The Frobenius and factor universality problems of the free monoid on a finite set of words

(Problem Frobeniusa oraz uniwersalności faktorowej wolnego monoidu na skończonym zbiorze słów)

Maksymilian Mika

Praca magisterska

Promotor: dr Marek Szykuła

Uniwersytet Wrocławski
Wydział Matematyki i Informatyki
Instytut Informatyki

Abstract

We settle complexity questions of two problems about the free monoid L^* generated by a finite set L of words over an alphabet Σ. The first one is the *Frobenius monoid problem*, which is whether for a given finite set of words L, the language L^* is cofinite. The open question concerning its computational complexity was originally posed by Shallit and Xu in 2009. The second problem is whether L^* is *factor universal*, which means that every word over Σ is a factor of some word from L^*. It is related to the longstanding Restivo's open question from 1981 about the maximal length of the shortest words which are not factors of any word from L^*. We show that both problems are PSPACE-complete, which holds even if the alphabet is binary. Additionally, we exhibit families of sets L that show exponential (in the sum of the lengths of words in L or in the length of the longest words in L) worst-case lower bounds on the lengths related to both problems: the length of the longest words not in L^* when L^* is cofinite, and the length of the shortest words that are not a factor of any word in L^* when L^* is not factor universal. The second family essentially settles in the negative the Restivo's conjecture and its weaker variations. As an auxiliary tool, we introduce the concept of *set rewriting systems*. Finally, we note upper bounds on the computation time and the length for both problems, which are exponential only in the length of the longest words in L.

Streszczenie

Odpowiadamy na pytania o złożoność dwóch problemów związanych z wolnym monoidem L^*, stworzonym nad skończonym zbiorem L słów nad alfabetem Σ. Pierwszy z nich to problem Frobeniusa w monoidzie, polegający na stwierdzeniu czy dla danego skończonego zbioru słów L, język L^* jest dopełnieniem języka skończonego. Otwarte pytanie dotyczące jego złożoności obliczeniowej zostało postawione przez Shallita i Xu w 2009 roku. Drugi problem to sprawdzenie czy L^* jest faktorowo uniwersalny, co oznacza, że każde słowo nad alfabetem Σ jest podsłowem jakiegoś słowa z L^*. Problem ten związany jest z długo otwartym problemem postawionym przez Restivo w roku 1981, dotyczącym maksymalnej długości najkrótszych słów, które nie są podsłowami żadnego słowa z L^*. Pokazujemy, że oba problemy są PSPACE-zupełne, co ma miejsce nawet w przypadku alfabetu binarnego. Dodatkowo definiujemy rodziny zbiorów L, które pokazują wykładniczą (w sensie sumy długości słów w L lub długości najdłuższego słowa w L), najgorszą, dolną granicę na długości związane z oboma problemami: długość najdłuższego słowa nienależącego do L^*, gdy L^* jest dopełnieniem języka skończonego, oraz długość najkrótszego słowa, które nie jest podsłowem żadnego ze słów w L^*, gdy L^* nie jest faktorowo uniwersalny. Druga rodzina obala hipotezę Restivo i jego słabsze warianty o wielomianowej długości. Jako narzędzie pomocnicze dla naszych konstrukcji wprowadzamy koncepcję *systemu przepisywania zbiorów*. Na zakończenie pokazujemy górne ograniczenia na złożoność czasową jak i długość dla obu problemów, które są wykładnicze jedynie zależnie od długości najdłuższego słowa w L.

Acknowledgments

Many thanks to my advisor Marek Szykuła for being a great help both in solving the problems as well as in writing down the solutions. I also thank Amir M. Ben-Amram for the idea of a simpler way for proving Theorem 2.2. This work was supported by the National Science Centre, Poland under project number 2017/25/B/ST6/01920.

Contents

Chapter 1

Introduction

Given a set of words L over an alphabet Σ, the language L^* (Kleene star or free monoid) contains all finite strings built by concatenating any number of words from L. In general, we can think about L as a dictionary and L^* as the language of all available phrases. One of the most basic question that one could ask is whether L generates all words over the alphabet Σ of L. The answer is, however, trivial, because this is the case if and only if L contains all single letters $a \in \Sigma$. Thus, more interesting relaxed universality properties are considered. In this paper, we consider two famous problems of this kind and settle their complexity.

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 $x = c_1 x_1 + \cdots + c_k x_k$. 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 $\gcd(x_1, \cdots, x_k) = 1$.

The Frobenius problem was extensively studied and found applications across many fields, e.g., to primitive sets of matrices [9], to the Shellsort algorithm [11], and to counting points in polytopes [2]. The problem of computing the largest non-expressible integer is NP-hard [16] when the integers are given in binary, and it can be solved polynomially if the number k of given integers is fixed [12].

