THE FROBENIUS AND FACTOR UNIVERSALITY PROBLEMS OF THE KLEENE STAR OF A FINITE SET OF WORDS

MAKSYMILIAN MIKA AND MAREK SZYKUŁA

Abstract. We solve open problems concerning the Kleene star L^* of a finite set L of words over an alphabet Σ. The *Frobenius monoid* problem is the question for a given finite set of words L, whether the language L^* is cofinite. We show that it is PSPACE-complete. We also exhibit an infinite family of sets L such that the length of the longest words not in L^* (when L^* is cofinite) is exponential in the length of the longest words in L and subexponential in the sum of the lengths of words in L. The *factor universality* problem is the question for a given finite set of words L, whether every word over Σ is a factor (substring) of some word from L^*. We show that it is also PSPACE-complete. Besides that, we exhibit an infinite family of sets L such that the length of the shortest words not being a factor of any word in L^* is exponential in the length of the longest words in L and subexponential in the sum of the lengths of words in L. This essentially settles in the negative the longstanding Restivo's conjecture (1981) and its weak variations. All our solutions base on one shared construction, and as an auxiliary general tool, we introduce the concept of *set rewriting systems*. Finally, we complement the results with upper bounds.

Keywords: cofinite language, complete set, completable word, factor universality, finite list of words, incompletable word, Frobenius monoid, Kleene star, mortality, Restivo's conjecture, universality

1. Introduction

Given a set of words L over a finite alphabet Σ, the language L^* contains all finite strings built by concatenating any number of words from L. In general, we can think of L as a dictionary and L^* as the language of all available phrases. One of the most basic questions that one could ask is whether L generates all words over the alphabet Σ. The answer is, however, trivial, because this is the case if and only if L contains all single letters $a \in \Sigma$. Thus, more useful relaxed questions are considered. In this paper, we consider two classical such problems, settling their computational complexity and solving the related combinatorial questions.

Let $\|L\|_1$ denote the sum of the lengths of the words in L, and let $\|L\|_\infty$ denote the maximum length of the words in L. The value $\|L\|_1$ can be treated as the size of the input. Note that $\|L\|_1$ can be exponentially larger than $\|L\|_\infty$. We consider complexity and bounds in terms of both values.

1.1. **Frobenius monoid problem.**

The classical Frobenius problem is, for given positive integers $x_1, \ldots, x_k$, to determine the largest integer x that is not expressible as a non-negative linear combination of them. An integer x is expressible as a non-negative linear combination if there are integers $c_1, \ldots, c_k \geq 0$ such that $x = c_1x_1 + \ldots + c_kx_k$. In a decision version of the problem, we ask whether the largest integer exists, i.e., whether the set of non-expressible positive integers is finite. It is well known that the answer is "yes" if and only if $\gcd(x_1, \ldots, x_k) = 1$.

Institute of Computer Science, University of Wrocław, Wrocław, Poland
E-mail addresses: mika.maksymilian@gmail.com, msz@cs.uni.wroc.pl.

The Frobenius problem was extensively studied and found applications across many fields, e.g., to primitive sets of matrices [10], the Shellsort algorithm [13], and counting points in polytopes [2]. The problem of computing the largest non-expressible integer is NP-hard [21] when the integers are given in binary, and it can be solved polynomially if the number k of given integers is fixed [15].

A generalization of the Frobenius problem to the setting of languages was introduced by Kao, Shallit, and Xu [16]. Instead of a finite set of integers, we are given a finite set of words over some finite alphabet Σ, and instead of multiplication, we have the usual word concatenation. The original question becomes whether all but a finite number of words can be expressed as a concatenation of the words from the given set. If L is our given finite language, then the problem is equivalent to deciding whether L^* is cofinite, i.e., the complement of L^* is finite.

Problem 1.1 (Frobenius Monoid Problem for a Finite Set of Words)**.** *Given a finite set of words L over a finite non-empty alphabet Σ, is L^* cofinite?*

It is a simple observation that, if Σ is a unary alphabet, then Problem 1.1 is equivalent to the original Frobenius problem on integers, thus it is polynomially solvable. There are also efficient algorithms for checking whether a *given* word is in L^* [9].

Example 1.1. The language $L = \{000, 00000\}$ over $\Sigma = \{0\}$ generates the cofinite language L^*; since $\gcd(3, 5) = 1$, the language L^* includes all words longer than $3 \cdot 5 - 3 - 5 = 7$.

Example 1.2. For the language $L = \{0, 01, 10, 11, 101\}$ over $\Sigma = \{0, 1\}$, the words in L^* are:

$$0, 00, 01, 10, 11, 000, 001, 010, 011, 100, 101, 110, \ldots.$$

We can see that $111 \notin L^*$ and also every word of the form $111(11)^*$ does not belong to L^*. However, if we add 111 to L, the answer becomes that L^* is cofinite; it contains every word except the word 1.

The problem can be seen as *almost universality* of the language L^*. Kao, Shallit, and Xu [16, 25] showed that, in particular, if L^* is cofinite, then the longest non-expressible words can be exponentially long in $\|L\|_\infty$; their construction is based on the so-called *multi-shift de Bruijn sequences* [17]. This is in contrast with the classical Frobenius problem, where the largest non-expressible integer is bounded quadratically in the largest given integer [7]. A quadratic bound exists also for a similar problem where the iterated shuffle is used instead of the Kleene star operation [20]. Since the shown examples also use exponentially many words in $\|L\|_\infty$, the question about the bound in terms of $\|L\|_1$ or $|L|$ remained open.

In 2009, Shallit and Xu posed the open question about the computational complexity of determining whether L^* is cofinite [25]. They also noted that it is NP-hard and in PSPACE when L is given as a regular expression [26]. This question appears on Shallit's list of open problems [24].

1.2. **Factor universality problem.** A word $u \in \Sigma^*$ is a *factor* (also called *substring*) of a word $w \in \Sigma^*$ if $vuv' = w$ for some words $v, v' \in \Sigma^*$. A language $K \subseteq \Sigma^*$ such that every word over Σ is a factor of some word from K is called *factor universal*.

Problem 1.2 (Factor Universality for a Finite Set of Words)**.** *Given a finite set of words L over a finite non-empty alphabet Σ, is L^* factor universal?*

Finite sets L such that L^* is factor universal are one of the basic concepts in the theory of codes [4, Section 1.5]. They are called *complete sets of words*, and words that are factors of some word in L^* are called *completable*.

Example 1.3. The set $L = \{01, 10, 11, 000\}$ over $\Sigma = \{0, 1\}$ is not complete, since the word 100010001 is not completable. To create a word that contains 1 surrounded by 0s, we have to use either 10 or 01, but then there is no way to build the succeeding 001 or preceding 100, respectively.

Example 1.4. The set $L = \{00, 01, 10, 11\}$ over $\Sigma = \{0, 1\}$ is complete, because every binary sequence of even length is in L^*. We can construct every odd-length binary sequence by removing the first letter of a suitable even-length sequence.

The question about the length of the shortest incompletable words was posed in 1981 by Restivo [23], who conjectured that if a finite set L is not complete, then the shortest incompletable words have length at most $2\|L\|_\infty^2$. The conjecture in this form turned out to be false [11] and $5\|L\|_\infty^2 - \mathcal{O}(\|L\|_\infty)$ was the best lower bound known so far [12], but the relaxed question whether there is a quadratic, or at least polynomial, upper bound remained open and became one of the longstanding unsolved problems in automata theory and the theory of codes. It was generally believed that Restivo's conjecture holds with a larger value of the constant [4]. On the other hand, a sophisticated experimental research dedicated just to this problem [14] suggested that the tight upper bound is unlikely to be quadratic. The best known upper bound was a trivial one, exponential in $\|L\|_1$ thus doubly-exponential in terms of $\|L\|_\infty$ [12].

A polynomial upper bound $\mathcal{O}(\|L\|_1^5)$ was recently derived for the class of sets L called *codes*, which guarantees a unique (unambiguous) factorization of any word to words from L [18]. Since $\|L\|_1$ can be exponentially larger than $\|L\|_\infty$, so the general question about a polynomial upper bound in $\|L\|_\infty$ for this subclass still remains open.

The computational complexity of Problem 1.2 was also an open question. In a more general setting, where instead of checking the factor universality of L^* we check it for an arbitrary regular language specified by an NFA, the problem was shown to be PSPACE-complete [22]. In contrast, it is solvable in linear time when the language is specified by a DFA [22]. Also, some upper bound on the length of the shortest incompletable words was recently derived for the case where the language is specified by an unambiguous NFA [6].

Both the computational complexity question and finding the tight upper bound on the length also appear as one of Berstel, Perrin, and Reutenauer's research problems [4, Resarch problems] and on Shallit's list [24]. The problem itself has been connected with a number of different problems, e.g., testing if all bi-infinite words can be generated by a given list of finite words [22], synchronizing automata and the famous Černý conjecture [8], and the matrix mortality problem [18]. In consequence for the mentioned problems, our solution reveals that the testing problem is PSPACE-complete, that any general weak version of Restivo's conjecture cannot be used to derive good upper bounds for synchronization of automata, and that the matrix mortality problem remains hard when the matrices are restricted to a specific form related to a list of words.

1.3. **Contribution.**

We show that both Problem 1.1 and Problem 1.2 are PSPACE-complete. We also show exponential in $\|L\|_\infty$ and subexponential in $\|L\|_1$ lower bounds for the related length questions. The complexity and the bounds hold even when the alphabet is binary. Since as the input we take a list of words, this also settles the complexity of all problem variants where L is given as a DFA, a regular expression, or an NFA.

To make the reduction feasible, we construct it in several steps. We introduce a rewriting system called *set rewriting* (Section 3), which is a basis for intermediate problems that we reduce from. We translate a set rewriting system first to a DFA, then to a binary DFA, and finally to a binary list of words. The solutions for both problems are based on the same construction of the reduction (Section 4 and 5), with some technical differences. Thus, it seems that the methods may be applicable to some other problems concerning the Kleene star.

The answer for the Frobenius monoid problem can be surprising because the problem is equally hard when L is represented by other common representations that are exponentially more succinct (i.e. DFA, regular expression, or NFA). Kao et al. [16] gave examples of finite languages L such that the longest words not present in the generated cofinite language L^* are of exponential length in $\|L\|_\infty$. However, the number of words in L is also exponential in these examples, thus they do not imply a large lower bound in terms of the size of the input $\|L\|_1$. We strengthen this result by exhibiting examples such that the longest words not present in cofinite L^* are of subexponential length in $\|L\|_1$. The examples are derived from our reduction and its complexity analysis.

The solution for the factor universality problem uses a similar construction. As well, as a corollary, we exhibit a family of sets L of binary words whose shortest incompletable words are of exponential length in $\|L\|_\infty$ and subexponential in $\|L\|_1$. This settles in the negative all weak variations of Restivo's conjecture and essentially closes the longstanding problem.

Finally (Section 7), we note that both problems can be solved in exponential time in $\|L\|_\infty$ while remaining polynomial in $|L|$ thus in $\|L\|_1$. This means that they can be effectively solved when the given set of words is dense, that is, $\|L\|_\infty$ is much smaller (e.g., logarithmic) than $|L|$. We also derive upper bounds on the length of the same order.

We conclude that for a finite list L of words over a fixed alphabet, $2^{\mathcal{O}(\|L\|_\infty)}$ is a tight upper bound on both the length of the longest words that are not in L^* when L^* is cofinite and the length of the shortest incompletable words when L^* is not factor universal. Furthermore, in terms of $\|L\|_1$, the subexponential length $2^{\Theta(\sqrt[5]{\|L\|_1})}$ is attainable.

2. Preliminaries

Let ε denote the empty word.

A *nondeterministic finite automaton* (*NFA*) is a quintuple $\mathcal{A} = (Q_\mathcal{A}, \Sigma, \delta_\mathcal{A}, q_0, F_\mathcal{A})$, where $Q_\mathcal{A}$ is a finite non-empty set of *states*, Σ is a finite non-empty *alphabet*, $\delta_\mathcal{A} \colon Q_\mathcal{A} \times (\Sigma \cup \{\varepsilon\}) \to 2^{Q_\mathcal{A}}$ is the *transition function*, $q_0 \in Q_\mathcal{A}$ is the *initial* state, and $F_\mathcal{A} \subseteq Q_\mathcal{A}$ is the set of *final* states. If for a state $q \in Q_\mathcal{A}$, the set of ε-transitions $\delta_\mathcal{A}(q, \varepsilon)$ is not explicitly defined, we assume $\delta_\mathcal{A}(q, \varepsilon) = \emptyset$ (no such transitions).

We extend $\delta_\mathcal{A}$ to a function $2^{Q_\mathcal{A}} \times \Sigma^* \to 2^{Q_\mathcal{A}}$ as usual. We assume that the extended $\delta_\mathcal{A}$ is complemented by ε-transitions, i.e., for a subset $C \subseteq Q_\mathcal{A}$ and a word $w \in \Sigma^*$, $\delta_\mathcal{A}(C, w)$ is the set of all the states that can be obtained from a state in C by applying sequentially a transition of each of the consecutive letters of w interleaved with any number of ε-transitions, which can be used also at the beginning and the end.

