + x_{m+1})\; 2^{\#(ka)+\#(kb)}.
\end{aligned}
$$

Note that $\sum_{i=1}^{m} x_i(b-1)$ and x_0 are bounded by kb and therefore have at most $\#(kb)$ bits. All other terms in the above equation are multiplied with $2^{\#(kb)}$ and therefore do not affect the first $\#(kb)$ bits. Similarly, x_{m+2} and $x_{m+1}(a-1)$ are bounded by $\#(ka)$ and therefore have at most $\#(ka)$ bits. We therefore split the above equation into three simultaneous equations:

$$
\begin{aligned}
\sum_{i=1}^{m} x_i(b-1) &= x_0 && (16)\\
x_{m+2} &= x_{m+1}(a-1) && (17)\\
\sum_{i=1}^{m} x_i c_i &= c + x_{m+1}. && (18)
\end{aligned}
$$

By (18) we have $x_{m+1} = \sum_{i=1}^{m} x_i c_i - c$. Together with (17) this gives us $x_{m+2} = (a-1)(\sum_{i=1}^{m} x_i c_i - c)$. Adding up all variables and using (16) yields

$$\begin{aligned} k \geq \sum_{i=0}^{m+2} x_i &= \sum_{i=1}^{m} x_i(b-1) + \sum_{i=1}^{m} x_i + \sum_{i=1}^{m} x_i c_i - c + (a-1)\left(\sum_{i=1}^{m} x_i c_i - c\right) \\ &= a\left(\sum_{i=1}^{m} x_i c_i - c\right) + b\sum_{i=1}^{m} x_i. \end{aligned}$$

Moreover since we have $x_{m+1} \geq 0$, (18) finally implies $\sum_{i=1}^{m} x_i c_i \geq c$ which proves the claim. □

By Claim 5.6 we obtain an equivalent instance of $\mathsf{pSSS}_{\leq}[\mathbb{Z}]$ by defining a list where for $i \in [0, m+2]$ the number d_i is contained exactly b_i times. □

5.1 NP-hardness of the approximative change-making problems for $a > 0$ and $b = 0$

Theorem 5.7. *Let $a \geq 1$ and $b = 0$. Then for all $d \geq 1$ the d^{th} slices of the problems* p-unbounded-change-approx*,* p-bounded-change-approx *and* p-0-1-change-approx *are* NP*-complete. Moreover, these three problems are in* para-NP*.*

Proof: Membership of all three problems in NP is straightforward. We guess numbers $x_i \in [0, \lceil \frac{c}{c_i} \rceil]$ (for p-unbounded-change-approx), $x_i \in [0, b_i]$ (for p-bounded-change-approx) and $x_i \in \{0, 1\}$ (for p-0-1-change-approx), respectively, and verify

$$\sum_{i=1}^{m} x_i c_i \geq c \quad \text{and} \quad a\left(\sum_{i=1}^{m} x_i c_i - c\right) \leq d.$$

The same algorithm shows membership in para-NP for the parameterized problems.

Now we show NP-hardness. The proof uses a modification of the construction given in the appendix of Goebbels et al. (2017). We start with an instance of the NP-complete problem SUBSET SUM where we are given positive integers $a_1 \leq a_2 \leq \cdots \leq a_m$ and $a \in \mathbb{N}$ in binary encoding. The question is whether there exist $x_1, \ldots, x_m \in \{0, 1\}$ such that $\sum_{i=1}^{m} x_i a_i = a$. W.l.o.g. we can assume $a \leq m a_m < 2^{\#(m a_m)}$. Otherwise the instance is negative and the reduction yields a fixed negative instance. Furthermore we can assume w.l.o.g. $a_i \neq 0$ for all $i \in [1, m]$. If $a > 0$ we can remove from the list all numbers that are equal to 0. In the case $a = 0$ the instance is trivially positive by choosing $x_i = 0$ for all variables and the reduction yields a fixed positive instance. We define for $i \in [1, m]$ the numbers

$$\begin{aligned} c_{2i-1} &= 0 + 2^{\#(m a_m)+(i-1)\#(m)} + 2^{\#(m a_m)+m\#(m)}, \\ c_{2i} &= a_i + 2^{\#(m a_m)+(i-1)\#(m)} + 2^{\#(m a_m)+m\#(m)}, \text{ and} \\ c &= a + \sum_{i=1}^{m} 2^{\#(m a_m)+(i-1)\#(m)} + m \cdot 2^{\#(m a_m)+m\#(m)}. \end{aligned}$$

Note that we have $c_1 < c_2 < \cdots < c_{2m}$.