A generalization of the Frobenius problem to the setting of free monoids was introduced by Kao, Shallit, and Xu [13]. 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 that whether all except 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 an 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. There are also efficient algorithms for checking whether a *given* word is in L^* [8].

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 $L = \{0, 01, 10, 11\}$ 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 actually 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; since we can build all words of length 2 and 3 over the alphabet $\{0, 1\}$, and $\gcd(2, 3) = 1$, we know that L^* must contain all sufficiently long words.

The problem can be seen as *almost universality* of the language L^*. It models a situation where we consider whether a given dictionary is sufficient to generate all sufficiently long sequences. For example, consider sound synthesis. A common method there is *unit selection*, which is generating the sound by concatenating various recorded sequences [21]. In a simple setting, if we do not care about short sequences (as for them we require all single-sound samples anyway), testing whether a given sound bank is strong enough to generate everything is equivalent to the Frobenius monoid problem.

Kao, Shallit, and Xu [13, 22] showed that, in particular, if L^* is cofinite, then the longest non-expressible words can be exponentially long in the length of the longest words from L. This is in contrast with the classical Frobenius problem, where the largest non-expressible integer is bounded quadratically in the largest given integer [6]. In 2009, Shallit and Xu posed the open question about the computational complexity of determining whether L^* is cofinite [22]. They also proved that it is NP-hard and in PSPACE when L is given as a regular expression [23]. The question about the computational complexity appears on the Shallit's list of open problems [20].

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 $v, v' \in \Sigma^*$.

Problem 1.2 (Factor Universality for a Finite Set of Words)**.** *Given a finite set of words L over an alphabet Σ, is every word over Σ a factor of a word from L^*?*

Sets L such that the language of all factors of the words in L^* is 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 word 100010001 is not completable. If we want to create 1 in the middle, we have to use either 10 or 01. In each case, one of the adjacent 0s is also consumed, so we cannot use word 000.

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 [18], who conjectured that if a set L is not complete, then the shortest incompletable words have length at most $2||L||^2_{\max}$, where $||L||_{\max}$ is the length of the longest words in L. The conjecture in this form turned out to be false [10] ($5||L||^2_{\max} - \mathcal{O}(||L||_{\max})$ is a lower bound), but the relaxed question whether there is a quadratic upper bound remained open and became one of the longstanding unsolved problems in automata theory.

There is a trivial exponential upper bound in the sum of the lengths of words in L. A sophisticated experimental research [19] suggested that the tight upper bound is unlikely quadratic and may be exponential. On the other hand, a polynomial upper bound $\mathcal{O}(||L||^5_{\text{sum}})$, where $||L||_{\text{sum}}$ is the sum of the lengths of all words in L, was derived for the subclass of sets L called *codes*, which guarantees a unique (unambiguous) factorization of any word to words from L [14]. Note that $||L||_{\text{sum}}$ can be exponentially larger than $||L||_{\max}$, and so the general question about polynomial bound in $||L||_{\max}$ for this subclass 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 language specified by an NFA, the problem was shown to be PSPACE-complete. In contrast, it is solvable in linear time when the language is specified by a DFA [17].

Both computational complexity question and finding the tight upper bound on the length also appear as one of the Berstel and Perrin's research problems [4, Resarch problems] and on the Shallit's list [20]. The problem itself has been connected with a number of different problems, e.g., testing if all bi-infinite words can be generated by the given list of words [17], the famous Černý conjecture [7], and the matrix mortality problem [14] in a restricted setting.

1.3 Contribution

We show that both Problem 1.1 and Problem 1.2 are PSPACE-complete, and we show exponential lower bounds on their related length problems. The complexity and bounds remain when the alphabet is binary. The solutions for both problems use similar constructions. Therefore, the ideas may be applicable to some other problems concerning the free monoid on a finite set of words.

The answer for the Frobenius monoid problem can be quite surprising because the problem is equally hard when L is represented by a popular more succinct representation, i.e., a DFA, a regular expression, or an NFA. Kao et al. [13] gave examples of finite languages L such that the longest words not present in the generated cofinite language L^* are of exponential length in the length of the longest words in L. However, the number of words in L is also exponential in these examples, thus they do not provide an exponential lower bound in terms of the size of the input L. Here, we additionally show stronger examples, where the longest words not present in cofinite L^* are of exponential length in the sum of the lengths of the words.

