THE SUBPOWER MEMBERSHIP PROBLEM FOR SEMIGROUPS

ANDREI BULATOV, MARCIN KOZIK, PETER MAYR, AND MARKUS STEINDL

Abstract. Fix a finite semigroup S and let $a_1, \dots, a_k, b$ be tuples in a direct power S^n. The subpower membership problem (SMP) asks whether b can be generated by $a_1, \dots, a_k$. If S is a finite group, then there is a folklore algorithm that decides this problem in time polynomial in nk. For semigroups this problem always lies in PSPACE. We show that the SMP for a full transformation semigroup on 3 or more letters is actually PSPACE-complete, while on 2 letters it is in P. For commutative semigroups, we provide a dichotomy result: if a commutative semigroup S embeds into a direct product of a Clifford semigroup and a nilpotent semigroup, then SMP(S) is in P; otherwise it is NP-complete.

1. Introduction

Deciding membership is a basic problem in computer algebra. For permutation groups given by generators, it can be solved in polynomial time using Sims' stabilizer chains [1]. For transformation semigroups, membership is PSPACE-complete by a result of Kozen [5].

In this paper we study a particular variation of the membership problem that was proposed by Willard in connection with the study of constraint satisfaction problems (CSP) [3, 10]. Fix a finite algebraic structure S with finitely many basic operations. Then the *subpower membership problem* (SMP) for S is the following decision problem:

SMP(S)
Input: $\{a_1, \dots, a_k\} \subseteq S^n, b \in S^n$
Problem: Is b in the subalgebra $\langle a_1, \dots, a_k \rangle$ of S^n generated by $\{a_1, \dots, a_k\}$?

For example, for a one-dimensional vector space S over a field F, SMP(S) asks whether a vector $b \in F^n$ is spanned by vectors $a_1, \dots, a_k \in F^n$.

Note that SMP(S) has a positive answer iff there exists a k-ary term function t on S such that $t(a_1, \dots, a_k) = b$, that is

$$t(a_{1i}, \dots, a_{ki}) = b_i \quad \text{for all} \quad i \in \{1, \dots, n\}. \tag{1}$$

Hence SMP(S) is equivalent to the following problem: Is the partial operation t that is defined on an n element subset of S^k by (1) the restriction of a term function on S?

Note that the input size of SMP(S) is essentially $n(k + 1)$. Since the size of $\langle a_1, \dots, a_k \rangle$ is limited by $|S|^n$, one can enumerate all elements in time exponential in n using a straightforward closure algorithm. This means that SMP(S) is in EXPTIME for each algebra S. Kozik constructed a class of algebras which actually have EXPTIME-complete subpower membership problems [6].

Date: August 11, 2016.
2000 *Mathematics Subject Classification.* Primary: 20M99; Secondary: 68Q25.
Key words and phrases. semigroup, direct power, membership problem.
The first author was supported by an NSERC Discovery grant, the second author was partially supported by the National Science Centre Poland: UMO-2014/13/B/ST6/01812, the last two by the Austrian Science Fund (FWF): P24285.

Still for certain structures the SMP might be considerably easier. For S a vector space, the SMP can be solved by Gaussian elimination in polynomial time. For groups the SMP is in P as well by an adaptation of permutation group algorithms [1, 11]. Even for certain generalizations of groups and quasigroups the SMP can be shown to be in P [7].

In the current paper we start the investigation of algorithms for the SMP of finite semigroups and its complexity. We will show that the SMP for arbitrary semigroups is in PSPACE in Theorem 2.1 For the full transformation semigroups T_n on n letters we will prove the following in Section 2.

Theorem 1.1. SMP(T_n) *is* PSPACE*-complete for all* $n \geq 3$*, while* SMP(T_2) *is in* P.

This is the first example of a finite algebra with PSPACE-complete SMP. As a consequence we can improve a result of Kozen from [5] on the intersection of regular languages in Corollary 2.4.

Moreover the following is the smallest semigroup and the first example of an algebra with NP-complete SMP.

Example 1.2. Let $Z_2^1 := \{0, a, 1\}$ denote the 2-element null semigroup adjoined with a 1, i.e., Z_2^1 has the following multiplication table:

Z_2^1	0	a	1
0	0	0	0
a	0	0	a
1	0	a	1

Then SMP(Z_2^1) is NP-complete. NP-hardness follows from Lemma 5.2 by encoding the exact cover problem. The NP-easiness for commutative semigroups is proved in Lemma 5.1.

Generalizing from this example we obtain the the following dichotomy for commutative semigroups.

Theorem 1.3. *Let S be a finite commutative semigroup. Then* SMP(S) *is in* P *if one of the following equivalent conditions holds:*

(1) *S is an ideal extension of a Clifford semigroup by a nilpotent semigroup;*
(2) *the ideal generated by the idempotents of S is a Clifford semigroup;*
(3) *for every idempotent $e \in S$ and every $a \in S$ where $ea = a$ the element a generates a group;*
(4) *S embeds into the direct product of a Clifford semigroup and a nilpotent semigroup.*

Otherwise SMP(S) *is* NP*-complete.*

Theorem 1.3 is proved in Section 5. Our way towards this result starts with describing a polynomial time algorithm for the SMP for Clifford semigroups in Section 4. In fact in Corollary 4.10 we will show that SMP(S) is in P for every (not necessarily commutative) ideal extension of a Clifford semigroup by a nilpotent semigroup.

Throughout the rest of the paper, we write $[n] := \{1, \dots, n\}$ for $n \in \mathbb{N}$. Also a tuple $a \in S^n$ is considered as a function $a \colon [n] \to S$. So the i-th coordinate of this tuple is denoted by $a(i)$ rather than a_i.

2. Full transformation semigroups

First we give an upper bound on the complexity of the subpower membership problem for arbitrary finite semigroups.

Theorem 2.1. *The* SMP *for a finite semigroup is in* PSPACE.