Claim 5.8. *If $\sum_{i=1}^{m} x_i a_i = a$ with $x_i \in \{0,1\}$ then we have*

$$\sum_{i=1}^{m}(x_i c_{2i} + (1-x_i)c_{2i-1}) = c.$$

Proof of Claim 5.8. We have

$$\begin{aligned}\sum_{i=1}^{m}(x_i c_{2i} + (1-x_i)c_{2i-1}) &= \sum_{i=1}^{m} x_i a_i + \sum_{i=1}^{m} 2^{\#(ma_m)+(i-1)\#(m)} + \sum_{i=1}^{m} 2^{\#(ma_m)+m\#(m)} \\ &= a + \sum_{i=1}^{m} 2^{\#(ma_m)+(i-1)\#(m)} + m \cdot 2^{\#(ma_m)+m\#(m)} \\ &= c,\end{aligned}$$

which proves the claim. □

Claim 5.9. *Assume that $\sum_{i=1}^{2m} x_i c_i = c$ for $x_1, \ldots, x_{2m} \in \mathbb{N}$. Then the following holds:*

1. $\sum_{i=1}^{2m} x_i \leq m$
2. *For $i \in [1,m]$ we have $x_{2i-1}, x_{2i} \in \{0,1\}$ and $x_{2i-1} = 1$ if and only if $x_{2i} = 0$.*
3. $\sum_{i=1}^{m} x_{2i} a_i = a$

Proof of Claim 5.9. For the first statement, suppose that $\sum_{i=1}^{2m} x_i > m$. Then we obtain

$$\begin{aligned}\sum_{i=1}^{2m} x_i c_i &\geq (m+1)\cdot c_1 \\ &= (m+1)\cdot(2^{\#(ma_m)} + 2^{\#(ma_m)+m\#(m)}) \\ &\geq 2^{\#(ma_m)} + (m+1)\cdot 2^{\#(ma_m)+m\#(m)} \\ &> a + (m+1)\cdot 2^{\#(ma_m)+m\#(m)} \\ &= a + m\cdot 2^{\#(ma_m)+m\#(m)} + 2^{\#(ma_m)}\cdot 2^{m\#(m)} \\ &> a + m\cdot 2^{\#(ma_m)+m\#(m)} + 2^{\#(ma_m)}\cdot \sum_{i=1}^{m}\left(2^{\#(m)}\right)^{i-1} \\ &= a + m\cdot 2^{\#(ma_m)+m\#(m)} + \sum_{i=1}^{m} 2^{\#(ma_m)+(i-1)\#(m)} \;=\; c,\end{aligned}$$

which is a contradiction.

Let us now show the second statement of Claim 5.9. By the first statement we have $\sum_{i=1}^{2m} x_i \leq m$. Hence, the sum $c = \sum_{i=1}^{2m} x_i c_i$ is a sum of m numbers c_i, possibly with repetitions. The binary representations of the c_i are divided into blocks. The first block contains the $\#(ma_m)$ first bits and the next

blocks consist of exactly $\#(m)$ bits each. The lengths of the blocks are such that in the sum $\sum_{i=1}^{2m} x_i c_i$ a carry cannot enter a block from the previous block. The bit at position $1 + \#(ma_m) + (i-1)\#(m)$ (i.e., the first bit of the i^{th} block) in c, c_{2i-1}, c_{2i} is 1 (due to the summand $2^{\#(ma_m)+(i-1)\#(m)}$ in these numbers). In all other numbers c_j, the bit at position $1 + \#(ma_m) + (i-1)\#(m)$ is zero and by adding at most m numbers c_i one cannot get a carry at position $1 + \#(ma_m) + (i-1)\#(m)$. This implies that $x_{2i-1}, x_{2i} \in \{0,1\}$ and $x_{2i-1} = 1$ if and only if $x_{2i} = 0$.