To make the reduction feasible, we construct it in several steps. We introduce a rewriting system called *set rewriting*, which is a basis for intermediate problems that we reduce from. In particular, we consider the immortality problem, which is whether there exists any configuration such that starting from it, we can apply rules infinitely long. This is in contrast with the usual settings where the initial configuration is given. It turns out that the existence of an arbitrary cycle is an essential property for Problem 1.1.

The solution for the factor universality problem uses similar construction to that of the previous problem with some technical differences. As a corollary, we exhibit a family of sets L of binary words whose minimal incompletable words are of exponential length in the length of the longest words in L or in the sum of the lengths of the words in L. This settles in the negative all weak variations of the Restivo's conjecture and essentially closes the problem.

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

Finally, we note that both problems can be solved in exponential time in the length of the longest word in L while polynomial in the sum of the lengths of words in L. This means that they can be effectively solved when the given set is dense, that is, the maximal length of words is much smaller than the sum of the lengths, e.g., the maximum length is logarithmic in the number of words.

Chapter 2

Set rewriting system

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

Definition 2.1. *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, 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 $S \cdot r_1 \cdot \cdots \cdot r_{k-1}$. The *resulting subset* from applying a legal sequence of rules is $S \cdot r_1 \cdot \cdots \cdot r_k$.

2.1 Immortality

In general, mortality is the problem of whether there exists any configuration such that there exist an infinite sequence of legally applied rules. In the case of systems with bounded configuration space, this is equivalent to the existence of a cycle in the configuration space. This is in contrast to the usual setting, where the initial configuration is given and we ask about reachability. For instance, mortality 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 the loop on the empty set. Therefore, we are interested only in non-trivial cycles, which do not contain the empty set, hence we add the additional restriction that the empty set is not reachable from any non-empty subset.

A set rewriting system is *non-emptiable* if for every element $p \in P$ and every rule $r \in R$, we have $r(p) \neq \emptyset$. It implies that for every non-empty subset S and a rule r, either $S \cdot r \neq \emptyset$ or r is illegal for S.

Problem 2.2 (Immortality of Set Rewriting)**.** *Given a non-emptiable set rewriting system (P, R), is there a non-empty subset $S \subseteq P$ and a non-empty sequence of rules $r_1, \ldots, r_k$ that is legal and yield S, i.e., $S \cdot r_1 \cdot \dots \cdot r_k = S$?*

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

Theorem 2.1. *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 $2^{|P|} - 2$. Furthermore, for every $n \geq 1$, there exist a set rewriting system (P, R) with $|P| = |R| = n$ and a non-empty subset of P that meets the bound.*

Proof. The upper bound follows since there are $2^{|P|} - 1$ distinct non-empty subsets and a legal sequence of $2^{|P|} - 2$ rules 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 $\mathit{val}(S, i) = 2^i$ if $b_i \in S$ and $\mathit{val}(S, i) = 0$ otherwise, and we set the *counter value* $\mathit{val}(S) = \sum_{0 \leq i \leq n-1} \mathit{val}(S, i)$. 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., $\mathit{val}(S) < \mathit{val}(S \cdot r_j)$. It is because we know that $\mathit{val}(S, i) = 0$ and

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

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 $\mathit{val}(S \cdot r_j) = \mathit{val}(S) + 1$. 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 above. The counter can be reset only if there exists a non-accepted word, which allows repeating a subset in the set rewriting system.

Theorem 2.2. *Problem 2.2 (Immortality of Set Rewriting) is PSPACE-complete.*

Proof. To solve the problem in PSPACE, it is enough to guess a subset S and a length k, and then guess at most k rules (without storing them), 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 $\mathcal{N} = (Q_{\mathcal{N}}, \Sigma_{\mathcal{N}}, \delta_{\mathcal{N}}, q_0, F_{\mathcal{N}})$, the question whether there is a word $w \in \Sigma_{\mathcal{N}}^*$ such that $\delta_{\mathcal{N}}(q_0, w) \cap F_{\mathcal{N}} = \emptyset$ is PSPACE-complete [1, Section 10.6].

Let $n = |Q_{\mathcal{N}}|$. We construct a set rewriting system (P, R). As an ingredient, we use the counter from the proof of Theorem 2.1. Let P be the disjoint union of $Q_{\mathcal{N}}$ 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 $\mathrm{val}(S, i) = 2^i$ if $b_i \in S$ and $\mathrm{val}(S, i) = 0$ otherwise, and we set $\mathrm{val}(S) = \sum_{0 \le i \le n-1} \mathrm{val}(S, i)$.