Proof. Let S be a finite semigroup. We show that

$$\text{SMP}(S) \text{ is in nondeterministic linear space.} \tag{2}$$

To this end, let $A \subseteq S^n$, $b \in S^n$ be an instance of SMP(S). If $b \in \langle A \rangle$, then there exist $a_1, \ldots, a_m \in A$ such that $b = a_1 \cdots a_m$.

Now we pick the first generator $a_1 \in A$ nondeterministically and start with $c := a_1$. Pick the next generator $a \in A$ nondeterministically, compute $c := c \cdot a$, and repeat until we obtain $c = b$. Clearly all computations can be done in space linear in $n \cdot |A|$. This proves (2). By a result of Savitch [9] this implies that SMP(S) is in deterministic quadratic space. □

Theorem 2.2. *SMP(T_3) is* PSPACE*-complete.*

Proof. Kozen [5] showed that the following decision problem is PSPACE-complete: input n and functions $f, f_1, \ldots, f_m : [n] \to [n]$ and decide whether f can be obtained as a composition[1] of f_i's. The size of the input for this problem is $(m+1)n \log n$.

To encode this problem into SMP(T_3) let T_3 be the full transformation semigroup of $0, 1$ and ∞. We identify g, an element of T_3, with the triple $(g(0), g(1), g(\infty))$ and name a number of elements of T_3:

- $\mathbf{0} = (0, 0, \infty)$ and $\mathbf{1} = (1, 1, \infty)$ are used to encode the functions $[n] \to [n]$;
- $\mathbf{id} = (0, 1, \infty), \mathbf{0} \mapsto \mathbf{0} = (0, \infty, \infty), \mathbf{0} \mapsto \mathbf{1} = (1, \infty, \infty)$ and $\mathbf{1} \mapsto \mathbf{0} = (\infty, 0, \infty)$ are used to model the composition.

We call an element of T_3 *bad* if it sends 0 or 1 to ∞; and we call a tuple of elements *bad* if it is bad on at least one position. Note that all the named elements send ∞ to ∞ so multiplying, on the right, a bad element by any of the named elements produces a bad element.

Let k and $f, f_1, \ldots f_m$ be an input to the Kozen's composition problem. We will encode it as SMP on $n^2 + mn$ positions. We start with an auxiliary notation. Every function $f' : [n] \to [n]$ can be encoded (on these $n^2 + nm$ positions) as follows: the first n positions are all $\mathbf{0}$ except for the $f'(1)$-th one which is $\mathbf{1}$, among positions $\{n+1, \ldots, 2n\}$ only $(n + f(1))$-th one is $\mathbf{1}$ and the rest is $\mathbf{0}$ and so on. Finally the remaining nm positions are all $\mathbf{0}$. We call such a tuple the *mapping tuple* for f'. Note that none of the mapping tuples are bad.

We introduce the generators of the subpower gradually. The first generator is the mapping tuple for the identity on $[n]$.

Next, for each f_i we add the *choice tuple* which is $\mathbf{id}$ on the first n^2 positions; is $\mathbf{0} \mapsto \mathbf{1}$ on positions $\{n^2 + (i-1)n + 1, \ldots, n^2 + (i-1)n + n\}$ and $\mathbf{0} \mapsto \mathbf{0}$ elsewhere. Multiplying (on the right) the mapping tuple for f' by the choice tuple for f_i, corresponds to deciding that f' will be composed with f_i.

Finally, for each f_i and $j, k \in [n]$ we add the *application tuple* with the semantics:

$$\text{apply } f_i \text{ on coordinate } j \text{ to } k$$

which is all $\mathbf{id}$ except for the position $n(j-1) + k$ which is $\mathbf{1} \mapsto \mathbf{0}$, position $n(j-1) + f_i(k)$ which is $\mathbf{0} \mapsto \mathbf{1}$ and position $n^2 + (i-1)n + j$ which is $\mathbf{1} \mapsto \mathbf{0}$. Multiplication by the application tuples computes the composition decided by the choice tuples.

It remains to choose an element which will be generated by all these tuples if and only if f is a composition of f_i's. This final element is the mapping tuple for f.

Lets analyze, from left to right, a product of the generators which produces a tuple which is not bad. Leftmost element of the product needs to be the mapping tuple of the identity – the only generator which itself is not bad.

The second element from the left can be the same, but then the product is the mapping tuple of identity again and we can disregard this case. The second element

[1] We will assume that the identity function can be obtained even from an empty set of functions. This little twist does not change the complexity of the problem.

cannot be an application tuple as the $\mathbf{1} \mapsto \mathbf{0}$ on one of the last nm positions would turn the result bad. Thus the only meaningful option is the choice tuple for some function f_i. Multiplication by this tuple turns n positions (among the last nm positions) of the mapping tuple for identity to $\mathbf{1}$.

Consider the next n elements of the product. Any one of them can be the mapping tuple of the identity, but then the product resets as any tuple which is not bad multiplied by the mapping tuple of the identity produces the mapping tuple of the identity. Yet again we disregard this case. None of the n elements can be a choice tuple: a multiplication by a choice tuple produces bad result unless the last nm positions of the left tuple are all $\mathbf{0}$.

Thus all n elements need to be application tuples, and focusing again on the last nm positions we find for each $j \in [n]$ exactly one tuple with semantics "apply f_i on coordinate j". Focusing on the first n^2 position we immediately get that the semantics needs to be "apply f_i on coordinate j to j".

It is easy to see that, after multiplying by these n tuples, we get the mapping tuple for f_i. Continuing the reasoning with the mapping tuple for f_i (instead of identity) and, say, f_j (instead of f_i) we get a mapping tuple for $f_j \circ f_i$ and so on. In the end we get a mapping tuple for f if and only if f can be obtained as a composition of the f_i's and the identity.