Finally, the third statement of Claim 5.9 follows from the identity $c = \sum_{i=1}^{2m} x_i c_i$ (where $\sum_{i=1}^{2m} x_i \leq m$ by the first statement) and only taking the first block consisting of the $\#(ma_m)$ low-order bits. □

Claim 5.9 gives us a reduction from the subset sum equation

$$\sum_{i=1}^{m} x_i a_i = a$$

to an equation

$$\sum_{i=1}^{2m} x_i c_i = c$$

with $c_1 < \cdots < c_{2m}$. Moreover since by Claim 5.9 (second statement) it is ensured that we have for all variables $x_i \in \{0,1\}$ this reduction works in both the bounded and unbounded setting.

Now we come to the actual reduction from SUBSET SUM to the d^{th} slice of p-unbounded-change-approx (respectively, p-bounded-change-approx and p-0-1-change-approx). We have $\sum_{i=1}^{2m} x_i c_i = c$ if and only if $\sum_{i=1}^{2m} x_i (d+1) c_i = (d+1)c$, which holds if and only if the following two conditions hold:

$$\begin{aligned} \sum_{i=1}^{2m} x_i (d+1) c_i &\geq (d+1)c, \\ a\left(\sum_{i=1}^{m} x_i (d+1) c_i - (d+1)c\right) &\leq d. \end{aligned}$$

Note that $a \geq 1$. This concludes the reduction. □

Theorem 5.10. *When restricted to $a \geq 1$ and $b = 0$,* p-unbounded-change-approx*,* p-bounded-change-approx *and* p-0-1-change-approx *are* para-NP*-complete.*

Proof: By (Flum and Grohe, 2006, Theorem 2.14) a nontrivial parameterized problem in para-NP is para-NP-complete under fpt-reductions if and only if the union of finitely many slices is NP-complete under polynomial time reductions (the d^{th} slice of a parameterized problem is the restriction of the problem, where the parameter is set to the fixed value d). By Theorem 5.7 already a single slice of the problems from the theorem is NP-complete under polynomial time reductions. □

6 Conclusion

In this paper we have shown the equivalence of $\mathsf{pF}[\mathsf{S}_*]$, $\mathsf{pKS}[\mathsf{S}_*]$ and $\mathsf{pSSS}[\mathsf{S}_*]$ with respect to fpt-reductions. This may provide a tool for showing new upper and lower bounds for $\mathsf{pF}[\mathsf{S}_*]$ in the W-hierarchy because the equivalence resolves the problem to ensure the generators to factor in a specific order. The equivalence allows now to give the generators in a suitable order.

Moreover we have shown equivalence with respect to fpt-reductions of several problems, namely subset sum problems (for integers), change making problems and the special case of permutation group factorization for cyclic permutation groups ($\mathsf{pF}[\mathsf{CycPG}]$). By this and the membership in $\mathsf{W}[3]$ for the subset sum problem Buss and Islam (2007) we obtain membership in $\mathsf{W}[3]$ of the change making problems which addresses the gap between $\mathsf{W}[1]$-hardness and membership in XP that was obtained in Goebbels et al. (2017). In the same paper it was also asked for better lower bounds. However, improving the $\mathsf{W}[1]$-hardness for any of the change making problems seems to be very challenging for several reasons.