A state $q' \in Q_\mathcal{A}$ is *reachable from a state* $q \in Q_\mathcal{A}$ if there exists a word $w \in \Sigma^*$ such that $q' \in \delta_\mathcal{A}(\{q\}, w)$. Similarly, a subset $S' \subseteq Q_\mathcal{A}$ is *reachable* from $S \subseteq Q_\mathcal{A}$ if there exists a word $w \in \Sigma^*$ such that $\delta_\mathcal{A}(S, w) = S'$. Then we say that q' (resp. S') is *reachable by the word* w *from* q (resp. S).

An automaton *accepts* a word $w \in \Sigma^*$ if $\delta_\mathcal{A}(\{q_0\}, w) \cap F_\mathcal{A} \neq \emptyset$. The set of all accepted words is the *language* of the automaton.

A state $q \in Q_\mathcal{A}$ is called *dead* if no final state is reachable from it, i.e., $\delta_\mathcal{A}(\{q\}, w) \cap F_\mathcal{A} = \emptyset$ for all words $w \in \Sigma^*$. Without affecting the language of an NFA, we can remove all dead states and remove all the transitions to them, i.e., replace $Q_\mathcal{A}$ with $Q_\mathcal{A} \setminus D$ and replace each $\delta_\mathcal{A}(q, a)$ with $\delta_\mathcal{A}(q, a) \setminus D$, where $q \in Q_\mathcal{A}$, $a \in \Sigma \cup \{\varepsilon\}$, and $D \subseteq Q_\mathcal{A}$ is the set of all dead states.

A special case of an NFA is a *deterministic finite automaton* (*DFA*), where for all $q \in Q_{\mathcal{A}}$ and $a \in \Sigma$ we have $|\delta_{\mathcal{A}}(q,a)| = 1$ and there are no ε-transitions (i.e., $\delta_{\mathcal{A}}(q,\varepsilon) = \emptyset$). In this case, we write $\delta_{\mathcal{A}}(q,a) = q'$ instead of $\delta_{\mathcal{A}}(q,a) = \{q'\}$.

Automaton recognizing the Kleene star. We will use the well-known standard construction of an NFA recognizing the Kleene star of the language specified by a DFA (see, e.g., [27]). Let $\mathcal{A} = (Q_{\mathcal{A}}, \Sigma, \delta_{\mathcal{A}}, q_0, F_{\mathcal{A}})$ be a DFA. Then $\mathcal{A}^* = (Q_{\mathcal{A}^*}, \Sigma, \delta_{\mathcal{A}^*}, q_0', \{q_0'\})$ is the NFA obtained from $\mathcal{A}$ as follows. The set of states $Q_{\mathcal{A}^*}$ is $Q_{\mathcal{A}} \cup \{q_0'\}$, where q_0' is a fresh state. The transition function $\delta_{\mathcal{A}^*}$ is defined as $\delta_{\mathcal{A}}(q,a)$ with some additional ε-transitions: we add a ε-transition from q_0' to q_0 and from every final state in $F_{\mathcal{A}}$ to q_0'. The language of the obtained NFA is L^*, where L is the language of $\mathcal{A}$.

We further simplify the construction by removing all the dead states, thus our $\mathcal{A}^*$ do not contain them. Also, if in $\mathcal{A}$ the initial state q_0 is not reachable from itself by any non-empty word (automata with this property are called *non-returning* in the literature), then we can identify q_0' with q_0, which is then the initial state and the unique final state. We will use both simplifications in the constructions in this paper.

3. Set rewriting system

We introduce *set rewriting systems*, which are an auxiliary intermediate formalism for our further reductions.

Definition 3.1. *A* set rewriting system *is a pair* (P,R)*, where* P *is a finite non-empty set of* elements *and* R *is a finite non-empty set of* rules. *A* rule *is a function* $r\colon P \to 2^P \cup \{\bot\}$.

Given a set rewriting system and a subset $S \subseteq P$, a rule r is *legal* if $\bot \notin r(S)$ (i.e., there is no $s \in S$ such that $r(s) = \bot$). The *resulting subset* from applying a legal rule r to S is $S \cdot r = \bigcup_{s \in S} r(s)$. Analogously, we inductively define that a sequence of rules $r_1, \ldots, r_k$ is *legal* if $r_1, \ldots, r_{k-1}$ is legal for S and r_k is legal for $S \cdot r_1 \cdot \ldots \cdot r_{k-1}$. The *resulting subset* from applying a legal sequence of rules is $S \cdot r_1 \cdot \ldots \cdot r_k$.

Note that when a set rewriting system is given as the input for a problem, we can assume polynomial input size in terms of $|P|$ and $|R|$ (e.g., a straightforward encoding requires writing at most $|R| \cdot |P|^2$ elements from $P \cup \{\bot\}$).

3.1. Immortality.

In general, immortality is a classical problem of whether there exists any configuration such that there is an infinite sequence of legally applied transitions to it. This is in contrast to the usual setting, where the initial configuration is given and we ask about reachability. In the case of systems with a bounded configuration space, this is equivalent to the existence of a cycle in the configuration space. For instance, mortality (also called *structural termination* in the literature) problems have been considered for Turing machines [5], where the problem is undecidable, and for linearly bounded Turing machines with a counter [3], where the problem is PSPACE-complete.

Considering our setting, every set rewriting system contains a trivial cycle which is a loop on the empty set. Therefore, in our mortality problem, we have to exclude the empty set as the cycle. A set rewriting system (P,R) is *immortal* if there exists a non-empty subset $S \subseteq P$ and a non-empty sequence of rules $r_1, \ldots, r_k$ that is legal and yields S, i.e., $S \cdot r_1 \cdot \ldots \cdot r_k = S$. It is called *mortal* otherwise.

Furthermore, we add the restriction that the empty set is not reachable from any non-empty subset; this will simplify reasoning in further reductions because otherwise reaching the empty set would need to be treated in a special way, as it does not count as a cycle but does not imply any restriction on any further rule applications. A set rewriting system is *non-emptiable* if for every non-empty subset $S \subseteq P$ and every rule $r \in R$, either $S \cdot r \neq \emptyset$ or r is illegal for S. This condition is equivalent to that for every element $p \in P$ and every rule $r \in R$, we have $r(p) \neq \emptyset$. Hence, it is easy to check if a set rewriting system is non-emptiable.

Problem 3.2 (Immortality of Set Rewriting)**.** *Given a non-emptiable set rewriting system, is it immortal?*

First, we show that a mortal set rewriting system can admit exponentially long sequences of legal rules.

Theorem 3.1. *For a mortal non-emptiable set rewriting system (P, R), for every non-empty subset of P, the length of any legal sequence of rules is at most $2^{|P|} - 2$. For every $n \geq 1$, there exist a set rewriting system (P, R) with $|P| = |R| = n$ and some subset of P that meet the upper bound.*

Proof. The upper bound is clear since there are $2^{|P|} - 1$ distinct non-empty subsets and a legal sequence of length $2^{|P|} - 2$ involves all of them.

To show tightness, we construct a set rewriting system (P, R) with $n = |P|$ rules. The elements will encode a specific binary counter. Let $P = \{b_0, \ldots, b_{n-1}\}$. For a subset $S \subseteq P$, we define $val(S, i) = 2^i$ if $b_i \in S$ and $val(S, i) = 0$ otherwise, and we set the *counter value* $val(S) = \sum_{0 \leq i \leq n-1} val(S, i)$. For every $j \in \{0, \ldots, n-1\}$, we introduce a rule r_j that, if it is legal, will increase the value of the counter by at least 1. The rules r_j are defined as follows:

- $r_j(b_j) = \bot$;
- $r_j(b_i) = \{b_j\}$ for $i \in \{0, 1, \ldots, j-1\}$;
- $r_j(b_i) = \{b_j, b_i\}$ for $i \in \{j+1, j+2, \ldots, n-1\}$.

First, we observe that each legal rule r_j applied to a non-empty set $S \subseteq P$ increases the counter value by at least 1, i.e., $val(S) < val(S \cdot r_j)$. It is because we know that $val(S, j) = 0$, as otherwise the rule would be illegal, and

$$\begin{aligned} val(S \cdot r_j) = \sum_{j<i<n} val(S \cdot r_j, i) + 2^j \quad &= \sum_{j<i<n} val(S, i) + 2^j \ > \\ > \sum_{j<i<n} val(S, i) + \sum_{0 \leq i<j} 2^i \ &\geq \sum_{0 \leq i<n} val(S, i) \qquad = \ val(S). \end{aligned}$$

Second, we observe that for every non-empty $S \subsetneq P$, there exists a rule r_j that increases the counter value exactly by 1. We choose the rule r_j for j being the smallest index such that $b_j \notin S$, and we have $val(S \cdot r_j) = val(S) + 1$. Furthermore, for $S = P$ there is no legal rule.

It follows that the set rewriting system is mortal, and for $S = \{b_0\}$, the longest possible legal sequence of rules has length $2^n - 2$. □

Now, we show the PSPACE-completeness of the immortality problem. The idea is a reduction from the non-universality of an NFA. The NFA is combined with the counter developed for the proof of Theorem 3.1. The NFA is encoded within the set rewriting system together with a counter incrementing its value with each transition. The counter can be reset only if there exists a non-accepted word by the NFA, in which case it allows repeating a subset in the set rewriting system.

Theorem 3.2. *Problem 3.2 (Immortality of Set Rewriting) is PSPACE-complete.*

Proof. To solve the problem in NPSPACE thus in PSPACE, given a set rewriting system (P, R), it is enough to guess a subset $S \subseteq P$ and a length $k \leq 2^{|P|}$, and then to guess at most k rules (storing only the current one), verifying whether the resulted subset is the same as S.

For PSPACE-hardness, we reduce from the non-universality problem for an NFA. Given an NFA $\mathcal{A} = (Q_{\mathcal{A}}, \Sigma, \delta_{\mathcal{A}}, q_0, F_{\mathcal{A}})$, the question whether there exists any not accepted word over Σ is PSPACE-complete (e.g., [1, Section 10.6]). We can assume that $\mathcal{A}$ does not have ε-transitions.

Let $n = |Q_{\mathcal{A}}|$. We construct a set rewriting system (P, R) of size polynomial in n. As an ingredient, we use the counter from the proof of Theorem 3.1. Let P be the disjoint union of $Q_{\mathcal{A}}$ and $C = \{b_i \mid i \in \{0, 1, \ldots, n-1\}\}$. The elements of C will encode the binary counter and for a subset $S \subseteq P$, we define $\operatorname{val}(S, i) = 2^i$ if $b_i \in S$ and $\operatorname{val}(S, i) = 0$ otherwise, and we set $\operatorname{val}(S) = \sum_{0 \le i \le n-1} \operatorname{val}(S, i)$.

For every letter $a \in \Sigma$ and every $j \in \{0, 1, \ldots, n-1\}$, we introduce a rule $r_{a,j}$ that acts as a in the NFA on $Q_{\mathcal{A}}$ and, on the counter part, sets the j-th position of the counter. The rules $r_{a,j}$ are defined as follows:

- $r_{a,j}(b_j) = \bot$;
- $r_{a,j}(b_i) = \{b_j\}$ for $i \in \{0, 1, 2, \ldots, j-1\}$;
- $r_{a,j}(b_i) = \{b_j, b_i\}$ for $i \in \{j+1, j+2, \ldots, n-1\}$;
- $r_{a,j}(q) = \delta_{\mathcal{A}}(q, a) \cup \{b_j\}$ for $q \in Q_{\mathcal{A}}$.

We also introduce the *reset rule* that is defined as:

- $r_{\text{reset}}(q) = \begin{cases} \bot, & \text{if } q \in F_{\mathcal{A}}; \\ \{q_0, b_0\} & \text{otherwise.} \end{cases}$

Now we observe the correctness. Assume that there is a word not accepted by $\mathcal{A}$. Note that if w is a shortest non-accepted word, then $q_0 \notin \delta_{\mathcal{A}}(q_0, u)$ for all non-empty prefixes u of w. Hence, there exists a non-accepted word $w = a_1 a_2 \ldots a_m$ of length at most 2^{n-1}. As observed in the proof of Theorem 3.1, we know that for each value x of the counter, there exists a rule that increments the counter value exactly by 1. Let $f(x)$ be the smallest index of a zero in the binary representation of x, where the least significant position of the binary representation is indexed by zero; hence a rule $r_{a_i, f(x)}$, if it is legal for S, increments the counter value of S by 1. Then the set $S = \{b_0, q_0\} \cdot r_{a_1, f(1)} \cdot r_{a_2, f(2)} \cdot \ldots \cdot r_{a_m, f(m)}$ has the property that $\operatorname{val}(S) = m < 2^n$ and $S \cap F = \emptyset$, because w is not accepted by $\mathcal{A}$. Thus, rule r_{reset} is legal, so $\{b_0, q_0\} \cdot r_{a_1, f(1)} \cdot r_{a_2, f(2)} \cdot \ldots \cdot r_{a_m, f(m)} \cdot r_{\text{reset}} = \{b_0, q_0\}$. Hence the set rewriting system is immortal.