The number of tuples we input into SMP is $mn^2 + m + 2$, so the total size of the input is $C(mn^2 + m + 2)(n^2 + nm)$ which is polynomial with respect to the size of the input of the original problem. $\square$

The next theorem states that the bound it tight, that is that already for T_2 the SMP is solvable in a polynomial time.

Theorem 2.3. *SMP(T_2) is in* P.

Proof. Let the underlying set of T_2 be $\{0, 1\}$ and the constants of T_2 be denoted by $\mathbf{0}$ and $\mathbf{1}$ and the non-constants by **id** and **not**. For a tuple $a \in T_2^k$ the *constant part* (or **cp**) of a are the i's such that $a_i \in T_2$ is a constant, the *non-constant part* (or **ncp**) are the remaining i's.

Let $a_1, \dots, a_n, b \in T_2^k$ be the input of the SMP(T_2). Before starting the algorithm we preprocess the instance, by removing all the a_i's with **cp** not included in **cp** of b. It is clear that the removed tuples cannot participate in a product producing b. Next we call the function **SMP**$(a_1, \dots, a_n, b)$ from Algorithm 1.

The following easy observation is the key ingredient of the algorithm: if for some i the tuple a_i and b agree on the **cp** of a_i then the original problem is equivalent to **SMP**$(a'_1, \dots, a'_n, b')$ where the primed version of elements are obtained from the unprimed by projecting to **ncp** of a_i. Indeed we have $b' = a'_{i_1} \cdots a'_{i_l}$ if and only if $b = a_{i_1} \cdots a_{i_l} a_i a_i$ (note that we need a_i twice since we need to compose **not** with itself to obtain **id** on coordinates from **ncp** of a_i).

We analyze the Algorithm 1 line by line. Note that if b has empty **cp** then, by the preprocessing, each a_i has empty **cp** as well and the problem reduces to SMP over $\mathbb{Z}_2$ (which is solvable in P). This is the essence behind line 3 of the algorithm: the function returns **TRUE** in line 4 or ignores the loop starting in line 6 (as $l = n$) and returns **FALSE** in line 18.

If b has non-empty **cp** we reason almost exactly as in the easy observation above. The loop in line 6 verifies if, modulo multiplication by the tuples with empty **cp**, a_i can play the role of a_i from the easy observation. If so, i.e. the check in line 10 returned true, then the recursive call in line 15 is justified by the equivalence of the projected problem from the easy observation.

The remaining part of the proof is structure as follows: First we show that, if b can be obtained as product of a_i's, then the recursive call in line 15 will happen. Then we assume, by induction on the size of **cp** of b, that the recursive call returns

Algorithm 1
Function **SMP**$(a_1, \ldots, a_n, b)$ solving SMP(T_2).

```
Input: $a_1, \ldots, a_n, b \in T_2^k$
Output: TRUE or FALSE
 1: let $a_1, \ldots, a_l$ be the $a_i$'s with empty cp
 2: and $a_{l+1}, \ldots, a_n$ with non-empty cp
 3: if $b$ has empty cp and SMP$_{\mathbb{Z}_2}(a_1, \ldots, a_l, b)$ then
 4:     return TRUE
 5: end if
 6: for $i = l+1 \ldots n$ do
 7:         $\triangleright$ checks if $a_i$ can be the last element of the product with non-empty cp
 8:     let $a'_1, \ldots, a'_l$ be projections of $a_1, \ldots, a_l$ to cp of $a_i$
 9:     let $b'$ (defined on cp of $a_i$) be $b'(j) =$ id if $a_i(j) = b(j)$ and $b'(j) =$ not else
10:     if SMP$_{\mathbb{Z}_2}(a'_1, \ldots, a'_l, b')$ then
11:             $\triangleright$ we found $a_j$'s with empty cp's which change $a_i$ to $b$ on cp of $a_i$
12:         let $j_1, \ldots, j_m$ be the sequence of indices from the call of SMP$_{\mathbb{Z}_2}$
13:         set $b'' = ba_{j_m} \cdots a_{j_1}$
14:         let $a''_1, \ldots, a''_n, b'''$ be projections of $a_1, \ldots, a_n, b''$ to ncp of $a_i$
15:         return SMP$(a''_1, \ldots, a''_n, b''')$
16:     end if
17: end for
18: return FALSE
```

a correct answer to the projected problem. We conclude the proof by showing that it is the answer to the original problem as well.

To see the first fact let $b = a_{j_1} \cdots a_{j_m}$ and let a_{j_p} be the last element of the product with non-empty **cp**. The suffix $a_{j_{(p+1)}} \cdots a_{j_m}$ consists of elements of empty **cp** which multiply a_{j_p}, on its **cp**, to b. This means that the condition on line 10 will be satisfied for some i (maybe with $i = j_p$, but maybe with some other i).

It remains to show that the answer to the recursive call in line 15 is the same as the answer to the original instance. For one direction: if the recursive call in line 15, in the loop iteration at i, answers **TRUE** with indices $i_1, \ldots, i_p$ then

$$b = a_{i_1} \cdots a_{i_p} a_i a_i a_{j_1} \cdots a_{j_m}.$$

Indeed on indices from the **cp** of a_i only the last $m+2$ elements matter and they provide proper values by the choice of the sequence $j_1, \ldots, j_m$ computed by the algorithm. For the **ncp** of a_i the recursive call provides b''. Since $a_i a_i$ is **id** on **ncp** of a_i and $a_{j_m} \cdots a_{j_1} a_{j_1} \cdots a_{j_m}$ is a tuple of **id**'s (since all the tuples in the product have empty **cp**'s) we obtain b on **ncp** of a_i as well.

For the other direction: we show that if the original question was **TRUE** then the recursive call will answer **TRUE** as well. This would be the case if $i = j_p$ two paragraphs above but obtaining it directly would require backtracking. Fortunately if b is a product of a_i's then so is $b'' = ba_{j_m} \cdots a_{j_1}$ (for any sequence computed in a successful test in line 10) and b''' is just a projection of b''.