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

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

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

Assume that there is a word that is not accepted by $\mathcal{N}$. Note that if w is a shortest non-accepted word, then $q_0 \notin \delta(q_0, u)$ for all non-empty prefixes u of w. Hence, there exists a non-accepted word $w = a_1 a_2 \cdots a_m$ of length at most 2^{n-1}.

As observed in the proof of Theorem 2.1, 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 zero index if the least significant position; 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 $S = \{b_0, q_0\} \cdot r_{a_1, f(1)} \cdot r_{a_2, f(2)} \cdots r_{a_m, f(m)}$

has the property that $\mathrm{val}(S) = m < 2^n$ and $S \cap F = \emptyset$, because w is not accepted by $\mathcal{N}$. Thus, rule r_{reset} is legal, so $\{b_0, q_0\} \cdot r_{a_1,f(1)} \cdot r_{a_2,f(2)} \cdots r_{a_m,f(m)} \cdot r_{\mathrm{reset}} = \{b_0, q_0\}$. 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 $S \cdot r_{j_1} \cdot r_{j_2} \cdots , r_{j_m} = S$. As observed in the proof of Theorem 2.1, we know that every rule different from r_{reset} increments the counter by at least 1. Hence, there must be some index $1 \leq k \leq m$ such that $r_{j_k} = r_{\mathrm{reset}}$. 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 $r_{a_1,i_1}, r_{a_2,i_2}, \ldots, r_{a_d,i_d}$ without any r_{reset} rule. Thus we know that $\{q_0, b_0\} \cdot r_{a_1,i_1} \cdot r_{a_2,i_2} \cdots r_{a_d,i_d} r_{\mathrm{reset}} = \{q_0, b_0\}$. Since r_{reset} is legal when applied, the word $a_1 a_2 \cdots a_d$ is such that $\delta(q_0, a_1 a_2 \cdots a_d) \cap F = \emptyset$ thus is not accepted by $\mathcal{N}$. □

By the following observation, for immortality, 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 exists an arbitrary long legal sequence of rules.

Lemma 2.3. *If a rule r is legal for a subset $S \subseteq P$, then it is also legal for every subset $S' \subseteq S$ and $S' \cdot r \subseteq S \cdot r$.*

A similar property is essential for Problem 1.1, because if a word $wu \notin L^*$ for a word $w \in L^*$, then also suffix $u \notin L^*$.

2.2 Emptying

The second problem under our consideration is the reachability of the empty set, which is related to factor universality.

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

We call a set rewriting system *permissive* if there are no forbidden rules by $\bot$. In other words, all rules are legal for P. A permissive set rewriting system (P, R) is equivalent to a semi-NFA whose set of states is P and the alphabet is R; the initial and final states are irrelevant.

Problem 2.3 (Emptying Set Rewriting)**.** *For a given permissive set rewriting system (P, R), does there exist a P-emptying sequence of rules?*

Let $\mathcal{N} = (Q_{\mathcal{N}}, \Sigma, \delta_{\mathcal{N}}, q_0, F_{\mathcal{N}})$ be an NFA. For a subset $S \subseteq Q_{\mathcal{N}}$, a word $w \in \Sigma^*$ is called *S-emptying* if $\delta_{\mathcal{N}}(S, w) = \emptyset$. If every state in $\mathcal{N}$ is reachable from the initial state q_0 and from every state a final state can be reached, then the following criterion holds: the language of $\mathcal{N}$ is factor universal if and only if there does not exist a $Q_{\mathcal{N}}$-emptying word [17]. It is also known that the problem of whether a given language specified by an NFA is factor universal is PSPACE-complete. Since it is also easy to solve Problem 2.3 in PSPACE, it follows that it has the same complexity.

Theorem 2.4 ([17])**.** *Problem 2.3 (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, the counter is reset to the maximal value.

Theorem 2.5. *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 $2^{|P|} - 1$. Furthermore, 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 $n \geq 2$, 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 the value of the counter to decrease 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. $r_j(b_j) = \{b_i \mid i \in \{0, 1, \ldots, j-1\}\}$;
2. $r_j(b_i) = P$ for $i \in \{0, 1, \ldots, j-1\}$;
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 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 $S \cdot r_k = P$. Thus, the only rule that decreases the counter is r_i, and then $\mathrm{val}(S \cdot r_i) = \mathrm{val}(S) - 1$. Hence, the shortest sequence of rules that is P-emptying has length $2^n - 1$. □

Chapter 3

The Frobenius monoid problem

Before we go for PSPACE-hardness, we note the known result about PSPACE-membership.