- Because of the equivalence with respect to fpt-reductions, a better lower bound for the change making problems would directly yield a better lower bound for permutation group factorization, for which in almost 30 years since the appearance of the paper Cai et al. (1997) nothing better than $\mathsf{W}[1]$-hardness has been shown.
- Proving $\mathsf{W}[3]$-completeness of the change making problems (and hence the problems from Theorem 4.20) would be surprising since it would imply

  $$\mathsf{W}[3] \subseteq \mathsf{W}^{\text{func}}[2] \subseteq \mathsf{A}[2],$$

  because permutation group factorization is a special case of transformation monoid factorization ($\mathsf{pF}[\mathsf{T}_*]$) which is contained in $\mathsf{W}^{\text{func}}[2]$ by Theorem 4.9. By Flum and Grohe (2006) we only know

  $$\mathsf{W}[2] \subseteq \mathsf{W}^{\text{func}}[2] \subseteq \mathsf{A}[2].$$

- The previous point still leaves room for an $\mathsf{W}[2]$-hardness proof. But there has been some progress in showing $\mathsf{W}[1]$-completeness for subset sum over $\mathbb{Z}$ Abboud et al. (2014), which makes it unlikely that better lower bounds are achievable. In Abboud et al. (2014) the following variant of subset sum over $\mathbb{Z}$ has been studied: The input consists of integers $z_1, \ldots, z_m \in \mathbb{Z}$ and $k \in \mathbb{N}$ (the parameter) and it is asked whether there are numbers $x_1, \ldots, x_m \in \{0, 1\}$ such that

  $$\sum_{i=1}^{m} x_i z_i = 0 \text{ with } \sum_{i=1}^{m} x_i = k.$$

  It is shown in Abboud et al. (2014) that this problem is $\mathsf{W}[1]$-complete when the numbers $z_1, \ldots, z_m$ are restricted to $[-n^{2k}, n^{2k}]$. In Abboud et al. (2014) the result was also improved to $\mathsf{W}[1]$-completeness for unrestricted integers $z_1, \ldots, z_m$ under the assumption that a certain circuit lower bound holds.

  Note that the subset sum version from Abboud et al. (2014) is a slight variant of our subset sum version $\mathsf{pSSS}[\mathbb{Z}]$, where the input consists of integers $z_1, \ldots, z_m, z \in \mathbb{Z}$ and $k \in \mathbb{N}$ (the parameter) and it is asked whether there are numbers $x_1, \ldots, x_m \in \{0, 1\}$ such that

  $$\sum_{i=1}^{m} x_i z_i = z \text{ with } \sum_{i=1}^{m} x_i = k. \tag{19}$$

But for $k \geq 1$ this is equivalent to asking whether there are numbers $x_1, \ldots, x_m \in \{0, 1\}$ such that

$$\sum_{i=1}^{m} x_i(kz_i - z) = 0 \text{ with } \sum_{i=1}^{m} x_i = k$$

In the case $k = 0$, (19) holds if and only if $z = 0$. Hence, the subset sum variant from Abboud et al. (2014) is equivalent to $\mathsf{pSSS}[\mathbb{Z}]$ with respect to fpt-reductions. Under the circuit lower bound assumption from Abboud et al. (2014) also all problems mentioned in Theorem 4.20 are $\mathsf{W}[1]$-complete since all these problems are equivalent to $\mathsf{pSSS}[\mathbb{Z}]$ with respect to fpt-reductions.

It would be also interesting to study the parameterized complexity of $\mathsf{pF}[\mathsf{T}_*]$ for restricted classes of transformation monoids (e.g. aperiodic or commutative transformation monoids). Due to the fpt-equivalence of $\mathsf{pF}[\mathsf{T}_*]$ and p-DFA-SW, this question should be related to the parameterized complexity of p-DFA-SW (Problem 4.8) for DFAs where the transformation monoid of the DFA is from a restricted class. For the non-parameterized version of p-DFA-SW, several NP-completeness results for some algebraic classes of transformation monoids (aperiodic monoids and commutative monoids among others) have been shown in Martyugin (2009).