The complexity of the algorithm is clearly polynomial: The function **SMP** works in a polynomial time, and the depth of recursion is bounded by k as during each recursive call we loose at least one coordinate. □

For proving that membership for transformation semigroups is PSPACE-complete, Kozen first showed that the following decision problem is PSPACE-complete [5].

AUTOMATA INTERSECTION PROBLEM

Input:	deterministic finite state automata $F_1, \dots, F_n$ with common alphabet Σ
Problem:	Is there a word in Σ^* that is accepted by all of $F_1, \dots, F_n$?

Using the wellknown connection between automata and transformation semigroups we obtain the following stronger version of Kozen's result

Corollary 2.4. *The Automata Intersection Problem restricted to automata with* 3 *states is* PSPACE*-complete.*

Proof. The Automata Intersection Problem is in PSPACE by [5]. For PSPACE-hardness we adapt our proof of Theorem 2.2. Lets fix $n, f, f_1, \dots, f_m$ as in the proof Theorem 2.2.

To imitate the tuples of elements in T_3 (which appeared in the proof) we introduce, for each position $i \leq n^2 + nm$, three automata: F_i^0, F_i^1 and F_i^∞ each with the set of states $\{0, 1, \infty\}$. These automata are responsible for storing the result of the function, on position i, on $0, 1$ and ∞ respectively.

The initial state of the automaton F_i^j is obtained by applying to j the function on the i-th position of the mapping tuple for the identity. The accepting state is obtained similarly from the mapping tuple for f. The language of the automata consists of all the generators of the subpower: on a letter (which is a mapping, a choice or an application tuple) the automat F_i^j changes state according to the function on the i-th position in the tuple.

It is clear that all the $3n^2 + 3nm$ automata accept a common word if, and only if, the mapping tuple for the identity can be multiplied on the right to a mapping tuple for f. This is exactly what we needed in the proof of Theorem 2.2 and thus the Automata Intersection Problem for automata with 3 states is PSPACE-hard. □

3. Nilpotent semigroups

Definition 3.1. A semigroup S is called *d-nilpotent* for $d \in \mathbb{N}$ if

$$\forall x_1, \dots, x_d, y_1, \dots, y_d \in S \colon x_1 \cdots x_d = y_1 \cdots y_d.$$

It is called *nilpotent* if it is d-nilpotent for some $d \in \mathbb{N}$. We let $0 := x_1 \cdots x_d$ denote the zero element of a d-nilpotent semigroup S.

Definition 3.2. An *ideal extension* of a semigroup I by a semigroup Q with zero is a semigroup S such that I is an ideal of S and the Rees quotient semigroup S/I is isomorphic to Q.

Theorem 3.3. *Let T be an ideal extension of a semigroup S by a d-nilpotent semigroup N. Then Algorithm 2 reduces* SMP(T) *to* SMP(S) *in polynomial time.*

Proof. Correctness of Algorithm 2. Let $A \subseteq T^n$, $b \in T^n$ be an instance of SMP(T).

Case $b \notin S^n$. Since T/S is d-nilpotent, a product that is equal to b cannot have more than $d-1$ factors. Thus Algorithm 2 verifies in lines 2 to 8 whether there are $\ell < d$ and $a_1, \dots, a_\ell \in A$ such that $b = a_1 \cdots a_\ell$. In line 5, Algorithm 2 returns true if such factors exist. Otherwise false is returned in line 9.

Case $b \in S^n$. Let B be as defined in line 11. We claim that

$$b \in \langle A \rangle \text{ iff } b \in \langle B \rangle. \tag{3}$$

The "if"-direction is clear. For the converse implication assume $b \in \langle A \rangle$. Then we have $\ell \in \mathbb{N}$ and $a_1, \dots, a_\ell \in A$ such that $b = a_1 \cdots a_\ell$. If $\ell < 2d$, then $b \in B$ and we are done. Assume $\ell \geq 2d$ in the following. Let $q \in \mathbb{N}$ and $r \in \{0, \dots, d-1\}$

Algorithm 2
Reduce SMP(T) to SMP(S) for an ideal extension T of S by d-nilpotent N.

```
Input: $A \subseteq T^n$, $b \in T^n$.
Output: Is $b \in \langle A \rangle$?
 1: if $b \notin S^n$ then
 2:     for $\ell \in [d-1]$ do
 3:         for $a_1, \dots, a_\ell \in A$ do
 4:             if $b = a_1 \cdots a_\ell$ then
 5:                 return true
 6:             end if
 7:         end for
 8:     end for
 9:     return false
10: else
11:     $B := \{a_1 \cdots a_k \in S^n \mid k < 2d, a_1, \dots, a_k \in A\}$
12:     return $b \in \langle B \rangle$                    ▷ instance of SMP($S$)
13: end if
```

such that $\ell = qd + r$. For $0 \leq j \leq q-2$ define $b_j := a_{jd+1} \cdots a_{jd+d}$. Further $b_{q-1} := a_{(q-1)d+1} \cdots a_\ell$. Since T/S is d-nilpotent, any product of d or more elements from A is in S^n. In particular $b_0, \dots, b_{q-1}$ are in B. Since

$$b = b_0 \cdots b_{q-1},$$

we obtain $b \in \langle B \rangle$. Hence (3) is proved.

Since Algorithm 2 returns $b \in \langle B \rangle$ in line 12, its correctness follows from (3).

Complexity of Algorithm 2. In lines 2 to 8, the computation of each product $a_1 \cdots a_\ell$ requires $n(\ell - 1)$ multiplications in S. There are $|A|^\ell$ such products of length ℓ. Thus the number of multiplications in S is at most $\sum_{\ell=2}^{d-1} n(\ell-1)|A|^\ell$. This expression is bounded by a polynomial of degree $d-1$ in the input size $n(|A|+1)$.