Proposition 3.1 ([22])**.** *Problem 1.1 is in PSPACE.*

Proof. If L^* is cofinite, then the longest words not in L have at most exponential length [13]. Otherwise, the length of such words is unbounded. Thus, we can construct an NFA recognizing L^* and verify in NPSPACE whether there exists a longer word that is not accepted [22]. $\square$

For PSPACE-hardness, we reduce from Problem 2.2 (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 when 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.

3.1 The DFA construction

We get a non-emptiable set rewriting system (P, R). Without loss of generality, we assume the set of elements $P = \{p_1, p_2, \ldots, p_\ell\}$ and the rules $R = \{r_1, r_2, \ldots, r_m\}$.

We construct a DFA $\mathcal{A} = (Q_{\mathcal{A}}, \Sigma, \delta, q_0, F)$ such that L^* is not cofinite, where L is the language recognized by $\mathcal{A}$, if and only if there exists a non-empty subset $S \subseteq P$ and a non-empty sequence of rules $r_{i_1}, \ldots, r_{i_k}$ such that $S \cdot r_{i_1} \cdot \cdots \cdot r_{i_k} = S$. Our reduction will be polynomial in $|P| + |R|$. The number and the lengths of words in L will be also polynomial, which will allow further polynomial reduction to the case of a list of words.

The alphabet of $\mathcal{A}$ is $\Sigma = R \cup \{\alpha\}$. The letters from R are the *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 *set rewriting elements.*
- $Q_\mathrm{F} = \{f_x \mid x \in \{0, 1, \ldots, \ell\}\}$; the *forcing states.*
- $\{s_x^{i,j} \mid i \in \{1, 2, \ldots, \ell\} \wedge j \in \{1, 2, \ldots, m\} \wedge x \in \{\ell, \ell-1, \ldots, 1\} \wedge r_j(p_i) \neq \bot\}$; the *setting states.*
- $\{q_\mathrm{g}\}$; the *guard state.*
- $\{q_\mathrm{s}\}$; the sink state.

The transition function and the final states will be defined later, after explaining the overall idea of the construction.

We use a standard NFA construction recognizing the Kleene star of a language specified by a DFA. Let $\mathcal{A}^* = (Q_{\mathcal{A}^*}, \Sigma, \delta_{\mathcal{A}^*}, q_0, F_{\mathcal{A}^*})$ be the NFA obtained from $\mathcal{A}$ as follows. The set of states $Q_{\mathcal{A}^*}$ is $Q_\mathcal{A} \setminus \{q_\mathrm{s}\}$; we remove the sink state since it is represented by the empty subset of states in the NFA. We construct the extended transition function $\delta_{\mathcal{A}^*} \colon 2^{Q_{\mathcal{A}^*}} \times \Sigma^* \to 2^{Q_{\mathcal{A}^*}}$ from δ by adding ε-transitions from every final state to the initial state q_0 and removing transitions to the sink state. We assume that $\delta_{\mathcal{A}^*}$ is closed under ε-transitions, i.e., for $C \subseteq Q_{\mathcal{A}^*}$ and $w \in \Sigma^*$, $\delta_{\mathcal{A}^*}(C, w)$ is the set of all states reachable from a state in C through a path labeled by w interleaved with any number of ε-transitions, which also can be used at the beginning and at the end. We say that $\delta_{\mathcal{A}^*}(C, w)$ is the set of *active* states after applying w to C. The set of final states $F_{\mathcal{A}^*}$ is $F \cup \{q_0\}$; we can make q_0 final in our NFA construction, since the DFA is *non-returning*, i.e., there is no non-empty word w such that $\delta(q_0, w) = q_0$ in the DFA. It is well known that the constructed NFA recognizes the language L^* (see, e.g., [24]).

A word $w \in \Sigma^*$ is *irrevocably accepted* if for every $u \in \Sigma^*$, the word wu belongs to L^*.

A word w is *simulating for a subset* $S \subseteq Q_\mathrm{P}$ if it is of the form $r_{i_1}\alpha^\ell r_{i_2}\alpha^\ell \cdots r_{i_k}\alpha^\ell$ and the sequence of the rules $r_{i_1}, r_{i_2}, \ldots, r_{i_k}$ in w is legal for S.

A word $w \in \Sigma^*$ is f_0*-omitting for a subset* C 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 subset $\{q_0\}$.

Now, we explain the idea of the construction. We have the property that whenever the word does not follow the simulating pattern, it is not f_0-omitting. When this happens, some forcing state is always active and the word is irrevocably accepted, which means that all its extensions are in L^*. The forcing states are responsible for this property of f_0. On the other hand, words following the simulating pattern are f_0-omitting and not irrevocably accepted. Thus, if there are infinitely many such simulating words, which is equivalent to the immortality of the set rewriting system, then infinitely many words are outside the language.