A generalization of the parameterized factorization problem $\mathsf{pF}[\mathsf{C}]$ is the *parameterized rational subset membership problem* for the class of monoids C. The input consists of a monoid $M \in \mathsf{C}$, a finite (nondeterministic) automaton $\mathcal{A}$, whose transitions are labelled with elements from M and an additional element $a \in M$. The automaton $\mathcal{A}$ defines a subset $L(\mathcal{A}) \subseteq M$ in the natural way (take the set of all paths in $\mathcal{A}$ from an initial state to a final state and for each path multiply the M-labels of the transitions in the order given by the path) and the question is whether $a \in L(\mathcal{A})$. For the parameter we take the number of states of $\mathcal{A}$. Clearly, all lower bounds for $\mathsf{pF}[\mathsf{C}]$ (for every class C) are inherited by the parameterized rational subset membership problem for C. One might therefore investigate whether better lower bounds can be shown for the parameterized rational subset membership problem or whether there is an fpt-reduction from the parameterized rational subset membership problem for a class C to $\mathsf{pF}[\mathsf{C}]$. The non-parameterized rational subset membership problem for permutation groups was shown to be NP-complete in Khashaev (2022); Lohrey et al. (2022).

References

A. Abboud, K. Lewi, and R. Williams. Losing weight by gaining edges. In A. S. Schulz and D. Wagner, editors, *Algorithms - ESA 2014 - 22th Annual European Symposium, Wroclaw, Poland, September 8-10, 2014. Proceedings*, volume 8737 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 2014. doi: 10.1007/978-3-662-44777-2_1.

F. Bassino, I. Kapovich, M. Lohrey, A. Miasnikov, C. Nicaud, A. Nikolaev, I. Rivin, V. Shpilrain, A. Ushakov, and P. Weil. Discrete optimization in groups. In *Complexity and Randomness in Group Theory*, chapter 5. De Gruyter, 2020. doi: 10.1515/9783110667028-005.

J. F. Buss and T. Islam. Algorithms in the W-hierarchy. *Theory of Computing Systems*, 41(3):445–457, 2007. doi: 10.1007/S00224-007-1325-3.

L. Cai, J. Chen, R. G. Downey, and M. R. Fellows. On the parameterized complexity of short computation and factorization. *Archive for Mathematical Logic*, 36(4-5):321–337, 1997. doi: 10.1007/S001530050069.

Y. Chen, J. Flum, and M. Grohe. Machine-based methods in parameterized complexity theory. *Theoretical Computer Science*, 339(2-3):167–199, 2005. doi: 10.1016/J.TCS.2005.02.003.

R. G. Downey and M. R. Fellows. Fixed-parameter tractability and completeness II: on completeness for W[1]. *Theoretical Computer Science*, 141(1&2):109–131, 1995. doi: 10.1016/0304-3975(94)00097-3.

R. G. Downey and M. R. Fellows. *Parameterized Complexity*. Monographs in Computer Science. Springer, 1999. doi: 10.1007/978-1-4612-0515-9.

S. Even and O. Goldreich. The minimum-length generator sequence problem is NP-hard. *Journal of Algorithms*, 2(3):311–313, 1981. doi: 10.1016/0196-6774(81)90029-8.

H. Fernau and J. Bruchertseifer. Synchronizing words and monoid factorization, yielding a new parameterized complexity class? *Mathematical Structures in Computer Science*, 32(2):189–215, 2022. doi: 10.1017/S0960129522000184.

H. Fernau, P. Heggernes, and Y. Villanger. A multi-parameter analysis of hard problems on deterministic finite automata. *Journal of Computer and System Sciences*, 81(4):747–765, 2015. doi: 10.1016/J.JCSS.2014.12.027.

J. Flum and M. Grohe. *Parameterized Complexity Theory*. Texts in Theoretical Computer Science. An EATCS Series. Springer, 2006. doi: 10.1007/3-540-29953-X.

M. L. Furst, J. E. Hopcroft, and E. M. Luks. Polynomial-time algorithms for permutation groups. In *21st Annual Symposium on Foundations of Computer Science, Syracuse, New York, USA, 13-15 October 1980*, pages 36–41. IEEE Computer Society, 1980. doi: 10.1109/SFCS.1980.34.