Similarly the size of B and the effort for computing its elements is bounded by a polynomial of degree $2d-1$ in $n(|A|+1)$. Hence Algorithm 2 runs in polynomial time. □

Corollary 3.4. *The* SMP *for every finite nilpotent semigroup is in* P.

Proof. Immediate from Theorem 3.3 □

4. Clifford semigroups

Clifford semigroups are also known as semilattices of groups. In this section we show that their SMP is in P. First we state some well-known facts on Clifford semigroups and establish some notation.

Lemma 4.1 (cf. [2, p. 12, Proposition 1.2.3])**.** *In a finite semigroup S, each $s \in S$ has an* idempotent power *s^m for some $m \in \mathbb{N}$, i.e., $(s^m)^2 = s^m$.*

Definition 4.2. A semigroup S is *completely regular* if every $s \in S$ is contained in a subsemigroup of S which is a also a group. A semigroup S is a *Clifford semigroup* if it is completely regular and its idempotents are central. The latter condition may be expressed by

$$\forall e, s \in S \colon (e^2 = e \Rightarrow es = se).$$

Definition 4.3. Let $\langle I, \wedge \rangle$ be a semilattice. For $i \in I$ let $\langle G_i, \cdot \rangle$ be a group. For $i, j, k \in I$ with $i \geq j \geq k$ let $\phi_{i,j} \colon G_i \to G_j$ be group homomorphisms such that $\phi_{j,k} \circ \phi_{i,j} = \phi_{i,k}$ and $\phi_{i,i} = \mathrm{id}_{G_i}$. Let $S := \dot{\bigcup}_{i \in I} G_i$, and

$$\text{for} \quad x \in G_i,\ y \in G_j \quad \text{let} \quad x * y := \phi_{i,i\wedge j}(x) \cdot \phi_{j,i\wedge j}(y).$$

Then we call $\langle S, *\rangle$ a *strong semilattice of groups.*

Theorem 4.4 (Clifford, cf. [2, p. 106–107, Theorem 4.2.1])**.** *A semigroup is a strong semilattice of groups iff it is a Clifford semigroup.*

Note that the operation $*$ extends the multiplication of G_i for each $i \in I$. It is easy to see that $\{G_i \mid i \in I\}$ are precisely the maximal subgroups of S. Moreover, each Clifford semigroup inherits a preorder $\leq$ from the underlying semilattice.

Definition 4.5. Let S be a Clifford semigroup constructed from a semilattice I and disjoint groups G_i for $i \in I$ as in Definition 4.3. For $x, y \in S$ define

$$x \leq y \quad \text{if} \quad \exists i, j \in I \colon i \leq j, x \in G_i, y \in G_j.$$

Lemma 4.6. *Let S be a Clifford semigroup and $x, y, z \in S$. Then*

1. *$x \leq yz$ iff $x \leq y$ and $x \leq z$,*
2. *$xyz \leq y$, and*
3. *$x \leq y$ and $y \leq x$ iff x and y are in the same maximal subgroup of S.*

Proof. Straightforward. □

The following mapping will help us solve the SMP for Clifford semigroups.

Definition 4.7. Let S be a finite Clifford semigroup constructed from a semilattice I and disjoint groups G_i for $i \in I$ as in Definition 4.3. Let

$$\gamma \colon S \to \prod_{i \in I} G_i \quad \text{such that} \quad \gamma(s)(i) := \begin{cases} s & \text{if } s \in G_i, \\ 1_{G_i} & \text{otherwise} \end{cases}$$

for $s \in S$ and $i \in I$.

Here $\prod$ denotes the direct product and 1_{G_i} the identity of the group G_i for $i \in I$. Note that the mapping γ is not necessarily a homomorphism.

Algorithm 3
For a Clifford semigroup $S = \dot{\bigcup}_{i \in I} G_i$, reduce SMP$(S)$ to SMP$(\prod_{i \in I} G_i)$.

```
Input: $A \subseteq S^n$, $b \in S^n$.
Output: True if $b \in \langle A \rangle$, false otherwise.
1: Set $\{a_1, \dots, a_k\} := \{a \in A \mid \forall i \in [n] \colon a(i) \geq b(i)\}$
2: Set $e$ to the idempotent power of $b$.
3: if $\exists i \in [n] \colon e(i) \notin \langle a_1(i), \dots, a_k(i) \rangle$ then
4:     return false
5: end if
6: return $\gamma(b) \in \langle \gamma(a_1 e), \dots, \gamma(a_k e) \rangle$        ▷ instance of SMP($\prod_{i \in I} G_i$)
```

Theorem 4.8. *Let S be a finite Clifford semigroup with maximal subgroups G_i for $i \in I$. Then Algorithm 3 reduces* SMP(S) *to* SMP$(\prod_{i \in I} G_i)$ *in polynomial time. The latter is the* SMP *of a group.*

Proof. Correctness of Algorithm 3. Assume $S = \langle \dot{\bigcup}_{i \in I} G_i, \cdot \rangle$ as in Definition 4.3. Fix an instance $A \subseteq S^n$, $b \in S^n$ of SMP(S). Let $a_1, \dots, a_k$ be as defined in line 1 of Algorithm 3.

First we claim that

$$b \in \langle A \rangle \quad \text{iff} \quad b \in \langle a_1, \dots, a_k \rangle. \tag{4}$$

To this end, assume that $b = c_1 \cdots c_m$ for $c_1, \dots, c_m \in A$. Fix $j \in [m]$. Lemma 4.6(1) implies that $b(i) \leq c_j(i)$ for all $i \in [n]$. Thus $c_j \in \{a_1, \dots, a_k\}$. Since j was arbitrary, we have $c_1, \dots, c_m \in \{a_1, \dots, a_k\}$ and (4) follows.