For the converse, assume that there exists a subset $S \subseteq P$ and a non-empty sequence of rules $r_{j_1}, r_{j_2}, \ldots, r_{j_m}$ such that $S \cdot r_{j_1} \cdot r_{j_2} \cdot \ldots \cdot r_{j_m} = S$. As observed in the proof of Theorem 3.1, we know that every rule different from r_{reset} increments the counter value by at least 1. Hence, there must be some index $1 \le k \le m$ such that $r_{j_k} = r_{\text{reset}}$. Consider the sequence of rules $r_{j_1}, r_{j_2}, \ldots, r_{j_m}, r_{j_1}, r_{j_2} \ldots, r_{j_m}$. In this sequence, r_{reset} appears at least twice. Taking a shortest sequence of rules between any two r_{reset} rules (not including the reset rules), we get a sequence $r_{a_1, i_1}, r_{a_2, i_2}, \ldots, r_{a_d, i_d}$ such that $\{q_0, b_0\} \cdot r_{a_1, i_1} \cdot r_{a_2, i_2} \cdot \ldots \cdot r_{a_d, i_d} r_{\text{reset}} = \{q_0, b_0\}$. Since r_{reset} is legal when applied, the word $a_1 a_2 \ldots a_d$ is such that $\delta_{\mathcal{A}}(q_0, a_1 a_2 \ldots a_d) \cap F = \emptyset$ thus is not accepted by $\mathcal{A}$. □

Lemma 3.3. *If a rule r is legal for a subset $S \subseteq P$, then it is also legal for every subset $S' \subseteq S$ and $S' \cdot r \subseteq S \cdot r$.*

By this observation, when showing if the system is immortal, it is enough to consider only singleton subsets S from which we start applying rules to find a cycle. Although a singleton does not necessarily occur in a cycle, a non-emptiable set rewriting system is immortal if and only if, for some singleton, there exist arbitrary long legal sequences of rules.

3.2. **Emptying.** The second problem is the reachability of the empty set. This is related to factor universality and is necessary for our further reduction.

For a subset $S \subseteq P$, a sequence of rules $r_1, \ldots, r_k$ such that $S \cdot r_1 \cdot \ldots \cdot r_k = \emptyset$ is called *S-emptying*.

We call a set rewriting system *permissive* if all rules are always legal. In other words, all rules are legal for P. A permissive set rewriting system (P, R) is equivalent to the *semi-NFA* whose set of states is P and the alphabet is R; the NFA initial and final states are irrelevant.

Problem 3.3 (Emptying Set Rewriting)**.** *For a given permissive set rewriting system (P, R), does there exist a P-emptying sequence of rules?*

Let $\mathcal{A} = (Q_\mathcal{A}, \Sigma, \delta_\mathcal{A}, q_0, F_\mathcal{A})$ be an NFA. Analogously to set rewriting, for a subset $S \subseteq Q_\mathcal{A}$, a word $w \in \Sigma^*$ is called *S-emptying* if $\delta_\mathcal{A}(S, w) = \emptyset$.

The following criterion for the factor universality of a language represented by an NFA is known.

Proposition 3.4 ([22])**.** *Let $\mathcal{A} = (Q_\mathcal{A}, \Sigma, \delta_\mathcal{A}, q_0, F_\mathcal{A})$ be an NFA such that every state is reachable from the initial state q_0 and there are no dead states. Then a word w is not a factor of a word accepted by $\mathcal{A}$ if and only if w is $Q_\mathcal{A}$-emptying. The language of $\mathcal{A}$ is factor universal if and only if there does not exist a $Q_\mathcal{A}$-emptying word.*

It is known that the problem of whether a given language specified by an NFA is factor universal is PSPACE-complete [22]. Since it is also easy to solve Problem 3.3 in PSPACE, we have:

Proposition 3.5. *Problem 3.3 (Emptying Set Rewriting) is PSPACE-complete.*

Additionally, we will need an exponential lower bound on the length of the shortest P-emptying sequences of rules. For this, we also develop a specific counter, but now counting downwards and allowing to decrease the value by at most 1. Instead of rules being illegal, undesired rule applications reset the counter to the maximal value.

Theorem 3.6. *For a permissive set rewriting system (P, R), if there exists a P-emptying sequence of rules, then the shortest such sequences have length at most $2^{|P|} - 1$. For every $n \geq 1$, there exists a set rewriting system (P, R) with $|P| = |R| = n$ that meets the bound.*

Proof. The upper bound $2^{|P|} - 1$ is trivial.

For every $n \geq 1$, we construct a permissive set rewriting system (P, R), which represents a binary counter of length n. Let $P = \{b_i \mid i \in \{0, 1, \ldots, n-1\}\}$. For a subset $S \subseteq P$, we define $\mathrm{val}(S, i) = 2^i$ if $b_i \in S$ and $\mathrm{val}(S, i) = 0$ otherwise, and $\mathrm{val}(S) = \sum_{0 \leq i \leq n-1} \mathrm{val}(S, i)$.

We define the rules that allow decreasing the value of the counter by 1. If a wrong rule is used, the counter is reset to its maximal value. The set of rules R consists of rules r_j for $j \in \{0, 1, \ldots, n-1\}$, where each r_j is defined as follows:

(1) $r_j(b_i) = P$ for $i \in \{0, 1, \ldots, j-1\}$;
(2) $r_j(b_j) = \{b_i \mid i \in \{0, 1, \ldots, j-1\}\}$;
(3) $r_j(b_i) = \{b_i\}$ for $i \in \{j+1, j+2, \ldots, n-1\}$.

We observe that emptying this set rewriting system corresponds to setting the counter value to 0. For a subset S, let i be the smallest index such that $b_i \in S$. Then for all the smaller positions $j < i$, $b_j \notin S$. Notice that for all rules r_k for $k \in \{1, 2, \ldots, n-1\} \setminus \{i\}$, we have $\mathrm{val}(S \cdot r_k) \geq \mathrm{val}(S)$. This is because if $k < i$, then $S \cdot r_k = S$, and if $k > i$, then $S \cdot r_k = P$. Hence, the only rule that decreases the counter is r_i, and then $\mathrm{val}(S \cdot r_i) = \mathrm{val}(S) - 1$. Thus, the shortest sequence of rules that is P-emptying has length $2^n - 1$. □

4. The Frobenius monoid problem

We note the known result about the PSPACE-membership.

Proposition 4.1 ([25, Corollary 5.5.8]). *Problem 1.1 is in PSPACE.*

For PSPACE-hardness, we reduce from Problem 3.2 (Immortality of Set Rewriting) to Problem 1.1 (Frobenius Monoid Problem for a Finite Set of Words). In the first step, we reduce to the case where L is specified as a DFA instead of a list of words. Then we binarize the DFA, and finally, we count the number of words in the language to bound the size of the list of words.

4.1. The DFA construction.

As the input for the reduction, we take a non-emptiable set rewriting system (P, R). Without loss of generality, we assume the set of elements $P = \{p_1, p_2, \dots, p_\ell\}$ and the rules $R = \{r_1, r_2, \dots, r_m\}$.

We construct a DFA $\mathcal{A} = (Q_\mathcal{A}, \Sigma, \delta_\mathcal{A}, q_0, F)$ recognizing a finite language L such that L^* is not cofinite if and only if (P, R) is immortal. The number and the lengths of words in L will be polynomial, which will allow further polynomial reduction to the case of a list of words. First, we define the DFA; then we describe its mechanism, and finally, we prove the correctness formally.

The DFA is presented in Fig. 1. Since this is a DFA construction, for every letter and every state there should be a transition; for a clearer picture, we have omitted drawing the transitions from the setting states to f_0 on R.

The alphabet of $\mathcal{A}$ is $\Sigma = R \cup \{\alpha\}$, where the letters from R are *rule letters*, and α is a fresh special letter that will be used to shift the states and separate applications of rule letters. The set of states $Q_\mathcal{A}$ is the disjoint sum of the following sets:

- $\{q_0\}$; the initial state.
- $Q_\mathrm{P} = P$; the elements of the set rewriting system.
- $Q_\mathrm{F} = \{f_i \mid i \in \{0, 1, \dots, \ell\}\}$; the *forcing states.*
- $\{s_h^{i,j} \mid i, h \in \{1, 2, \dots, \ell\} \wedge j \in \{1, 2, \dots, m\} \wedge r_j(p_i) \neq \perp\}$; the *setting states*; a setting state $s_h^{i,j}$ is dedicated to the element p_i and the rule r_j.
- $\{q_\mathrm{g}\}$; the *guard state.*
- $\{q_\mathrm{s}\}$; the *sink state*, which is the unique dead state.

The transition function $\delta_\mathcal{A}$ is defined as follows (see also Fig. 1):

- $\delta_\mathcal{A}(q_0, \alpha) = p_1$.
- $\delta_\mathcal{A}(p_i, \alpha) = p_{i+1}$ for all $i \in \{0, 1, \dots, \ell - 1\}$.
- $\delta_\mathcal{A}(p_\ell, \alpha) = f_0$.
- $\delta_\mathcal{A}(p_i, r_j) = \begin{cases} s_\ell^{i,j}, & \text{if } r_j(p_i) \neq \perp \\ f_0, & \text{otherwise} \end{cases}$
  for all $i \in \{1, 2, \dots, \ell\}$ and $j \in \{1, 2, \dots, m\}$; the transition of a rule letter maps the elements from Q_P either to the first setting state in the dedicated chain, when the rule is legal, or directly to f_0, otherwise.
- $\delta_\mathcal{A}(q_0, r_j) = f_0$ for all $j \in \{1, 2, \dots, m\}$.
- $\delta_\mathcal{A}(s_h^{i,j}, \alpha) = s_{h-1}^{i,j}$ for all $i \in \{1, 2, \dots, \ell\}$, $j \in \{1, 2, \dots, m\}$, and $h \in \{\ell, \ell - 1, \dots, 2\}$.
- $\delta_\mathcal{A}(s_1^{i,j}, \alpha) = q_\mathrm{g}$ for all $i \in \{1, 2, \dots, \ell\}$ and $j \in \{1, 2, \dots, m\}$.
- $\delta_\mathcal{A}(s_h^{i,j}, r_k) = f_0$ for all $i, h \in \{1, 2, \dots, \ell\}$ and $j, k \in \{1, 2, \dots, m\}$.
- $\delta_\mathcal{A}(q_\mathrm{g}, \alpha) = f_0$.
- $\delta_\mathcal{A}(q_\mathrm{g}, r_j) = q_\mathrm{s}$ for all $j \in \{1, 2, \dots, m\}$.
- $\delta_\mathcal{A}(f_i, \alpha) = f_{i+1}$ for all $i \in \{0, 1, \dots, \ell - 1\}$.

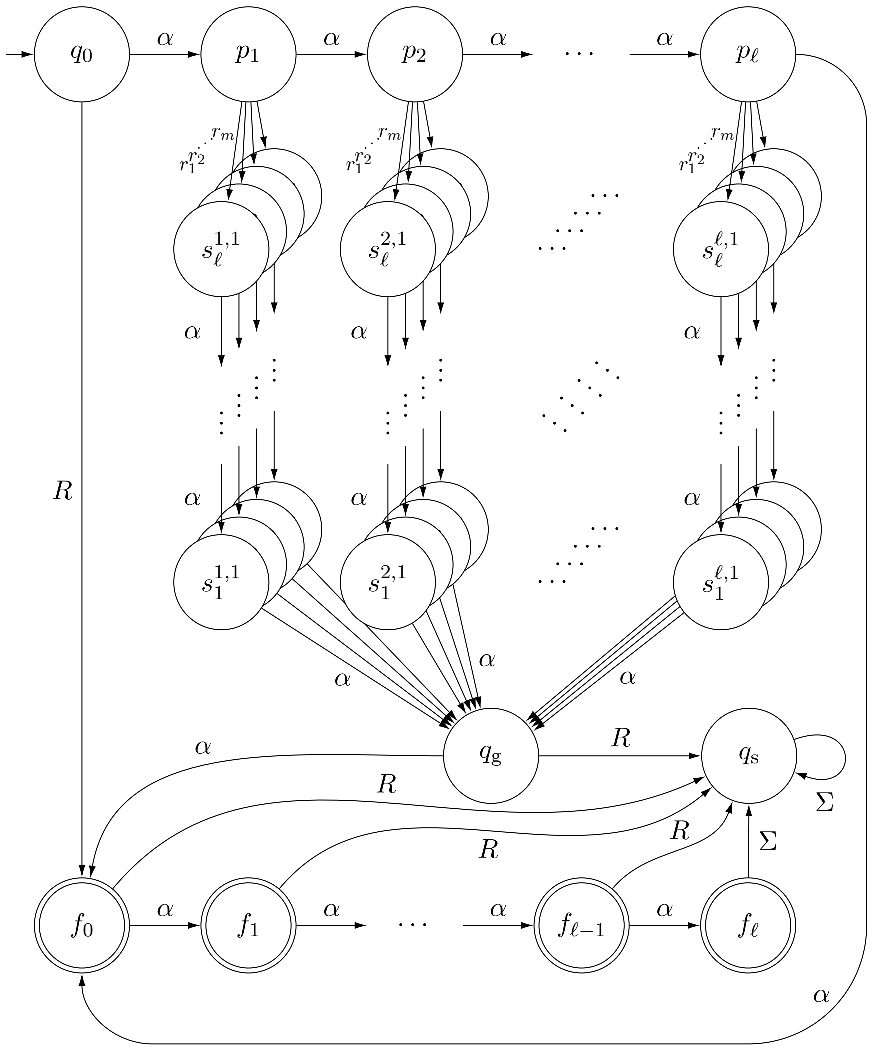

FIGURE 1. The scheme of the DFA $\mathcal{A}$ for a set rewriting system. The transitions from all $s_h^{i,j}$ to f_0 on R are not drawn.