S. Goebbels, F. Gurski, J. Rethmann, and E. Yilmaz. Change-making problems revisited: a parameterized point of view. *Journal of Combinatorial Optimization*, 34(4):1218–1236, 2017. doi: 10.1007/S10878-017-0143-Z.

P. Goralčík and V. Koubek. Rank problems for composite transformations. *International Journal of Algebra and Computation*, 5(3):309–316, 1995. doi: 10.1142/S0218196795000185.

S. Guillemot. Parameterized complexity and approximability of the longest compatible sequence problem. *Discrete Optimization*, 8(1):50–60, 2011. doi: 10.1016/J.DISOPT.2010.08.003.

C. S. Iliopoulos. On the computational complexity of the abelian permutation group structure, membership and intersection problems. *Theoretical Computer Science*, 56:211–222, 1988. doi: 10.1016/0304-3975(88)90078-3.

K. Ireland and M. Rosen. *A classical introduction to modern number theory, second edition*, volume 84 of *Graduate texts in mathematics*. Springer, 1990. doi: 10.1007/978-1-4757-2103-4.

M. Jerrum. The complexity of finding minimum-length generator sequences. *Theoretical Computer Science*, 36:265–289, 1985. doi: 10.1016/0304-3975(85)90047-7.

A. A. Khashaev. On the membership problem for finite automata over symmetric groups. *Discrete Mathematics and Applications*, 32(6):389–395, 2022. doi: doi:10.1515/dma-2022-0033.

M. Lohrey, A. Rosowski, and G. Zetzsche. Membership problems in finite groups. In S. Szeider, R. Ganian, and A. Silva, editors, *47th International Symposium on Mathematical Foundations of Computer Science, MFCS 2022, August 22-26, 2022, Vienna, Austria*, volume 241 of *LIPIcs*, pages 71:1–71:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022. doi: 10.4230/LIPICS.MFCS.2022.71.

P. Martyugin. Complexity of problems concerning reset words for some partial cases of automata. *Acta Cybernertica*, 19(2):517–536, 2009. URL `https://cyber.bibl.u-szeged.hu/index.php/actcybern/article/view/3780`.

P. McKenzie and S. A. Cook. The parallel complexity of abelian permutation group problems. *SIAM Journal on Computing*, 16(5):880–909, 1987. doi: 10.1137/0216058.

Á. Seress. *Permutation Group Algorithms*. Cambridge Tracts in Mathematics. Cambridge University Press, 2003. doi: 10.1017/CBO9780511546549.

P. R. Vaidyanathan. On the parameterized complexity of permutation group factorization and related problems. Master's thesis, Indian Institute of Technology Gandhinagar, Gujarat, India, 2019.

A Additional material on cyclic permutation groups

Lemma A.1. *There is an algorithm that given two permutations $\alpha, \beta \in S_n$ checks in polynomial time whether $\langle \alpha, \beta \rangle$ is cyclic and if so returns a permutation σ such that $\langle \alpha, \beta \rangle = \langle \sigma \rangle$ and*

$$\mathrm{lcm}(\mathrm{ord}(\alpha), \mathrm{ord}(\beta)) = \mathrm{ord}(\sigma).$$

Proof: On input $\alpha, \beta \in S_n$, the algorithm computes the following data:

1. $\mathrm{ord}(\alpha)$ and $\mathrm{ord}(\beta)$,
2. the prime factorizations $\mathrm{ord}(\alpha) = p_1^{a_1} \cdots p_m^{a_m}$ and $\mathrm{ord}(\beta) = p_1^{b_1} \cdots p_m^{b_m}$ (we assume here that a_i or b_i is non-zero for every $i \in [1, m]$),
3. the numbers $r = \prod_{i=1}^m p_i^{x_i}$ and $s = \prod_{i=1}^m p_i^{y_i}$, where for all $i \in [1, m]$ we have
$$x_i = \begin{cases} 0 & \text{if } a_i \geq b_i \\ a_i & \text{if } a_i < b_i \end{cases} \text{ and } y_i = \begin{cases} 0 & \text{if } b_i > a_i \\ b_i & \text{if } b_i \leq a_i \end{cases}$$
4. the permutations $\alpha' = \alpha^r$, $\beta' = \beta^s$, and $\gamma = \alpha'\beta'$.