Let e be the idempotent power of b. If the condition in line 3 of Algorithm 3 is fulfilled, then neither e nor b are in $\langle a_1, \dots, a_k\rangle$. In this case false is returned in line 4. Now assume the condition in line 3 is violated, i.e.,

$$\forall i \in [n] \colon e(i) \in \langle a_1(i), \dots, a_k(i)\rangle.$$

We claim that

$$e \in \langle a_1, \dots, a_k\rangle. \tag{5}$$

For each $i \in [n]$ let $d_i \in \langle a_1, \dots, a_k\rangle$ such that $d_i(i) = e(i)$. Further let f be the idempotent power of $d_1 \cdots d_n$. We show $f = e$. Fix $i \in [n]$. Since $d_i(i) = e(i)$, we have $f(i) \leq e(i)$ by Lemma 4.6(2). On the other hand, $e(i) \leq b(i) \leq a_j(i)$ for all $j \leq k$. Hence $e(i) \leq f(i)$ by multiple applications of Lemma 4.6(1). Thus $f(i)$ and $e(i)$ are idempotent and are in the same group by Lemma 4.6(3). So $e(i) = f(i)$. This yields $e = f$ and thus (5) holds.

Next we show

$$b \in \langle a_1, \dots, a_k\rangle \quad \text{iff} \quad b \in \langle a_1 e, \dots, a_k e\rangle. \tag{6}$$

If $b = c_1 \cdots c_m$ for $c_1, \dots, c_m \in \{a_1, \dots, a_k\}$, then $b = be = c_1 \cdots c_m e = (c_1 e) \cdots (c_m e)$ since idempotents are central in Clifford semigroups. This proves (6).

Next we claim that

$$b \in \langle a_1 e, \dots, a_k e\rangle \quad \text{iff} \quad \gamma(b) \in \langle \gamma(a_1 e), \dots, \gamma(a_k e)\rangle. \tag{7}$$

Fix $i \in [n]$. By Lemma 4.6(3) the elements $a_1 e(i), \dots, a_k e(i)$, and $b(i)$ all lie in the same group, say G_l. Note that $\gamma|_{G_l} \colon G_l \to \prod_{i\in I} G_i$ is a semigroup monomorphism. This means that the componentwise application of γ to $\langle a_1 e, \dots, a_k e, b\rangle$, namely

$$\gamma|_{\langle a_1 e, \dots, a_k e, b\rangle} \colon \langle a_1 e, \dots, a_k e, b\rangle \to (\prod_{i\in I} G_i)^n,$$

is also a semigroup monomorphism. This implies (7).

In line 6, the question whether $\gamma(b) \in \langle \gamma(a_1 e), \dots, \gamma(a_k e)\rangle$ is an instance of $\mathrm{SMP}(\prod_{i\in I} G_i)$, which is the SMP of a group. By (4), (6) and (7), Algorithm 3 returns true iff $b \in \langle A\rangle$.

Complexity of Algorithm 3. Line 1 requires at most $\mathcal{O}(n|A|)$ calls of the relation $\leq$. For line 2, let $(s_1, \dots, s_{|S|})$ be a list of the elements of S and let $v \in \mathbb{N}$ minimal such that $(s_1, \dots, s_{|S|})^v$ is idempotent. Then $e = b^v$. Since v only depends on S but not on n or $|A|$, computing e takes $\mathcal{O}(n)$ steps. Line 3 requires $\mathcal{O}(n|A|)$ steps. Altogether the time complexity of Algorithm 3 is $\mathcal{O}(n|A|)$. □

Corollary 4.9. *The* SMP *for finite Clifford semigroups is in* P.

Proof. Let S be a finite Clifford semigroup. Fix an instance $A \subseteq S^n$, $b \in S^n$ of $\mathrm{SMP}(S)$. Algorithm 3 converts this instance into one of the SMP of a group with maximal size of $|S|^{|S|}$ in $\mathcal{O}(n|A|)$ time. Both instances have input size $n(|A| + 1)$. The latter can be solved by Willard's modification [10] of the concept of strong generators, known from the permutation group membership problem [1]. This requires $\mathcal{O}(n^3 + n|A|)$ time according to [11, p. 53, Theorem 3.4]. Hence $\mathrm{SMP}(S)$ is decidable in $\mathcal{O}(n^3 + n|A|)$ time. □

Corollary 4.10. *Let S be a finite ideal extension of a Clifford semigroup by a nilpotent semigroup. Then* $\mathrm{SMP}(S)$ *is in* P.

Proof. By Theorem 3.3 and Corollary 4.9. □

In the next lemma we give some conditions equivalent to the fact that a semigroup is an ideal extension of a Clifford semigroup by a nilpotent semigroup.

Lemma 4.11. *Let S be a finite semigroup. Then the following are equivalent:*

(1) *S is an ideal extension of a Clifford semigroup C by a nilpotent semigroup N;*
(2) *the ideal I generated by the idempotents of S is a Clifford semigroup;*
(3) *all idempotents in S are central, and for every idempotent $e \in S$ and every $a \in S$ where $ea = a$ the element a generates a group;*
(4) *S embeds into the direct product of a Clifford semigroup C and a nilpotent semigroup N.*

Proof. (1) $\Rightarrow$ (2): We show $I = C$. Since $S \setminus C$ cannot contain idempotent elements, all idempotents are in the ideal C. Thus we have $I \subseteq C$. Now let $c \in C$. Let $e \in I$ be the idempotent power of c. Then $c = ce \in I$. So $C \subseteq I$.

(2) $\Rightarrow$ (3): First we claim that all idempotents are central in S. To this end, let $e \in S$ be idempotent and $a \in S$. Then

$$\begin{aligned} ae &= (ae)e \\ &= e(ae) && \text{since } e, ae \in I \text{ and } e \text{ is central in } I, \\ &= (ea)e \\ &= e(ea) && \text{since } e, ea \in I \text{ and } e \text{ is central in } I, \\ &= ea. \end{aligned}$$

Next assume that $ea = a$. Since $ea \in I$, we have that $\langle a \rangle = \langle ea \rangle$ is a group.