- $\delta_\mathcal{A}(f_i, r_j) = q_\mathrm{s}$ for all $i \in \{0, 1, \ldots, \ell\}$ and $j \in \{1, 2, \ldots, \ell\}$.
- $\delta_\mathcal{A}(f_\ell, \alpha) = q_\mathrm{s}$.

The set of final states $F_\mathcal{A}$ is the disjoint union of the following sets:

- Q_F; all forcing states are final.
- $\{s_h^{i,j} \mid i, h \in \{1, 2, \ldots, \ell\} \wedge j \in \{1, 2, \ldots, m\} \wedge r_j(p_i) \neq \bot \wedge p_h \in r_j(p_i)\}$; states in a setting chain are final according to the rule r_j applied to the element p_i.

4.1.1. *The mechanism.* We describe the idea of the construction and introduce a few necessary notions for the analysis.

To reason about L^*, we construct the NFA $\mathcal{A}^* = (Q_{\mathcal{A}^*}, \Sigma, \delta_{\mathcal{A}^*}, \{q_0\}, \{q_0\})$ recognizing the language L^* as described in Section 2. This NFA is $\mathcal{A}$ with added ε-transitions from every final state to q_0 and with the dead state q_{s} removed.

Given a word w and a subset C by the context, the states that are obtained by applying the action of w to C, i.e., $\delta_{\mathcal{A}^*}(C, w)$, are called *active*.

A word $w \in \Sigma^*$ is *irrevocably accepted* if for every $u \in \Sigma^*$, the word wu belongs to L^*. One of the key properties of our construction is that all words such that the state f_0 becomes active when starting from $\{q_0\}$ are irrevocably accepted. This means that for a non-accepted word w, state f_0 (as every other forcing state) cannot be activated by the application of any prefix of w to the initial subset $\{q_0\}$.

In general, $\mathcal{A}^*$ simulates the set rewriting system. A subset of Q_{P} corresponds to the same subset of elements in the set rewriting system. Sequences of rules translate to words of a specific form. When a sequence of rules is not legal for a subset, the corresponding word activates f_0 at some point. The same holds for a word that violates the specific form. This provides a correspondence between sequences of legal rules and possibly non-accepted words.

For a subset $S \subseteq Q_{\mathrm{P}}$, applying the word $r_j\alpha^\ell$, for some $r_j \in R$, corresponds to applying the rule r_j in the set rewriting system for S, i.e., $S' = \delta_{\mathcal{A}^*}(S, r_j\alpha^\ell) = (S \cdot r_j) \cup \{q_{\mathrm{g}}\}$ when r_j is legal for S. The chain of the setting states for a state $p_i \in Q_{\mathrm{P}}$ has its final states in the DFA defined accordingly to the action of the rule r_j for the element p_i. The indices keep the correspondence that if a state $s_h^{i,j}$ is final in the DFA, then after applying $r_j\alpha^\ell$, the state p_h becomes active if and only if $p_i \in S$ (assuming that the rule r_j is legal for S). In the end, the guard state q_{g} is additionally activated, which ensures that we must use a rule letter as the next one since the transition of α maps the guard state to f_0. Without the guard state, sometimes one could use more α letters to shift the states within Q_{P} and in this way cheat by obtaining a different subset of Q_{P}. If the rule r_j is not legal for S, then the transitions of r_j directly activate f_0.

A word u is *simulating for a subset* $S \subseteq Q_{\mathrm{P}}$ if it is in the form of $r_{i_1}\alpha^\ell r_{i_2}\alpha^\ell \cdot \ldots \cdot r_{i_k}\alpha^\ell$ (for $k \geq 0$) and the sequence of the rules $r_{i_1}, r_{i_2}, \ldots, r_{i_k}$ is legal for S in the set rewriting system. The actions of these words correspond to applying the contained sequence of rules. The construction ensures that, for a subset containing a non-empty $S \subseteq Q_{\mathrm{P}}$ together with the guard state q_{g}, using a simulating word is the only possibility to avoid f_0.

A special case occurs at the beginning, i.e., for the initial subset $\{q_0\}$. To avoid activation of f_0, we must start with α^i for any $1 \leq i \leq \ell$. Then we obtain the subset $\delta_{\mathcal{A}^*}(\{q_0\}, \alpha^i) = \{p_i\}$; this corresponds to the selection of the initial singleton in the set rewriting system (cf. Lemma 3.3). After that, a simulating word must be used, unless f_0 is activated. Note that $Q_{\mathrm{P}} \cup \{q_{\mathrm{g}}\}$ does not contain final states, thus in this way we get non-accepted words. It follows that we can find arbitrarily long such words if and only if the set rewriting system is immortal.

4.1.2. *Correctness.* We prove the correctness formally through the following lemmas.

The first lemma states that whenever f_0 becomes active, all subsequent words will be accepted, thus f_0 must be avoided when constructing a non-accepted word.

A word $w \in \Sigma^*$ is *f_0-omitting for a subset $C \subseteq Q_{\mathcal{A}^*}$* if there is no prefix u of w such that $f_0 \in \delta_{\mathcal{A}^*}(C, u)$. It is simply *$f_0$-omitting* if it is f_0-omitting for $\{q_0\}$.

Lemma 4.2. *If a word $w \in \Sigma^*$ is not f_0-omitting, then it is irrevocably accepted.*

Proof. If $w \in \Sigma^*$ is not f_0-omitting, then there is a prefix u of w such that $f_0 \in \delta_{\mathcal{A}^*}(\{q_0\}, u)$. It is enough to observe that for every word $v \in \Sigma^*$, the set $\delta_{\mathcal{A}^*}(\{f_0\}, v)$ contains a forcing state. All forcing states are final, thus uv and, in particular, all words containing w as a prefix will be accepted. Suppose this is not the case, and let v be a shortest word such that $\delta_{\mathcal{A}^*}(\{f_0\}, v)$ does not contain any forcing states. Then for every non-empty proper prefix v' of v, $\delta_{\mathcal{A}^*}(\{f_0\}, v')$ does not contain f_0, as otherwise the suffix v'', where $v'v'' = v$, would be a shorter word than v with the same property. Note that $\delta_{\mathcal{A}^*}(\{f_0\}, \varepsilon) = \{f_0, q_0\}$, and in general, whenever a forcing state is active, q_0 is also active. Thus the only possibility for v is to start with $\alpha^{\ell+1}$; otherwise, active state q_0 would be mapped to f_0 by the transition of a rule letter after α^i for an $i \leq \ell$. However, the action of $\alpha^{\ell+1}$ through the chain on Q_P also maps q_0 to f_0, which yields a contradiction. □

Applying a simulating word corresponds to applying the sequence of rules that is contained in it. The following lemma formalizes this claim.

Lemma 4.3. *Let $C \subseteq Q_\mathrm{P} \cup \{q_\mathrm{g}\}$ be such that $S = C \cap Q_\mathrm{P}$ is non-empty, and let $w = r_{i_1}\alpha^\ell \dots r_{i_k}\alpha^\ell$ be a simulating word for S. Then $\delta_{\mathcal{A}^*}(C, w) = (S \cdot r_{i_1} \cdot \dots \cdot r_{i_k}) \cup \{q_\mathrm{g}\}$.*

Proof. Let C and S be as in the lemma, and let r_j be a rule that is legal for S. The transitions of the letter r_j map each state $p_i \in S$ to $s^{i,j}_\ell$. Then the action of α^ℓ maps these active states along the setting chains, activating state q_0 whenever an active setting state is final. Eventually, they are mapped to q_g. A state $s^{i,j}_h$ is final if and only if $p_h \in r_j(p_i)$. From the construction, if $s^{i,j}_h$ is final, then q_0 becomes active after applying $\alpha^{\ell-h}$. Then q_0 is mapped to p_h by the action of the remaining α^h. After the last occurrence of α, the last active setting states are mapped to the guard state q_g, and there is at least one such active state since S is non-empty. Finally, if besides S, C contains the guard state, the action of r_j deactivates it (maps to the empty set). Hence, we have $\delta_{\mathcal{A}^*}(C, r_j\alpha^\ell) = (S \cdot r_j) \cup \{q_\mathrm{g}\}$.

Since the set rewriting system is non-emptiable, the set $S \cdot r_j$ is non-empty, thus we can apply the argument iteratively. Hence, the lemma follows by induction on k. □

We show that, unless f_0 is activated, a word applied to a subset $C \subseteq Q_\mathrm{P} \cup \{q_\mathrm{g}\}$ must be a prefix of a simulating word for $S = C \cap Q_\mathrm{P}$. The required condition is that the guard state is also present in C, so one cannot shift the states on Q_P by using α.

Lemma 4.4. *Let $C = S \cup \{q_\mathrm{g}\}$, where $S \subseteq Q_\mathrm{P}$ is non-empty. If w is f_0-omitting for C, then w is a prefix of a simulating word for S.*

Proof. First, we observe that every word w which does not activate f_0 starting from C, unless it is the empty word, must start with a rule letter r_j, since using α maps q_g to f_0 and we have assumed $q_\mathrm{g} \in C$. Additionally, r_j must be legal for S, as otherwise f_0 would be activated. Afterwards, at least one of the first setting states must be active, because $S \neq \emptyset$. Hence α^ℓ must be used, unless w ends before that and thus is a prefix of this pattern. By Lemma 4.3 for C and $w = r_j\alpha^\ell$, we know that the set of active states is now $(S \cdot r_j) \cup \{q_\mathrm{g}\}$. By iterating this argument, we conclude that between each rule letter there must be exactly ℓ letters α, it must start with a rule letter, and at the end, there are at most ℓ letters α. Furthermore, the rule letters must form a legal sequence of rules for S. Therefore, we know that word w has to be a prefix of some simulating word for S. □

In the beginning, before we may apply a simulating word, we can choose an arbitrary singleton $\{p_i\}$ as the initial subset. Then a simulating word must be applied, as otherwise f_0 is activated.

Lemma 4.5. *If a word w is f_0-omitting, then w is a prefix of $\alpha^i w'$, where $1 \leq i \leq \ell$ and w' is a simulating word for $\{p_i\}$.*

Proof. Let w be a f_0-omitting word, and write $w = \alpha^i w'$ for $i \geq 0$ and $w' \in \Sigma^*$ that does not start with α. Since we start from $\{q_0\}$, we know that $1 \leq i \leq \ell$ unless w is empty. We have

$\delta_{\mathcal{A}^*}(\{q_0\}, \alpha^i) = \{p_i\}$. Then w' begins with some rule letter r_j, which must be a legal rule for $\{p_i\}$ in the set rewriting system, followed by α^ℓ, unless w' is shorter and thus is a prefix of this pattern.

Hence $w = \alpha^i r_j \alpha^\ell u$. By Lemma 4.3, we have $C = \delta_{\mathcal{A}^*}(\{q_0\}, \alpha^i r_j \alpha^\ell) = S \cup \{q_\mathrm{g}\}$ for $S = \{p_i\} \cdot r_j$. Since the set rewriting is non-emptiable, $S \neq \emptyset$. By Lemma 4.4 applied to C, we know that u must be a prefix of a simulating word for S. It follows that $r_j \alpha^\ell u$ is a prefix of a simulating word for $\{p_i\}$. ☐

Finally, we show the equivalence between the immortality of the set rewriting system and the non-cofiniteness of the language of $\mathcal{A}^*$.

Lemma 4.6. *The set rewriting system (P, R) is immortal if and only if there are infinitely many words not accepted by $\mathcal{A}^*$.*

Proof. Suppose that the set rewriting system is immortal. For every $k > 0$, we will construct a non-accepted word w of length at least $k \cdot (\ell + 1)$. Since the system is immortal and by Lemma 3.3, there exists a singleton $\{p_i\}$ and a legal sequence $r_{i_1}, \ldots, r_{i_k}$ of k rules for $\{p_i\}$. Hence, $w = r_{i_1}\alpha^\ell \ldots r_{i_k}\alpha^\ell$ is a simulating word for $S = \{p_i\}$. By Lemma 4.3, we know that $\delta_{\mathcal{A}^*}(\{q_0\}, \alpha^i w) \subseteq Q_\mathrm{P} \cup \{q_\mathrm{g}\}$, which does not contain any final states, thus $\alpha^i w$ is not accepted.