If $\langle \alpha, \beta \rangle = \langle \gamma \rangle$ then the algorithm returns γ, otherwise it returns "not cyclic".

We first show that the algorithm is correct, then we argue that it runs in polynomial time. Clearly, if $\langle \alpha, \beta \rangle$ is not cyclic then the algorithm will return "not cyclic". Vice versa suppose there is a permutation σ such that $\langle \alpha, \beta \rangle = \langle \sigma \rangle$. Then we have $\mathrm{ord}(\sigma) = |\langle \alpha, \beta \rangle| = \mathrm{lcm}(\mathrm{ord}(\alpha), \mathrm{ord}(\beta))$, where lcm denotes the least common multiple. Now let α', β' and γ be as defined in the algorithm. Then $\mathrm{ord}(\alpha') = \mathrm{ord}(\alpha)/r$ and $\mathrm{ord}(\beta') = \mathrm{ord}(\beta)/s$ and these numbers are coprime. We obtain $\mathrm{ord}(\gamma) = \mathrm{ord}(\alpha')\,\mathrm{ord}(\beta') =$

$\mathrm{lcm}(\mathrm{ord}(\alpha), \mathrm{ord}(\beta)) = \mathrm{ord}(\sigma)$. Moreover $\gamma \in \langle \alpha, \beta \rangle = \langle \sigma \rangle$. By this it follows that γ is also a generator of $\langle \sigma \rangle$, i.e., $\langle \gamma \rangle = \langle \alpha, \beta \rangle$. Hence, checking $\langle \alpha, \beta \rangle = \langle \gamma \rangle$ succeeds and the algorithm correctly returns γ.

We now argue that the algorithm runs in polynomial time. Recall that n (the degree of the permutations) is given in unary notation. To compute the order of a permutation we simply have to compute the length of the disjoint cycles and compute the least common multiple of the lengths which can be computed in polynomial time. By Lagrange's theorem both $\mathrm{ord}(\alpha)$ and $\mathrm{ord}(\beta)$ divide $n!$ and thus $p_i \leq n$ for $i \in [1, m]$. Moreover $m \leq n$ because every prime p_i is bounded by n. Furthermore the exponents a_i and b_i are bounded by $\mathcal{O}(\log n)$ because every cycle has length at most n. Hence the prime factorizations of $\mathrm{ord}(\alpha)$ and $\mathrm{ord}(\beta)$ can be computed in polynomial time. Clearly α', β' and γ can be computed in polynomial time. Finally, in order to check $\langle \alpha, \beta \rangle = \langle \gamma \rangle$, it suffices to check whether $\alpha \in \langle \gamma \rangle$ and $\beta \in \langle \gamma \rangle$. This can be done in polynomial time by Furst et al. (1980). □

Lemma A.2. *There is an algorithm that given a list of permutations $\pi_1, \ldots, \pi_m \in S_n$ checks in polynomial time whether $\langle \pi_1, \ldots, \pi_m \rangle$ is cyclic and if so returns a permutation σ such that $\langle \pi_1, \ldots, \pi_m \rangle = \langle \sigma \rangle$.*

Proof: If $\langle \pi_1, \ldots, \pi_m \rangle$ is cyclic then also $\langle \pi_1, \ldots, \pi_l \rangle$ is cyclic for every $l \leq m$. Therefore, using Lemma A.1 we can first check whether $\langle \pi_1, \pi_2 \rangle$ is cyclic. If this is not the case, then $\langle \pi_1, \ldots, \pi_m \rangle$ is not cyclic. Otherwise, Lemma A.1 returns a permutation $\pi_{1,2}$ such that $\langle \pi_1, \pi_2 \rangle = \langle \pi_{1,2} \rangle$. We then recursively check whether $\langle \pi_{1,2}, \pi_3, \ldots, \pi_m \rangle$ is cyclic. □