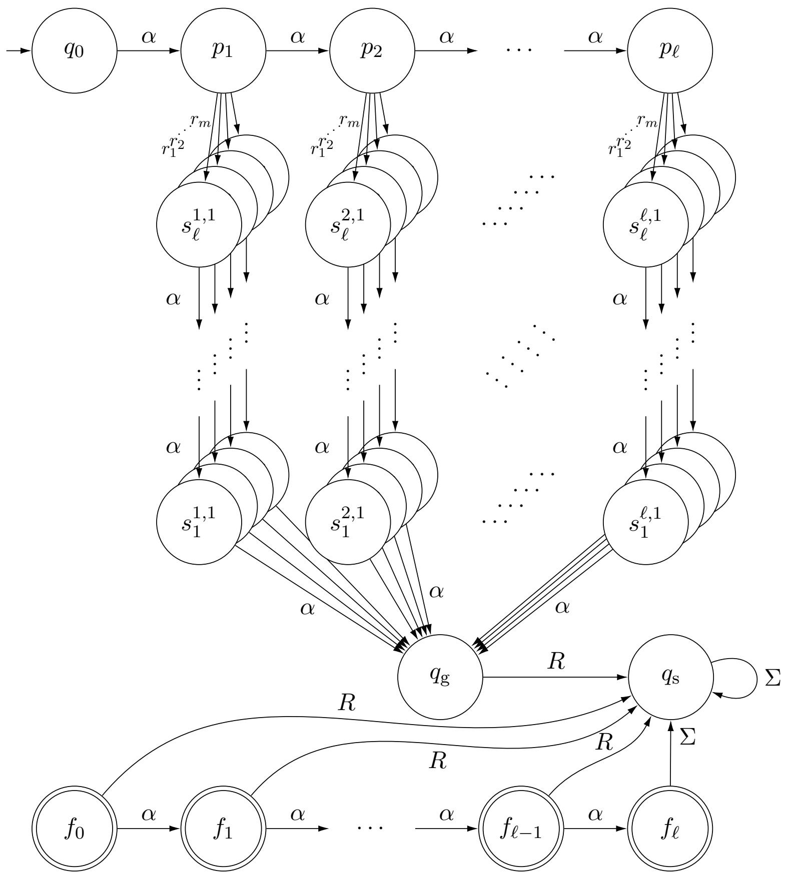

Figure 3.1: The scheme of the DFA $\mathcal{A}$ for a set rewriting system. All omitted transitions go to f_0.

The construction is presented in Fig. 3.1. States Q_{P}, together with the initial state q_0, form a chain on the transition of letter α, which is ended by f_0, i.e., $q_0 \xrightarrow{\alpha} p_1 \xrightarrow{\alpha} \cdots \xrightarrow{\alpha} p_\ell \xrightarrow{\alpha} f_0$. A subset of active states $S \subseteq Q_{\mathrm{P}}$ corresponds to the current subset of elements in our set rewriting system. By applying a rule letter r_j to S, which corresponds to applying the rule r_j in the set rewriting system, the states from Q_{P} are mapped into the setting states. If the rule is not legal for some element, that state in S is mapped directly to f_0 instead. The setting states form chains $s_\ell^{i,j}, \ldots, s_1^{i,j}$ on letter α, for every rule r_j and every element $p_i \in Q_{\mathrm{P}}$. Each such chain has its final states defined according to the action of the rule r_j for the element p_i. When $s_\ell^{i,j}$ becomes active, one must apply the word α^ℓ in order to avoid f_0. The setting states that are final in the chain activate q_0 at some point, which is then mapped to the right state of Q_{P} by the action of the remaining α letters. One cannot apply more than ℓ letters α in such a simulation step because of the guard state q_{g}, which is at the end of every setting chain. The guard state becomes active after α^ℓ applied for any non-empty $S \subseteq Q_{\mathrm{P}}$; it allows performing only the transitions of rule letters, which map the guard state to the empty subset (to the sink state in the DFA).

Therefore, if in the set rewriting system one has $S \subseteq Q_{\mathrm{P}}$ and applies a sequence of rules that results in $S' = S \cdot r_{i_1} \cdot \dots \cdot r_{i_k}$, then this corresponds to applying the word $r_{i_1}\alpha^\ell \dots r_{i_k}\alpha^\ell$, which is a simulating word for S.

A special case occurs at the beginning, when the subset of active states is $\{q_0\}$. Since no other states (in particular, the guard state) are active, we can use an arbitrary sequence α^i, for $1 \le i \le \ell$, before the first rule letter. This determines the first singleton subset from which we start applying rules.