Conversely, suppose that L^* is not cofinite. Then there are infinitely many words that are not accepted, which, in particular, by Lemma 4.2, must be f_0-omitting. Let w be a f_0-omitting word of length at least $\ell + (\ell + 1)2^{|P|}$. By Lemma 4.5, we know that w has the form of $\alpha^i w'$, where $1 \leq i \leq \ell$ and w' is a prefix of a simulating word for $\{p_i\}$. This simulating word must have length at least $(\ell + 1)2^{|P|}$, hence it contains a sequence of $k \geq 2^{|P|}$ rule letters. We conclude that this sequence $r_{i_1}, r_{i_2}, \ldots, r_{i_k}$ is legal for $\{p_i\}$, and it does not lead to the empty set as the set rewriting system is non-emptying. If we look at the sequence of the sets of elements $S_j = \{p_i\} \cdot r_{i_1} \cdot \ldots \cdot r_{i_j}$, for $j \in \{0, \ldots, 2^{|P|}\}$, then we can find two distinct indices x and y such that $x < y$ and $S_x = S_y$. Hence, the rewriting system is immortal due to S_x and the sequence $r_{i_{x+1}}, r_{i_{x+2}}, \ldots, r_{i_y}$. ☐

We conclude this part with

Theorem 4.7. *Problem 1.1 is PSPACE-hard if L is specified by a DFA over a given alphabet.*

4.2. Binarization.

We show that the PSPACE-hardness still holds when the alphabet is restricted to two letters. Note that a standard binarization of an arbitrary language, where we uniformly replace each letter with an equal-length binary encoding, does not work here. Except for some trivial cases, if incomplete encodings are accepted, then the Kleene star will be always cofinite, and if they are not accepted, then it will never be cofinite. Therefore, we have to use a different way and utilize specific properties of the construction.

First, we define the following binary encoding $bin\colon \Sigma \to \{0,1\}^*$: let $bin(\alpha) = 0$, $bin(r_i) = 1^i 0$ for all $0 \leq i \leq m-1$, and $bin(r_m) = 1^m$. We extend the function bin to a function $bin\colon \Sigma^* \to \{0,1\}^*$ in a natural way. Note that our encoding is a maximal prefix code, which means that $bin(u) \neq bin(v)$ for $u \neq v$, and also, every binary word $w' \in \{0,1\}^*$ contains a unique maximal prefix that is the encoding of some word over Σ; this prefix has length at least $|w'| - (m-1)$.

We modify the construction of $\mathcal{A}$ from Subsection 4.1 using the same notation. We construct a binary DFA $\mathcal{B} = (Q_\mathcal{B}, \{0,1\}, \delta_\mathcal{B}, q_0, F_\mathcal{B})$, where $Q_\mathcal{B}$ is $Q_\mathcal{A}$ with some states added, and q_0 and the set of final states $F_\mathcal{B} = F_\mathcal{A}$ are the same as in the original $\mathcal{A}$. All the transitions labeled by α are now labeled by 0. For each state $p_i \in Q_\mathrm{P}$, we introduce $m-1$ new intermediate *choice states* in the way that the binary word encoding $bin(r_j)$ of a rule letter r_j acts as r_j on p_i in $\mathcal{A}$. The construction of these states is shown in Fig. 2. Formally, we add states $c^{i,j}$ for all $i \in \{1, \ldots, \ell\}$ and $j \in \{1, \ldots, m-1\}$, and the related transitions for all i are defined by:

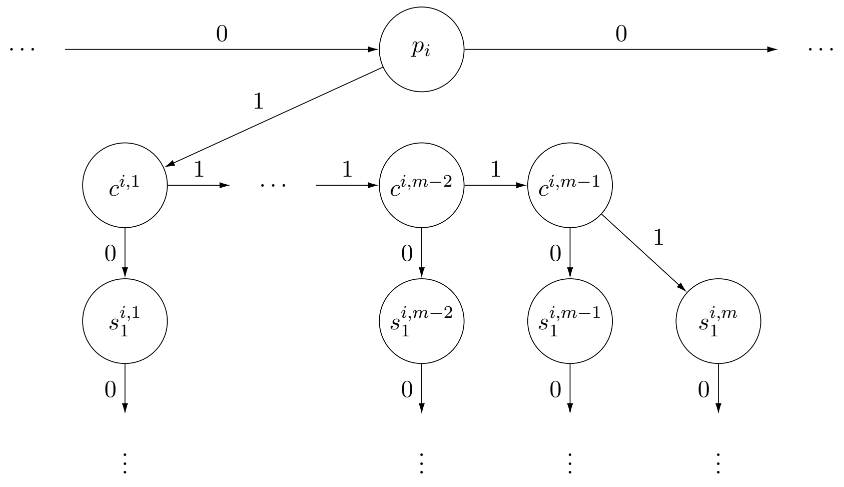

FIGURE 2. The fragment of the binary DFA $\mathcal{B}$ with the choice states of a state p_i. The transitions on 1 from the setting states are not drawn and go to f_0.

- $\delta_{\mathcal{B}}(p_i, 1) = c^{i,1}$.
- $\delta_{\mathcal{B}}(c^{i,j}, 1) = c^{i,j+1}$ for all $j \in \{1, \dots, m-2\}$.
- $\delta_{\mathcal{B}}(c^{i,j}, 0) = \begin{cases} s_1^{i,j}, & \text{if } r_j(p_i) \neq \bot \\ f_0, & \text{otherwise} \end{cases}$
for all $j \in \{1, \dots, m-1\}$.
- $\delta_{\mathcal{B}}(c^{i,m-1}, 1) = \begin{cases} s_1^{i,m}, & \text{if } r_m(p_i) \neq \bot \\ f_0, & \text{otherwise.} \end{cases}$

The transitions of the rule letters on $Q_{\mathcal{B}} \setminus Q_{\mathrm{P}}$ are simply replaced with one transition labeled by 1; all of those transitions in $\mathcal{A}$ are the same for every $r_j \in R$ and lead to either f_0 or q_{s}.

The correctness of the binarization is observed through the following lemmas. First, we state that for a word that is f_0-omitting in the original automaton, the corresponding binary encoding works the same in the binarized one. Then we have a similar statement for not f_0-omitting words but restricted to keeping the property of being f_0-omitting.

Lemma 4.8. *If a word $w \in \Sigma^*$ is f_0-omitting for a subset $C \subseteq Q_{\mathcal{A}^*} \setminus Q_{\mathrm{F}}$ in $\mathcal{A}^*$, then $\delta_{\mathcal{B}^*}(C, bin(w)) = \delta_{\mathcal{A}^*}(C, w)$.*

Proof. This can be observed by analyzing the transitions of each letter in Σ from each state in $Q_{\mathcal{A}^*} \setminus Q_{\mathrm{F}}$ in $\mathcal{A}^*$ together with the actions of the corresponding binary encodings in $\mathcal{B}^*$. □

From Lemma 4.8, in particular, if a word $w \in \Sigma^*$ is f_0-omitting for a subset C in $\mathcal{A}$, then $bin(w)$ is also f_0-omitting for C in $\mathcal{B}^*$.

Lemma 4.9. *If a word $w \in \Sigma^*$ is not f_0-omitting for a subset $C \subseteq Q_{\mathcal{A}^*} \setminus Q_{\mathrm{F}}$ in $\mathcal{A}^*$, then $bin(w)$ is not f_0-omitting for C in $\mathcal{B}^*$.*

Proof. Suppose that a prefix of w activates f_0 when applied to $\{q_0\}$; let ua be a shortest such a prefix, where $u \in \Sigma^*$ and $a \in \Sigma$. Since u is f_0-omitting, from Lemma 4.8, we know that

$T = \delta_{\mathcal{A}^*}(C, u) = \delta_{\mathcal{B}^*}(C, bin(u))$. If $a = \alpha$, then 0 applied to T activates f_0 in $\mathcal{B}^*$, as α does it in $\mathcal{A}^*$. If $a \in R$, then in $\mathcal{A}^*$, we can have $q_0 \in T$, a state $s_k^{i,j} \in T$, or a state $p_i \in T$ mapped to f_0 by the transition of a. In the first two cases, in $\mathcal{B}^*$, letter 1 activates f_0 from T, and in the third case, $bin(u)bin(a)$ activates f_0 from T. Both $bin(u)1$ and $bin(u)bin(a)$ are prefixes of $bin(w)$. □

From Lemma 4.8, one direction of the cofiniteness equivalence follows easily. For the second, we still need to consider binary words that are not complete encodings of words over the original alphabet. We also need to observe that Lemma 4.2 holds for $\mathcal{B}^*$ as well.

Lemma 4.10. *The language of $\mathcal{B}^*$ is cofinite if and only the language of $\mathcal{A}^*$ is cofinite.*

Proof. From Lemma 4.8 and by the fact that all not f_0-omitting words are accepted by $\mathcal{A}^*$, we know that if a word $w \in \Sigma^*$ is not accepted by $\mathcal{A}^*$, then $bin(w)$ is not accepted by $\mathcal{B}^*$. Thus, if infinitely many words are not accepted by $\mathcal{A}^*$, then the language of $\mathcal{B}^*$ is also not cofinite.

Assume now that the language of $\mathcal{B}^*$ is not cofinite. For an integer $t \geq m$, let $w' \in \{0,1\}^*$ be a binary word not accepted by $\mathcal{B}^*$ that has length at least t. Let u' be the longest prefix of w' that properly encodes a word $u \in \Sigma^*$, i.e., $bin(u) = u'$; then u' is shorter by at most $m-1$ letters than w'. Observe that Lemma 4.2 holds for $\mathcal{B}^*$; hence, since w' is not accepted, u' must be f_0-omitting. From Lemma 4.9, we know that u also must be f_0-omitting. The length of u is at least $(t-m+1)/m$, since each letter from Σ is encoded by at most m letters from $\{0,1\}$. Now we follow similarly as in the proof of Lemma 4.6. From Lemma 4.5, we conclude that u has a prefix of length at least $|u| - \ell$ that is not accepted. Hence, for every length k, we can choose a suitable $t \geq m(k+\ell)+m-1$ to find a non-accepted word of length at least k, thus there are infinitely many non-accepted words by $\mathcal{A}^*$. □

This finishes the reduction to the case of a binary DFA.

4.3. List of words.

Finally, we count the largest length and the number of words in the language accepted by $\mathcal{B}$.

Lemma 4.11. *In the language of $\mathcal{B}$, the maximum length of words is equal to $3\ell + m + 1$ and the number of words is at most $m\ell^2 + (1 + \ell m(2+\ell))(1+\ell) \in \Theta(m^2\ell^2)$.*

Proof. The maximum length of words accepted by our binary DFA $\mathcal{B}$ is equal to $3\ell + m + 1$, which is the length of the longest path from q_0 to a final state: $q_0 \xrightarrow{0^\ell} p_\ell \xrightarrow{1^m} s_\ell^{\ell,m} \xrightarrow{0^{\ell-1}} s_1^{\ell,m} \xrightarrow{0} q_{\mathrm{g}} \xrightarrow{0} f_0 \xrightarrow{0^\ell} f_\ell$.

For the number of words in the recognized language, we consider all final states. The first type of final states is setting states. Each setting state is reachable from q_0 by a unique word, which gives at most $m\ell^2$ words. The second type is forcing states. A forcing state f_i is reachable by different words, but all such words have a prefix for reaching f_0 followed by the unique suffix α^i to map f_0 to f_i. For the number of the first parts, observe that all the states in $Q_{\mathcal{B}} \setminus (Q_{\mathrm{F}} \cup \{q_{\mathrm{g}}, q_{\mathrm{s}}\})$, whose number is at most $1 + \ell m(1+\ell)$, are reachable by a unique word, and q_{g} is reachable by at most $m\ell$ words. Also, from each of these states, only one transition leads directly to f_0, with the possible exception of the last choice states $c^{i,m-1}$ when both rules r_{m-1} and r_m are illegal for an element p_i; however, in this case, we count fewer words than when they are legal, as the chain of setting states does not exist. Thus, combining with the second parts, there are at most $(1 + \ell m(2+\ell))(1+\ell)$ words of this type. □

We conclude with

Theorem 4.12. *Problem 1.1 is PSPACE-complete if L is a finite list of binary words.*

Using the construction, we can also infer the hardness for every fixed-sized alphabet larger than binary. For this, it is sufficient to add a suitable number of additional letters to $\mathcal{B}$ with the transitions mapping $Q_{\mathcal{B}} \setminus (Q_{\mathrm{F}} \cup \{q_{\mathrm{s}}\})$ to f_0 and mapping $Q_{\mathrm{F}} \cup \{q_{\mathrm{s}}\}$ to q_{s}, thus acting as rule letters that are always illegal in the set rewriting system.