(3) $\Rightarrow$ (4): Let $k \in \mathbb{N}$ such that x^k is idempotent for each $x \in S$. For $x \in S$ and an idempotent $e \in S$ we have

$$ex = (ex)^{k+1} = ex^{k+1} \tag{8}$$

since $\langle ex \rangle$ is a group and idempotents are central. We claim that

$$\alpha\colon S \to S,\ x \mapsto x^{k+1} \quad \text{is a homomorphism with} \quad \alpha^2 = \alpha. \tag{9}$$

For $x, y \in S$,

$$\begin{aligned} (xy)^{k+1} &= (xy)^k xy \\ &= (xy)^k x^{k+1} y && \text{by (8) since } (xy)^k \text{ is idempotent,} \\ &= (xy)^k x^{k+1} y^{k+1} && \text{by (8) since } x^k \text{ is idempotent,} \\ &= (xy)^{k+1} x^k y^k && \text{since } x^k, y^k \text{ are central,} \\ &= xyx^k y^k && \text{by (8) since } x^k \text{ is idempotent,} \\ &= x^{k+1} y^{k+1} && \text{since } x^k, y^k \text{ are central.} \end{aligned}$$

Also,

$$(x^{k+1})^{k+1} = x^{k^2+2k+1} = x^{k+1}.$$

This proves (9). Let $C := \alpha(S)$. We claim that C is an ideal. For $x, y \in S \cup \{1\}$ and $z^{k+1} \in C$,

$$\begin{aligned} xz^{k+1}y &= xzyz^k && \text{since } z^k \text{ is central,} \\ &= (xzy)^{k+1} z^k && \text{by (8),} \\ &= (xz^{k+1}y)^{k+1} && \text{since } z^k \text{ is central and idempotent,} \\ &\in C. \end{aligned}$$

Now consider the Rees quotient $N := S/C$. We claim that

$$N \text{ is } |N|\text{-nilpotent.} \tag{10}$$

Let $n_1, \ldots, n_{|N|} \in S$. First assume

$$\exists i, j \in \{1, \ldots, |N|\},\ i < j\colon n_1 \cdots n_i = n_1 \cdots n_j. \tag{11}$$

Then $n_{i+1}\cdots n_j$ is a right identity of $n_1\cdots n_i$. Thus

$$n_1\cdots n_i = n_1\cdots n_i(n_{i+1}\cdots n_j)^{k+1} \in C$$

since C is an ideal. So $n_1\cdots n_{|N|} \in C$.

If (11) does not hold, then $n_1, n_1n_2, \ldots, n_1\cdots n_{|N|}$ are $|N|$ distinct elements and at least one of them is in C. Again $n_1\cdots n_{|N|} \in C$ by the ideal property of C. This proves (10). Now let

$$\beta\colon S \to C \times N,\ s \mapsto (\alpha(s), s/C).$$

Apparently β is a homomorphism. It remains to prove that β is injective. Assume $\beta(x) = \beta(y)$ for $x, y \in S$. If $x \notin C$, then also $y \notin C$. Now $x/C = y/C$ implies $x = y$. Assume $x \in C$. Then $x = \alpha(x) = \alpha(y) = y$ since $\alpha^2 = \alpha$. We proved item (4) of Lemma 4.11.

(4) $\Rightarrow$ (1): Assume $S \leq C \times N$. Then $J := S \cap (C \times \{0\})$ is an ideal of S. At the same time J is a subsemigroup of a Clifford semigroup. By Definition 4.2 also J is a Clifford semigroup. It is easy to see that the Rees quotient $N_1 := S/J$ is nilpotent. Thus S is an ideal extension of the Clifford semigroup J by the nilpotent semigroup N_1. □

5. Commutative semigroups

The main result of Section 4 was that ideal extensions of Clifford semigroups by nilpotent semigroups have the SMP in P. In this section we show that if a commutative semigroup does not have this property, then its SMP is NP-complete. This will complete the proof of our dichotomy result, Theorem 1.3.

First we give an upper bound on the complexity of the SMP for commutative semigroups.

Lemma 5.1. *The* SMP *for a finite commutative semigroup is in* NP.

Proof. Let $\{a_1, \ldots, a_k\} \subseteq S^n$, $b \in S^n$ be an instance of SMP(S). Let $x := (s_1, \ldots, s_{|S|})$ be a list of all elements of S, and $r := |\langle x\rangle|$. Now $\langle x\rangle = \{x^1, \ldots, x^r\}$, and for each $\ell \in \mathbb{N}$ there is some $m \in [r]$ such that $x^\ell = x^m$. Since x contains all elements of S, we have

$$\forall y \in S^n\, \forall \ell \in \mathbb{N}\, \exists m \in [r]\colon y^\ell = y^m.$$

If $b \in \langle a_1, \ldots, a_k\rangle$, then there is a witness $(\ell_1, \ldots, \ell_k) \in \{0, \ldots, r\}^k$ such that $b = {a_1}^{\ell_1}\cdots {a_k}^{\ell_k}$. The size of this witness is $\mathcal{O}(k\log(r))$. Note that r depends only on S and not on the input size $n(k+1)$. Given $\ell_1, \ldots, \ell_k$ we can verify $b = {a_1}^{\ell_1}\cdots {a_k}^{\ell_k}$ in time polynomial in $n(k+1)$. Hence SMP(S) is in NP. □

Lemma 5.2. *Let S be a finite semigroup, $e \in S$ be idempotent, and $a \in S$. Assume that $ea = ae = a$ and $\langle a\rangle$ is not a group. Then* SMP(S) *is* NP*-hard.*

Proof. We reduce EXACT COVER to SMP(S). The former is one of Karp's 21 NP-complete problems [4].

EXACT COVER

Input:	$n \in \mathbb{N}$, sets $C_1, \ldots, C_k \subseteq [n]$
Problem:	Are there disjoint sets $D_1, \ldots, D_m \in \{C_1, \ldots, C_k\}$ such that $\bigcup_{i=1}^m D_i = [n]$?