The transition function δ is formally defined as follows:

- $\delta(q_0, \alpha) = p_1$.
- $\delta(p_i, \alpha) = p_{i+1}$ for all $i \in \{0, 1, \dots, \ell - 1\}$.
- $\delta(p_\ell, \alpha) = f_0$; this is required for the irrevocably accepting property of f_0.
- $\delta(p_i, r_j) = \begin{cases} s_\ell^{i,j}, & \text{if } r_j(p_i) \neq \bot \\ f_0, & \text{otherwise} \end{cases}$
  for all $i \in \{1, 2, \dots, \ell\}$ and $j \in \{1, 2, \dots, m\}$; when a rule is used, these transitions map a state from Q_P to the beginning of the corresponding setting chain or to f_0 if the rule is not legal when p_i is in the subset.
- $\delta(q_0, r_j) = f_0$ for all $j \in \{1, 2, \dots, m\}$; this forbids applying rule letters when q_0 is active.
- $\delta(s_x^{i,j}, \alpha) = s_{x-1}^{i,j}$ for all $i \in \{1, 2, \dots, \ell\}$, $j \in \{1, 2, \dots, m\}$, and $x \in \{\ell, \ell - 1, \dots, 2\}$; these are the setting chains on α.
- $\delta(s_1^{i,j}, \alpha) = q_\mathrm{g}$ for all $i \in \{1, 2, \dots, \ell\}$ and $j \in \{1, 2, \dots, m\}$; the setting chains end with the guard state.
- $\delta(s_x^{i,j}, r_y) = f_0$ for all $i, x \in \{1, 2, \dots, \ell\}$ and $j, y \in \{1, 2, \dots, m\}$; when the simulation pattern is not yet complete (less than ℓ letters α were applied, so there are some active states in the setting chains), this forbids using rule letters.
- $\delta(q_\mathrm{g}, \alpha) = f_0$; this forbids applying α when the guard state is active.
- $\delta(q_\mathrm{g}, r_j) = q_\mathrm{s}$; rule letters are allowed when the guard state is active and they deactivate it.
- $\delta(f_i, \alpha) = f_{i+1}$ for all $i \in \{0, 1, \dots, \ell - 1\}$; this chain of forcing states provides the property that whenever f_0 becomes active, the word is irrevocably accepted.
- $\delta(f_i, r_j) = q_\mathrm{s}$ for all $i \in \{0, 1, \dots, \ell\}$ and $j \in \{1, \dots, \ell\}$; rule letters clean the forcing states.
- $\delta(f_\ell, \alpha) = q_\mathrm{s}$; the chain of the forcing states ends with the sink state.

The set of final states F is the union of:

- $Q_{\rm F}$; all forcing states are final.
- $\{s_k^{i,j} \mid i,k \in \{1,2,\ldots,\ell\} \wedge j \in \{1,2,\ldots,m\} \wedge r_j(i) \neq \bot \wedge p_k \in r_j(i)\}$; states in a setting chain are final according to the rule of that chain.

Whenever a final state becomes active, q_0 becomes active through an ε-transition. Note that the indices in the setting chains are decreasing. This keeps the correspondence that if a state $s_k^{i,j}$ is final and p_i is in a subset C, then p_k is be active after applying $r_j\alpha^\ell$ to C.

Correctness. The correctness is observed through the following lemmas.

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

Lemma 3.2. *If a word $w \in \Sigma^*$ is not f_0-omitting, then it is irrevocably accepted.*

Proof. 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, $\delta_{\mathcal{A}^*}(\{f_0\}, v)$ 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 a forcing state. Then for every non-empty proper prefix v' of v, $\delta_{\mathcal{A}^*}(\{f_0\}, v')$ does not contain f_0, which would contradict that u is a shortest word. 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 $i \le \ell$. However, the transition of $\alpha^{\ell+1}$ through the chain on $Q_{\rm P}$ also maps q_0 to f_0, which yields a contradiction. □

The following lemma precises the meaning of that a simulating word corresponds to applying the sequence of rules that it contains.

Lemma 3.3. *Let $C \subseteq Q_{\rm P} \cup \{q_{\rm g}\}$, let $S = C \cap Q_{\rm P}$ be non-empty, and let $w = r_{i_1}\alpha^\ell \cdots r_{i_k}\alpha^\ell$ be a simulating word for S. Then $C' = (S \cdot r_{i_1} \cdot \cdots \cdot r_{i_k}) \cup \{q_{\rm g}\}$.*

Proof. Let C and S be as in the lemma, and let r_j be a rule. The transitions of r_j map each state $p_i \in S$ to $s_\ell^{i,j}$. Then the transitions of α^ℓ map these active states along the setting chains, maybe activating state q_0 when the setting state is final. Eventually, they are mapped to $q_{\rm g}$. A state $s_h^{i,j}$ is final if and only if $p_h \in r_j(p_i)$. From the construction, if $s_h^{i,j}$ is final, then q_0 becomes active after $\alpha^{\ell-h}$, which is then mapped to p_h by the transition of the remaining α^h. After the last α letter, the setting states are mapped to guard state $q_{\rm g}$ Hence, we have $C' = (S \cdot r_j) \cup \{q_{\rm g}\}$.