5. The factor universality problem

We follow similarly as in Section 4. We reduce from Problem 3.3 (Emptying Set Rewriting) to Problem 1.2 (Factor Universality for a Finite Set of Words) when L is given as a finite list of binary words.

5.1. **DFA construction.** In the first step, we reduce to Problem 1.2 when L is specified as a DFA instead of a list of words.

We slightly modify the DFA construction $\mathcal{A}$ from Subsection 4.1 as follows. We remove the last forcing state f_ℓ and end the chain of the forcing states with $f_{\ell-1}$. Thus, the set of forcing states Q_{F} becomes $\{f_i \mid i \in \{0, 1, \ldots, \ell-1\}\}$, and we redefine the transition $\delta_{\mathcal{A}}(f_{\ell-1}, \alpha) = q_{\mathrm{s}}$. There are no other differences.

5.1.1. *The mechanism.* The idea of the modified construction is as follows.

As before, we build the NFA $\mathcal{A}^*$ recognizing the language L^*, where L is the language of $\mathcal{A}$. In the NFA $\mathcal{A}^*$, all states are reachable from the initial state q_0, and since we have also removed the sink state q_{s}, the NFA meets the criterion for factor universality from Proposition 3.4. Thus the language of $\mathcal{A}^*$ is factor universal if and only if there is a $Q_{\mathcal{A}^*}$-emptying word.

Simulating words in our NFA correspond to applications of sequences of rules in the set rewriting system in the same way as before in Subsection 4.1. In the modified construction, the forcing states have the property that whenever f_0 is activated, the only way to get rid of all active forcing states is to make the whole set Q_{P} active again. When f_0 is active, this is done by applying the word α^ℓ. In this way, the construction ensures that to map the whole set Q_{P} to the empty set, there must exist a simulating word whose sequences of rules is P-emptying in the set rewriting system. A special case occurs at the beginning, where we start with all the states $Q_{\mathcal{A}^*}$ and first have to reduce the active set of states to Q_{P}.

5.1.2. *Correctness.* The correctness is observed through the following lemmas.

Recall that a word $w \in \Sigma^*$ is *f_0-omitting for a subset $C \subseteq Q_{\mathcal{A}^*}$* if there is no prefix u of w such that $f_0 \in \delta_{\mathcal{A}^*}(C, u)$. Since in this problem our starting set is Q_{P} instead of $\{q_0\}$, we redefine that a word is simply *f_0-omitting* if it is f_0-omitting for Q_{P}.

We start with a simple observation, which follows directly from the construction and allows reducing the problem of emptying the whole set of states to emptying Q_{P}.

Lemma 5.1. *We have:*

(1) $\delta_{\mathcal{A}^*}(Q_{\mathcal{A}^*}, r_j^2) = \{f_0, q_0\}$ for each $r_j \in R$, and
(2) $\delta_{\mathcal{A}^*}(\{f_0\}, \alpha^\ell) = Q_{\mathrm{P}}$.

We show that when f_0 is activated, the only way to get rid of all forcing states is to activate the whole Q_{P} at some point.

Lemma 5.2. *Let $C \subseteq Q_{\mathcal{A}^*}$ be such that $f_0 \in C$, and let w be a word such that $\delta_{\mathcal{A}^*}(C, w) \cap Q_{\mathrm{F}} = \emptyset$. There exists a prefix u of w such that $Q_{\mathrm{P}} \subseteq \delta_{\mathcal{A}^*}(C, u)$.*

Proof. It is enough to prove the lemma for $C = \{f_0\}$. Let w be a shortest word such that $\delta_{\mathcal{A}^*}(\{f_0\}, w) \cap Q_{\mathrm{F}} = \emptyset$. Hence, there is no non-empty prefix u of w such that $f_0 \in \delta_{\mathcal{A}^*}(\{f_0\}, u)$, as otherwise also $\delta_{\mathcal{A}^*}(\{f_0\}, u') \cap Q_{\mathrm{F}} = \emptyset$ where $uu' = w$, and u' would be shorter than w.

Observe that $\delta_{\mathcal{A}^*}(\{f_0\}, \alpha^i) = \{f_i, q_0, p_1, \dots, p_i\}$ for all $0 \le i \le \ell - 1$. Thus, if w would start with $\alpha^i r_j$ for $0 \le i \le \ell - 1$ and some rule letter r_j, then the active state q_0 would be mapped to f_0 by the transition of r_j, which yields a contradiction. The only remaining possibility is that w begins with the prefix $u = \alpha^\ell$, which is such that $\delta_{\mathcal{A}^*}(\{f_0\}, u) = Q_{\mathrm{P}}$. □

We show the properties of a simulating word, in particular, that they correspond to rule applications in the set rewriting system. A special case occurs when we reach the empty set; then we do not activate the guard state.

Lemma 5.3. *Let $C \subseteq Q_{\mathrm{P}} \cup \{q_{\mathrm{g}}\}$ be such that $S = C \cap Q_{\mathrm{P}}$ is non-empty, and let $w = r_{i_1}\alpha^\ell \dots r_{i_k}\alpha^\ell$ ($k \ge 1$) be a simulating word for S. Then w is f_0-omitting and*

$$\delta_{\mathcal{A}^*}(C, w) = \begin{cases} (S \cdot r_{i_1} \cdot \dots \cdot r_{i_k}) \cup \{q_{\mathrm{g}}\}, & \text{if } S \cdot r_{i_1} \cdot \dots \cdot r_{i_{k-1}} \neq \emptyset \\ \emptyset = (S \cdot r_{i_1} \cdot \dots \cdot r_{i_k}), & \text{otherwise.} \end{cases}$$

Proof. In the case of $S \cdot r_{i_1} \cdot \dots \cdot r_{i_{k-1}} \neq \emptyset$, the proof is the same as that of Lemma 4.3, since for all $0 \le j \le k-1$, we have $S \cdot r_{i_1} \cdot \dots \cdot r_j \neq \emptyset$, thus all preconditions apply.

Otherwise, let $j < k$ be the smallest index such that the set $S \cdot r_{i_1} \cdot \dots \cdot r_{i_j}$ is empty. By the argument for the first case, we know that $\delta_{\mathcal{A}^*}(S, r_{i_1}\alpha^\ell \dots r_{i_j}\alpha^\ell) = \{q_{\mathrm{g}}\}$. Applying the next letter $r_{i_{j+1}}$ removes this single state, yielding the empty set. □

For the other direction, words that are f_0-omitting must involve simulating words. A special case occurs when we reach the empty set; then, there are no further restrictions, in particular, on that, we must continue with a simulating word.

Lemma 5.4. *Let $C = S \cup \{q_{\mathrm{g}}\}$, where $S \subseteq Q_{\mathrm{P}}$ is non-empty. If a word w is f_0-omitting for C, then either:*

(1) w is a prefix of a simulating word for S, or
(2) a prefix of w is a simulating word for S whose sequence of rules is S-emptying.

Proof. Following the proof of Lemma 4.4, we observe that the word w must start with $r_j\alpha^\ell$, unless it ends prematurely. Then, by Lemma 5.3, we have $\delta_{\mathcal{A}^*}(C, r_j\alpha^\ell) = (S \cdot r_j) \cup \{q_{\mathrm{g}}\}$. We apply this argument iteratively for the obtained subset and the remainder of w, until either w ends, in which case (1) holds, or the set of resulting active states becomes $\{q_{\mathrm{g}}\}$, in which case (2) holds. □

Using our ingredients collected so far, we can show that the existence of a Q_{P}-emptying word implies the existence of a P-emptying sequence of rules.

Lemma 5.5. *Let w be a word such that $\delta_{\mathcal{A}^*}(Q_{\mathrm{P}}, w) = \emptyset$. Then w contains a factor v which is a simulating word for Q_{P} whose sequence of rules is P-emptying.*

Proof. It is enough to prove the lemma for words w that do not have a non-empty prefix u such that $Q_{\mathrm{P}} \subseteq \delta_{\mathcal{A}^*}(Q_{\mathrm{P}}, u)$; otherwise, we can search for a factor v in w with u removed. Hence, by Lemma 5.2, w must be f_0-omitting. By Lemma 5.4, we have two possibilities (1) and (2). In case (2), we immediately know that w contains a prefix that is a simulating word for Q_{P} whose sequence of rules is P-emptying. In case (1), w is a prefix of a simulating word for Q_{P}. If w itself is a simulating word, let $v = w$; otherwise, write $w = vr_{i_{k+1}}\alpha^i$ for a simulating word $v = r_{i_1}\alpha^\ell \dots r_{i_k}\alpha^\ell$ for Q_{P} ($k \ge 0$) and some $0 \le i < \ell$. Let $C' = \delta_{\mathcal{A}^*}(Q_{\mathrm{P}}, v)$. By Lemma 5.3, $C' \subseteq Q_{\mathrm{P}} \cup \{q_{\mathrm{g}}\}$ and

$S' = C' \cap Q_{\mathrm{P}} = P \cdot r_{i_1} \cdot \ldots \cdot r_{i_k}$. If $S' \neq \emptyset$, then the action of the possibly remaining suffix $r_{i_{k+1}}\alpha^i$ do not map S' to $\emptyset$, which yields a contradiction with the assumption about w. Therefore, $S' = \emptyset$, thus the sequence of rules in v is P-emptying. □

Finally, we combine all the facts to show the equivalence between the reduced problems.

Lemma 5.6. *The following conditions are equivalent:*

(1) The permissive set rewriting system (P, R) admits a P-emptying sequence of rules.
(2) There exists a Q_{P}-emptying and f_0-omitting word for $\mathcal{A}^*$.
(3) There exists a $Q_{\mathcal{A}^*}$-emptying word for $\mathcal{A}^*$.

Proof. (1) ⇒ (2): Suppose that for the set rewriting system there is a sequence of rules $r_{i_1}, \ldots, r_{i_k}$ that is P-emptying. We take the word $w = r_1\alpha^\ell \ldots r_k\alpha^\ell$, which is a simulating word for Q_{P}. By Lemma 5.3, we conclude that $\delta_{\mathcal{A}^*}(Q_{\mathrm{P}}, w) \subseteq \{q_{\mathrm{g}}\}$. Thus, wr_1 is a Q_{P}-emptying and f_0-omitting word.
(2) ⇒ (3): If w is a Q_{P}-emptying word, then, by Lemma 5.1, $\delta_{\mathcal{A}^*}(Q_{\mathcal{A}^*}, r_1^2\alpha^\ell w) = \emptyset$.
(3) ⇒ (1): If there exists a $Q_{\mathcal{A}^*}$-emptying word $w \in \Sigma^*$, then, in particular, $\delta_{\mathcal{A}^*}(Q_{\mathrm{P}}, w) = \emptyset$. By Lemma 5.5, w contains a factor v which is a simulating word for Q_{P} whose sequence of rules is P-emptying. □

We conclude this part with

Theorem 5.7. *Problem 1.2 is PSPACE-hard if L is specified by a DFA over a given alphabet.*

5.2. **Binarization.** We construct a binary DFA $\mathcal{B}$ exactly in the same way as in Subsection 4.2 and use the same notation and the same binary encoding *bin*. Note that the criterion for the factor universality from Proposition 3.4 still holds for $\mathcal{B}^*$.

We observe that Lemma 4.8 and Lemma 4.9 hold also in this case; it is because both constructions of $\mathcal{B}^*$ differ only on the set Q_{F}, whose transitions are irrelevant for the observations. Also, Lemma 5.2 holds for $\mathcal{B}^*$ as it holds for $\mathcal{A}^*$:

Lemma 5.8. *Let $C \subseteq Q_{\mathcal{B}^*}$ be such that $f_0 \in C$, and let $w \in \{0,1\}^*$ be a word such that $\delta_{\mathcal{B}^*}(C, w) \cap Q_{\mathrm{F}} = \emptyset$. There exists a prefix u of w such that $Q_{\mathrm{P}} \subseteq \delta_{\mathcal{B}^*}(C, u)$.*

Proof. The proof is the same as that of Lemma 5.2: it is sufficient to replace each α with 0 and each r_j with 1. □

Now we show the equivalence of the existence of Q_{P}-emptying words in both $\mathcal{A}^*$ and $\mathcal{B}^*$.

Lemma 5.9. *There is a Q_{P}-emptying and f_0-omitting word for $\mathcal{A}^*$ if and only if there is such a word for $\mathcal{B}^*$. In particular, if $w' \in \{0,1\}^*$ is such a word for $\mathcal{B}^*$, then $w'0 = bin(w)$ for some word $w \in \Sigma^*$ with this property for $\mathcal{A}^*$.*

Proof. Let w be a Q_{P}-emptying and f_0-omitting word for $\mathcal{A}^*$. From Lemma 4.8, we know that $bin(w)$ is f_0-omitting and such that $\delta_{\mathcal{B}^*}(Q_{\mathrm{P}}, bin(w)) = \delta_{\mathcal{A}^*}(Q_{\mathrm{P}}, w) = \emptyset$.