Fix an instance $n, C_1, \ldots, C_k$ of EXACT COVER. Now we define characteristic functions $c_1, \ldots, c_k, b \in S^n$ for $C_1, \ldots, C_k, [n]$, respectively. For $j \in [k]$, $i \in [n]$, let

$$b(i) := a \quad\text{and}\quad c_j(i) := \begin{cases} a & \text{if } i \in C_j, \\ e & \text{otherwise.} \end{cases}$$

Now let $\{c_1, \dots, c_k\} \subseteq S^n$, $b \in S^n$ be an instance of SMP(S). We claim that

$$b \in \langle c_1, \dots, c_k \rangle \quad \text{iff} \quad \exists \text{ disjoint } D_1, \dots, D_m \in \{C_1, \dots, C_k\} \colon \bigcup_{i=1}^{m} D_i = [n].$$

"$\Rightarrow$": Let $d_1, \dots, d_m \in \{c_1, \dots, c_k\}$ such that $b = d_1 \cdots d_m$. Let $D_1, \dots, D_m$ be the sets corresponding to $d_1, \dots, d_m$, respectively. Then $\bigcup_{i=1}^{m} D_i = [n]$. The union is disjoint since $a \notin \{a^2, a^3, \dots\}$.

"$\Leftarrow$": Fix $D_1, \dots, D_m$ whose disjoint union is $[n]$. Let $d_1, \dots, d_m \in \{c_1, \dots, c_k\}$ be the characteristic functions of $D_1, \dots, D_m$, respectively. Then $b = d_1 \cdots d_m$. □

Corollary 5.3. *Let S be a finite commutative semigroup that does not fulfill one of the equivalent conditions of Lemma 4.11. Then* SMP(S) *is* NP*-hard.*

Proof. The semigroup S violates condition (3) of Lemma 4.11. Since the idempotents are central in S, there are $e \in S$ idempotent and $a \in S$ such that $ea = ae = a$ and $\langle a \rangle$ is not a group. Now the result follows from Lemma 5.2. □

Now we are ready to prove our dichotomy result for commutative semigroups.

Proof of Theorem 1.3. The conditions in Theorem 1.3 are the ones from Lemma 4.11 adapted to the commutative case. Thus they are equivalent. If one of them is fulfilled, then SMP(S) is in P by Corollary 4.10.

Now assume the conditions are violated. Then SMP(S) is NP-complete by Lemma 5.1 and Corollary 5.3. □

6. Conclusion

We showed that the SMP for finite semigroups is always in PSPACE and provided examples of semigroups S for which SMP(S) is in P, NP-complete, PSPACE-complete, respectively. For the SMP of commutative semigroups we obtained a dichotomy between the NP-complete and polynomial time solvable cases. Further we showed that the SMP for finite ideal extensions of a Clifford semigroup by a nilpotent semigroup is in P. For non-commutative semigroups there are several open problems.

Problem 6.1. Is the SMP for every finite semigroup either in P, NP-complete or PSPACE-complete?

Bands (idempotent semigroups) are well-studied. Still we do not know the following:

Problem 6.2. What is the complexity of the SMP for finite bands? More generally, what is the complexity in case of completely regular semigroups?

References

[1] M. Furst, J. Hopcroft, and E. Luks. Polynomial-time algorithms for permutation groups. In *Foundations of Computer Science, 1980., 21st Annual Symposium on*, pages 36–41, Oct 1980.

[2] J. Howie. *Fundamentals of Semigroup Theory.* Clarendon Oxford University Press, 1995.

[3] P. Idziak, P. Marković, R. McKenzie, M. Valeriote, and R. Willard. Tractability and learnability arising from algebras with few subpowers. *SIAM J. Comput.*, 39(7):3023–3037, 2010.

[4] R. M. Karp. Reducibility among combinatorial problems. In R. E. Miller, J. W. Thatcher, and J. D. Bohlinger, editors, *Complexity of Computer Computations*, The IBM Research Symposia Series, pages 85–103. Springer US, 1972.

[5] D. Kozen. Lower bounds for natural proof systems. In *18th Annual Symposium on Foundations of Computer Science (Providence, R.I., 1977)*, pages 254–266. IEEE Comput. Sci., Long Beach, Calif., 1977.

[6] M. Kozik. A finite set of functions with an EXPTIME-complete composition problem. *Theoretical Computer Science*, 407(1–3):330–341, 2008.

[7] P. Mayr. The subpower membership problem for Mal'cev algebras. *International Journal of Algebra and Computation*, 22(07):1250075, 2012.

[8] C. H. Papadimitriou. *Computational complexity.* Addison-Wesley Publishing Company, Reading, MA, 1994.

[9] W. J. Savitch. Relationships between nondeterministic and deterministic tape complexities. *J. Comput. System. Sci.*, 4:177–192, 1970.

[10] R. Willard. Four unsolved problems in congruence permutable varieties. Talk at International Conference on Order, Algebra, and Logics, Vanderbilt University, Nashville (June 12–16, 2007), 2007.

[11] S. Zweckinger. Computing in direct powers of expanded groups. Master's thesis, Johannes Kepler Universität Linz, Austria, 2013.

(Andrei Bulatov) School of Computing Science, Simon Fraser University, Burnaby BC, Canada
E-mail address: `andrei.bulatov@gmail.com`

(Marcin Kozik) Theoretical Computer Science, Faculty of Mathematics and Computer Science, Jagiellonian University, Poland
E-mail address: `marcin.kozik@uj.edu.pl`

(Peter Mayr) Institute for Algebra, Johannes Kepler University Linz, Austria
E-mail address: `peter.mayr@jku.at`

(Markus Steindl) Institute for Algebra, Johannes Kepler University Linz, Austria
E-mail address: `markus.steindl_1@jku.at`