Conversely, assume that there is a Q_{P}-emptying and f_0-omitting binary word w' for $\mathcal{B}^*$. Since $\delta_{\mathcal{B}^*}(Q_{\mathrm{P}}, w') = \emptyset$, we know that $w'0$ has the same properties. Furthermore, $w'0$ must be such that $w'0 = bin(w)$ for some word $w \in \Sigma^*$ because 0 can appear only at the end in $bin(a)$ for each $a \in \Sigma$. Then, from Lemma 4.9, w must be f_0-omitting. From Lemma 4.8, we conclude that w has to be also Q_{P}-emptying as $w'0$ is. □

Finally, we show the equivalence of the existence of a $Q_{\mathcal{B}^*}$-emptying word and of a Q_{P}-emptying word, which finishes the reduction to the case of a binary DFA.

Lemma 5.10. *For $\mathcal{B}^*$, there exists a Q_{P}-emptying and f_0-omitting word if and only if there exists a $Q_{\mathcal{B}^*}$-emptying word. In particular, a $Q_{\mathcal{B}^*}$-emptying word contains a factor that is Q_{P}-emptying and f_0-omitting.*

Proof. Assume that there is a Q_{P}-emptying word w. We have $\delta_{\mathcal{B}^*}(Q_{\mathcal{B}^*}, 1^{m+1}) = \{f_0, q_0\}$ and $\delta_{\mathcal{B}^*}(\{f_0\}, 0^\ell) = Q_{\mathrm{P}}$. Thus, $\delta_{\mathcal{B}^*}(Q_{\mathcal{B}^*}, 1^{m+1}0^\ell w) = \emptyset$.

Conversely, let w be a $Q_{\mathcal{B}^*}$-emptying word. Let u be the longest prefix of w such that $Q_{\mathrm{P}} \subseteq \delta_{\mathcal{B}^*}(Q_{\mathcal{B}^*}, u)$, and write $w = uv$. By Lemma 5.8, v has to be f_0-omitting, as otherwise u would be longer. Hence, v is a $Q_{\mathcal{P}}$-emptying and f_0-omitting word, and it is a factor of w. □

5.3. **List of words.**

Lemma 5.11. *in the language of $\mathcal{B}$, the maximum length of words is equal to $3\ell + m$ and the number of words is at most $m\ell^2 + (1 + \ell m(2 + \ell))\ell$.*

Proof. We count the maximum length and the words as in the proof of Lemma 4.11, taking into account that the chain of forcing states is shorter by 1. □

We conclude with

Theorem 5.12. *Problem 1.2 is PSPACE-complete when the alphabet is binary.*

As for the Frobenius monoid problem, in the same way, by adding a suitable number of letters, it is possible to show the hardness for every fixed-sized alphabet larger than binary.

6. Lower bounds

6.1. **The longest omitted words.** It is known that for each odd integer $n \geq 5$, there exists a set of binary words L, which have length at most n, such that L^* is cofinite and the longest words not in L^* are of length $\Omega(n^2 2^{\frac{n}{2}})$ [16]. However, the constructed L contains exponentially many words in n, thus a large lower bound in terms of the size of L could not be inferred.

We show a subexponential lower bound in $|L|$ and $\|L\|_1$ on the length of the longest words not in L^* when L^* is cofinite. The idea is to construct a list of binary words from a mortal set rewriting system whose longest legal sequences of rules have an exponential length (Theorem 3.1).

Theorem 6.1. *There exists an infinite family of finite sets L of binary words such that L^* is cofinite and the longest words not in L^* are of length at least $\frac{\|L\|_\infty - 1}{4} \cdot 2^{\frac{\|L\|_\infty - 1}{4}}$, and this length is $2^{\Omega(\sqrt[4]{|L|})}$ in terms of $|L|$ and $2^{\Omega(\sqrt[5]{\|L\|_1})}$ in terms of $\|L\|_1$.*

Proof. For an $n \geq 2$, from Theorem 3.1, we take the set rewriting system (P, R) with $|P| = |R| = n$ and the subset S meeting the bound $2^n - 2$. Then we use the construction from Section 4 to create a binary DFA $\mathcal{B}$ and its list of binary words L. Since the set rewriting system is mortal, L^* is cofinite.

From Lemma 4.11 ($\ell = m = n$), the length of the longest words in L is equal to $4n + 1$ and there are at most $n^3 + (1 + n^2(2 + n))(1 + n) = n^4 + 4n^3 + 2n^2 + n + 1$ words, thus $|L| \in \mathcal{O}(n^4)$ and $\|L\|_1 \in \mathcal{O}(n^5)$.

We take a binary simulating word w' with the longest possible legal sequence of rules for S, thus also for some singleton $S' \subseteq S$. From Lemma 4.3 and Lemma 4.8, we know that $0^i w' \notin L^*$, for $1 \leq i \leq n$ corresponding to the initial singleton S'. For $n \geq 2$, we can lower bound the length of the binary encoding of each rule letter by 2. Since the longest possible legal sequence of rules has length $2^n - 2$ and one rule application corresponds to at least $n + 2$ letters (i.e., $\mathit{bin}(r_j)0^n$ for a rule letter r_j), the length of the word $0^i w'$ is at least $(2^n - 2) \cdot (n+2) + 1$. For $n \geq 2$, we have $(2^n - 2) \cdot (n+2) + 1 \geq 2^n \cdot n$.

Since $n = \frac{\|L\|_\infty - 1}{4}$, $n \in \Omega(\sqrt[4]{|L|})$, and $n \in \Omega(\sqrt[5]{\|L\|_1})$, respectively, the length of the word $0^i w'$ is as in the theorem. □

6.2. **The shortest incompletable words.** We show that when L^* is not factor universal, the length of the shortest incompletable words can be exponential in $\|L\|_\infty$ and subexponential in $|L|$ and $\|L\|_1$. The idea is to construct a list of binary words from a permissive set rewriting system whose shortest legal sequences of rules that are P-emptying are of exponential length (Theorem 3.6).

Theorem 6.2. *There exists an infinite family of finite sets L of binary words such that the shortest incompletable words are of length at least $\frac{\|L\|_\infty}{4} \cdot 2^{\frac{\|L\|_\infty}{4}}$, and this length is $2^{\Omega(\sqrt[4]{|L|})}$ in terms of $|L|$ and $2^{\Omega(\sqrt[5]{\|L\|_1})}$ in terms of $\|L\|_1$.*

Proof. For an $n \geq 2$, from Theorem 3.6, we take the set rewriting system (P, R) with $|P| = |R| = n$ where the shortest P-emptying rule sequences have length equal to $2^n - 1$. Then we apply the construction from Subsection 5 to create a binary DFA $\mathcal{B}$ and its list of binary words L. Since there exists a P-emptying sequence of rules, we know that there exists a $Q_{\mathcal{B}^*}$-emptying word in $\mathcal{B}^*$, thus L^* is not factor universal.

We show a lower bound on the length of the shortest incompletable words. Let $w' \in \{0,1\}^*$ be an incompletable word. From the criterion from Proposition 3.4, w' is also $Q_{\mathcal{B}^*}$-emptying in $\mathcal{B}^*$. From Lemma 5.10, we know that w' contains a factor u' that is Q_{P}-emptying and f_0-omitting for Q_{P}. From Lemma 5.9, we know that the word $u \in \Sigma^*$ such that $bin(u) = u'0$ is Q_{P}-emptying in $\mathcal{A}^*$. By Lemma 5.5, u contains as a factor a simulating word v whose sequence of rules is P-emptying. Since the shortest such a sequence of rules has length $2^n - 1$, the word v and also u have length at least $(2^n - 1) \cdot (n+2)$. Moreover, both these words contain at least $(2^n - 1)$ rule letters, as we have taken such a set rewriting system. Since, for $n \geq 2$, each rule letter is encoded by at least two binary symbols, we conclude that u', thus also w', has length at least $(2^n - 1) \cdot (n+2) - 1$. We have $(2^n - 1) \cdot (n+2) - 1 \geq 2^n \cdot n$.

Since $n = \frac{\|L\|_\infty}{4}$, $n \in \Omega(\sqrt[4]{|L|})$, and $n \in \Omega(\sqrt[5]{\|L\|_1})$, respectively, the length of every Q_{P}-emptying word is as in the theorem. □

7. Upper bounds

We show algorithms and upper bounds on the related lengths for both problems, which are exponential only in $\|L\|_\infty$ while remain polynomial in $|L|$ thus also in $\|L\|_1$.

For the Frobenius monoid problem, the upper bound $\frac{2}{2|\Sigma|-1}(2^{\|L\|_\infty}|\Sigma|^{\|L\|_\infty} - 1)$ on the length of the longest words not in L^* when L^* is cofinite was known [16, Theorem 6.1]. We show an upper bound that involves both $\|L\|_\infty$ and $|L|$.

Theorem 7.1. *Problem 1.1 can be solved in time exponential in $\|L\|_\infty$ while polynomial in $|L|$. If L^* is cofinite, then the longest words not in L^* have length at most $|L| \cdot (2^{\|L\|_\infty} - 1) - 1$.*

Proof. The proof uses a similar idea to that in [16, Theorem 6.1] ([25, Theorem 3.2.5]), but involves the number of words $|L|$. We can assume that $\varepsilon \notin L$ and $L \neq \emptyset$.

We construct a DFA $\mathcal{A} = (Q_{\mathcal{A}}, \Sigma, \delta_{\mathcal{A}}, q_\varepsilon, F_{\mathcal{A}})$ recognizing L in a standard way that it forms a tree. Thus $Q_{\mathcal{A}} = \{q_u \mid u \text{ is a prefix of a word in } L\} \cup \{q_{\mathrm{s}}\}$, where q_{s} is the unique dead state. For $q_u \in Q_{\mathcal{A}}$ and $a \in \Sigma$, we define $\delta_{\mathcal{A}}(q_u, a) = q_{ua}$ if ua is a prefix of a word in L and $\delta_{\mathcal{A}}(q_u, a) = q_{\mathrm{s}}$, otherwise. Then each word $w \in L$ has the action mapping the initial state q_0 to a distinct state.

By the standard construction for the Kleene star (Section 2), we construct an NFA $\mathcal{A}^* = (Q_{\mathcal{A}^*}, \Sigma, \delta_{\mathcal{A}^*}, q_\varepsilon, \{q_\varepsilon\})$ recognizing L^*. Hence $\mathcal{A}^*$ is $\mathcal{A}$ with added an ε-transition from every final state to q_ε and with the dead state q_s removed. For a word $w \in \Sigma^*$, let the set of *active* states be $\delta_{\mathcal{A}^*}(\{q_\varepsilon\}, w)$.

We are going to observe that for every word w, there are no more than $|Q_{\mathcal{A}^*}| \cdot 2^{\|L\|_\infty} + 1$ active states. We define the *level* of a state $q_u \in Q_{\mathcal{A}^*}$ to be the length of u. For every state q_u and a letter a, the set $\delta_{\mathcal{A}^*}(\{q_u\}, a)$ contains q_{ua}, if this state exists, and q_ε, if q_{ua} was final in the DFA. Hence, for a subset $C \subseteq Q_{\mathcal{A}*}$ with at most one state for each level, the transition of every letter, thus also the action of every word, preserves this property. Since we start with $\{q_\varepsilon\}$, after reading any word for every level at most one state can be active. Moreover, if a state q_u is the active state with the largest level, then the set of all the states with smaller levels that can be active is determined, as they are the states $q_{u'}$ with u' being a prefix of u. We call them the *possibly active states* of q_u. Hence, the number of reachable subsets from $\{q_0\}$ is bounded by the number of the choices for the set of possibly active states and for its subset with actually active states.

Note that for a state q_u where u is a proper prefix of some word $w \in L$, the set of possibly active states of q_u is contained in that set of q_w. Thus it is sufficient to count only the sets of the possibly active states of q_u with $u \in L$.

Also, the initial state q_ε is active if and only if a final state in the DFA is active, with the exception of the initial subset $\{q_\varepsilon\}$. Altogether, we have at most $|L|$ choices for the set of possibly active states combined with at most $2^{\|L\|_\infty} - 1$ choices for the non-empty subset of actually active states. Additionally, there are the empty set and the initial singleton. We obtain the upper bound $2 + |L| \cdot (2^{\|L\|_\infty} - 1)$ on the number of reachable subsets from $\{q_\varepsilon\}$.

The problem of whether $\mathcal{A}^*$ recognizes a non-cofinite language is equivalent to whether in the space of reachable subsets there exists a cycle such that a subset without the unique final state q_ε is reachable from it. Thus, we can check this in time exponential in $\|L\|_\infty$ and polynomial in $|L|$.

If the language is cofinite, the empty subset is not reachable (which was counted in the upper bound), and there exists a reachable cycle from which we can reach only subsets with the final state. Thus, the longest words not in L^* have length at most $|L| \cdot (2^{\|L\|_\infty} - 1) - 1$. □

For the factor universality problem, only the trivial upper bound $2^{\|L\|_1 - \|L\|_\infty + 1}$ was known [12]. Note that it is doubly-exponential if represented only in terms of $\|L\|_\infty$.

Theorem 7.2. *Problem 1.2 can be solved in time exponential in $\|L\|_\infty$ while polynomial in $|L|$. If the set $L \neq \emptyset$ is not complete, then the shortest incompletable words have length at most $\|L\|_\infty + |L| \cdot 2^{\|L\|_\infty}$.*

Proof. The statement is trivial when $L = \{\varepsilon\}$, and we can assume $\varepsilon \notin L$. We construct a DFA $\mathcal{A}$ and an NFA $\mathcal{A}^*$ for L^* as in the proof of Theorem 7.1. The language L^* is factor universal if and only if there exists a $Q_{\mathcal{A}^*}$-emptying word (Proposition 3.4).

We follow similarly as in the proof of Theorem 7.1, obtaining an upper bound on the set of reachable subsets from $Q_{\mathcal{A}^*}$. Note that for every word w of length at least $\|L\|_\infty$, in $\delta_{\mathcal{A}^*}(Q_{\mathcal{A}^*}, w)$ there is at most one state for each level. Thus, when restricted to such words, there are at most $1 + |L| \cdot (2^{\|L\|_\infty} - 1)$ reachable subsets (not counting $\{q_\varepsilon\}$ this time, since it is not reachable from $Q_{\mathcal{A}^*}$ as long as $L \nsubseteq \{\varepsilon\}$). Since we start from $Q_{\mathcal{A}^*}$, at the beginning there could be more reachable subsets by words shorter than $\|L\|_\infty$.

If there exists a $Q_{\mathcal{A}^*}$-emptying word w, then for every word u, the word uw is also $Q_{\mathcal{A}^*}$-emptying. Hence, to solve the problem, we can start from an arbitrary word u of length $\|L\|_\infty$, and then check the reachability of the empty set. The length of the shortest $Q_{\mathcal{A}^*}$-emptying words is at most $\|L\|_\infty + |L| \cdot (2^{\|L\|_\infty} - 1)$. $\square$

Under a fixed-sized alphabet (as otherwise $\|L\|_1$ can be arbitrarily large with respect to $\|L\|_\infty$), we have $|L| \leq |\Sigma|^{\|L\|_\infty}$. We conclude that $2^{\mathcal{O}(\|L\|_\infty)}$ is a tight upper bound on the lengths related to both problems.

Acknowledgments

We thank Amir M. Ben-Amram for the idea of a simpler way for proving Theorem 3.2. We also thank all the anonymous reviewers for their comments. This work was supported by the National Science Centre, Poland under project number 2017/25/B/ST6/01920.

References

[1] A. V. Aho and J. E. Hopcroft. *The Design and Analysis of Computer Algorithms.* Addison-Wesley Longman Publishing Co., Inc., 1st edition, 1974.

[2] M Beck, R. Diaz, and S. Robins. The Frobenius Problem, Rational Polytopes, and Fourier–Dedekind Sums. *Journal of Number Theory*, 96:1–21, 2002.

[3] A. M. Ben-Amram. Mortality of iterated piecewise affine functions over the integers: Decidability and complexity. *Computability*, 4(1):19–56, 2015.

[4] J. Berstel, D. Perrin, and C. Reutenauer. *Codes and Automata.* Encyclopedia of Mathematics and its Applications. Cambridge University Press, 2010.

[5] V. D. Blondel, J. Cassaigne, and C. Nichitiu. On the presence of periodic configurations in Turing machines and in counter machines. *Theoretical Computer Science*, 289(1):573–590, 2002.

[6] A. Boccuto and A. Carpi. On the length of uncompletable words in unambiguous automata. *RAIRO-Theor. Inf. Appl.*, 53(3–4):115–123, 2019.

[7] A. Brauer. On a problem of partitions. *Amer. J. Math.*, 64(1):299–312, 1942.

[8] A. Carpi and F. D'Alessandro. On incomplete and synchronizing finite sets. *Theoretical Computer Science*, 664:67–77, 2017.

[9] J. Clément, J.-P. Duval, G. Guaiana, D. Perrin, and G. Rindone. Parsing with a finite dictionary. *Theoretical Computer Science*, 340(2):432–442, 2005.

[10] A. L. Dulmage and N. S. Mendelsohn. Gaps in the exponent set of primitive matrices. *Illinois J. Math.*, 8(4):642–656, 1964.

[11] G. Fici, E. V. Pribavkina, and J. Sakarovitch. On the Minimal Uncompletable Word Problem. `https://arxiv.org/abs/1002.1928`, 2010.

[12] V. V Gusev and E. V. Pribavkina. On Non-complete Sets and Restivo's Conjecture. In Giancarlo Mauri and Alberto Leporati, editors, *DLT*, pages 239–250. Springer, 2011.

[13] J. Incerpi and R. Sedgewick. Improved upper bounds on shellsort. *Journal of Computer and System Sciences*, 31(2):210–224, 1985.

[14] S. Julia, A. Malapert, and J. Provillard. A Synergic Approach to the Minimal Uncompletable Words Problem. *Journal of Automata, Languages and Combinatorics*, 22(4):271–286, 2017.

[15] R. Kannan. Lattice translates of a polytope and the Frobenius problem. *Combinatorica*, 12:161–177, 1992.

[16] J.-Y. Kao, J. Shallit, and Z. Xu. The Frobenius Problem in a Free Monoid. In Susanne Albers and Pascal Weil, editors, *STACS*, volume 1 of *LIPIcs*, pages 421–432, 2008.

[17] L. Kari and Z. Xu. De bruijn sequences revisited. *International Journal of Foundations of Computer Science*, 23(06):1307–1321, 2012.

[18] S. Kiefer and C. Mascle. On Finite Monoids over Nonnegative Integer Matrices and Short Killing Words. In *STACS*, LIPIcs, pages 43:1–43:13, 2019.

[19] M. Mika and M. Szykuła. The Frobenius and factor universality problems of the free monoid on a finite set of words. `https://arxiv.org/abs/1902.06702`, 2019.

[20] J. Nicholson and N. Rampersad. The Frobenius problem for the shuffle operation. *Semigroup Forum*, 96:160–177, 2018.

[21] J. L. Ramírez-Alfonsín. Complexity of the Frobenius problem. *Combinatorica*, 16:143–147, 1996.

[22] N. Rampersad, J. Shallit, and Z. Xu. The Computational Complexity of Universality Problems for Prefixes, Suffixes, Factors, and Subwords of Regular Languages. *Fundamenta Informaticae*, 116(1–4):223–236, 2012.

[23] A. Restivo. Some remarks on complete subsets of a free monoid. In *Quaderni de "La Ricerca Scientifica". Non-Commutative Structures in Algebra and Geometric Combinatorics*, volume 109, pages 19–25. CNR Roma, 1981.

[24] J. Shallit. Open problems in automata theory: an idiosyncratic view, LMS Keynote Address in Discrete Mathematics, BCTCS 2014, April 10 2014, Loughborough, England. https://cs.uwaterloo.ca/~shallit/Talks/bc4.pdf.

[25] Z. Xu. *The Frobenius Problem in a Free Monoid*. PhD thesis, University of Waterloo, Waterloo, Ontario, Canada, 2009.

[26] Z. Xu and J. Shallit. An NP-hardness Result on the Monoid Frobenius Problem. https://arxiv.org/abs/0805.4049, 2008.

[27] S. Yu, Q. Zhuang, and K. Salomaa. The state complexities of some basic operations on regular languages. *Theoretical Computer Science*, 125(2):315–328, 1994.

Appendix

Large length of the shortest incompletable words. We define explicitly the family from the proof Theorem 6.1 of sets of words L for which the shortest incompletable words in L^* are of exponential length $\frac{\|L\|_\infty}{4} \cdot 2^{\frac{\|L\|_\infty}{4}}$ in terms of $\|L\|_\infty$ and subexponential length $2^{\Omega(\sqrt[5]{\|L\|_1})}$ in terms of $\|L\|_1$.

For a given $n \geq 2$, the words in L are as follows. The paths in the construction from the initial state to a final state, which correspond to words in L, are also listed. We rename the elements in the set $P = \{b_0, b_1, \ldots, b_{n-1}\}$ from the set rewriting system in the proof to the elements from $\{p_1, p_2, \ldots, p_n\}$ such that $b_i = p_{i+1}$ as in the reduction. In this way, the construction keeps the property that if $s_k^{i,j}$ is final and p_i is active, then after 1^j00^n (or 1^j0^n if $j = n$), p_k will be active.

The words coming from final states f_x for $x \in \{0, 1, \ldots, n-1\}$:

- 10^x for $x \in \{0, \ldots, n-1\}$; $(q_0 \xrightarrow{1} f_0 \xrightarrow{0^x} f_x)$
- 0^n00^x for $x \in \{0, \ldots, n-1\}$; $(q_0 \xrightarrow{0^n} p_n \xrightarrow{0} f_0 \xrightarrow{0^x} f_x)$
- $0^i1^j00^k10^x$ for $i \in \{1, \ldots, n\}$, $j \in \{1, \ldots, n-1\}$, $k \in \{0, \ldots, n-1\}$, and $x \in \{0, \ldots, n-1\}$; $(q_0 \xrightarrow{0^i} p_i \xrightarrow{1^j0} s_n^{i,j} \xrightarrow{0^k} s_{n-k}^{i,j} \xrightarrow{1} f_0 \xrightarrow{0^x} f_x)$
- $0^i1^n0^k10^x$ for $i \in \{1, \ldots, n\}$, $k \in \{0, \ldots, n-1\}$, and $x \in \{0, \ldots, n-1\}$; $(q_0 \xrightarrow{0^i} p_i \xrightarrow{1^n} s_n^{i,n} \xrightarrow{0^k} s_{n-k}^{i,n} \xrightarrow{1} f_0 \xrightarrow{0^x} f_x)$
- $0^i1^j00^n00^x$ for $i \in \{1, \ldots, n\}$, $j \in \{1, \ldots, n-1\}$, and $x \in \{0, \ldots, n-1\}$; $(q_0 \xrightarrow{0^i} p_i \xrightarrow{1^j0} s_n^{i,j} \xrightarrow{0^n} q_{\mathrm{g}} \xrightarrow{0} f_0 \xrightarrow{0^x} f_x)$
- $0^i1^n0^n00^x$ for $i \in \{1, \ldots, n\}$ and $x \in \{0, \ldots, n-1\}$; $(q_0 \xrightarrow{0^i} p_i \xrightarrow{1^n} s_n^{i,n} \xrightarrow{0^n} q_{\mathrm{g}} \xrightarrow{0} f_0 \xrightarrow{0^x} f_x)$

The words coming from the final setting states corresponding to the transition $r_j(p_j) = \{p_i \mid i \in \{0, 1, 2, \ldots, j-1\}\}$:

- $0^j1^j00^{n-k}$ for $j \in \{1, \ldots, n-1\}$ and $k \in \{1, \ldots, j-1\}$; $(q_0 \xrightarrow{0^j} p_j \xrightarrow{1^j0} s_n^{j,j} \xrightarrow{0^{n-k}} s_k^{j,j})$
- $0^n1^n0^{n-k}$ for $k \in \{1, \ldots, n-1\}$; $(q_0 \xrightarrow{0^n} p_n \xrightarrow{1^n} s_n^{n,n} \xrightarrow{0^{n-k}} s_k^{n,n})$

The words coming from the final setting states corresponding to the transition $r_j(p_i) = P$ for $i \in \{0, 1, 2, \ldots, j-1\}$:

- $0^i1^j00^{n-k}$ for $j \in \{1, 2, \ldots, n-1\}$, $i \in \{1, \ldots, j-1\}$, and $k \in \{1, 2, \ldots, n\}$; $(q_0 \xrightarrow{0^i} p_i \xrightarrow{1^j0} s_n^{i,j} \xrightarrow{0^{n-k}} s_k^{i,j})$
- $0^i1^n0^{n-k}$ for $i \in \{1, \ldots, n-1\}$ and $k \in \{1, 2, \ldots, n\}$; $(q_0 \xrightarrow{0^i} p_i \xrightarrow{1^n} s_n^{n,n} \xrightarrow{0^{n-k}} s_k^{n,n})$

The words coming from the final setting states corresponding to the transition $r_j(p_i) = \{p_i\}$ for $i \in \{j+1, j+2, \ldots, n-1\}$:

- $0^i1^j00^{n-i}$ for $j \in \{1, 2, \ldots, n-1\}$ and $i \in \{j+1, \ldots, n\}$; $(q_0 \xrightarrow{0^i} p_i \xrightarrow{1^j0} s_n^{i,j} \xrightarrow{0^{n-i}} s_i^{i,j})$

A program generating these examples is also available at [19] as a source file.