Since the set rewriting system is non-emptiable, the set $S' = S \cdot r_j$ is non-empty, and 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 $C \subseteq Q_{\mathcal{A}^*}$ must be a prefix simulating word for $C \cap Q_{\mathrm{P}}$. The required condition is that the guard state is also in C, so one cannot shift the states on Q_{P} by using α.

Lemma 3.4. *Let $S \subseteq Q_{\mathrm{P}}$ be non-empty, and let $C = S \cup \{q_{\mathrm{g}}\}$. 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, 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, some of the first setting states must be active, because $S \neq \emptyset$. Hence α^ℓ must be used, unless w ends. By Lemma 3.3 for C and $r_j\alpha^\ell$, we know that the set of active states is $C' = (S \cdot r_j) \cup \{q_{\mathrm{g}}\}$. By iterating this argument, we observe that between each rule letter there must be exactly ℓ letters α, and at the end, there are at most ℓ letters α. Furthermore, each of the rules applied must be legal. Therefore, we know that word w has to be a prefix of some simulating word for S. □

In the beginning, before we can 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 3.5. *If a word w is f_0-omitting, then w is a prefix of $\alpha^i w'$ for $1 \leq i \leq \ell$ and some w' that is a simulating word for $\{p_i\}$.*

Proof. Let w be a f_0-omitting word. Since we start from $\{q_0\}$, we know that w must start with α^i for some $1 \leq i \leq \ell$, unless it is empty. Then, unless w ends, there is some rule letter r_j, which must be legal for $\{p_i\}$, followed by α^ℓ.

Hence $w = \alpha^i r_j \alpha^\ell w''$ for some suffix w'' of w. By Lemma 3.3, 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 3.4 applied to C, since f_0 cannot be activated, we know that w'' must be a prefix of a simulating word for S. We let $w' = r_j \alpha^\ell w''$, which 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 3.6. *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 2.3, there exists a singleton $\{p_i\}$ and a sequence of k legally applied rules $r_{i_1}, \ldots, r_{i_k}$ to $\{p_i\}$. Hence, $w = \alpha^i r_{i_1} \alpha^\ell \cdots r_{i_k} \alpha^\ell$ is a simulating word for $S = \{p_i\}$. By Lemma 3.3, we know that $\delta_{\mathcal{A}^*}(\{q_0\}, w) \subseteq Q_{\mathrm{P}} \cup \{q_{\mathrm{g}}\}$, which does not contain any final states, thus w is not accepted.

Conversely, assume that L^* is not cofinite. Thus there are infinitely many words that are not accepted, which, in particular, by Lemma 3.2, are f_0-omitting.

Let w be a f_0-omitting word of length at least $\ell + (\ell+1)2^{|Q_{\mathrm{P}}|}$. By Lemma 3.5, we know that w has the form of $\alpha^i w'$, where $i \leq \ell$ and w' is a prefix of a simulating word for $\{p_i\}$.

This simulating word must have length at least $(\ell+1)2^{|Q_{\mathrm{P}}|}$, hence it contains a sequence of $k \geq 2^{|Q_{\mathrm{P}}|}$ rule letters. We conclude that this sequence $r_{i_1} \cdot \dots \cdot r_{i_k}$ is legal for $\{p_i\}$, and it does not lead to the empty set as it is unreachable from a non-empty subset. If we look at the sequence of sets $S_j = \{p_i\} \cdot r_{i_1} \cdot \dots \cdot r_{i_j}$, for $j \in \{0, \dots, 2^{|Q_{\mathrm{P}}|}\}$, then there must be some distinct indices x and y such that $x < y$ and $S_x = S_y$. Hence, the rewriting system is immortal because of S_x and the sequence $r_{i_{x+1}}, r_{i_{x+2}}, \dots, r_{i_y}$. □

We conclude this part with

Theorem 3.7. *Problem 1.1 is PSPACE-hard if L is specified by a DFA over a given (growing) alphabet.*

3.2 Binarization

To show that the PSPACE-hardness remains when the alphabet is restricted to binary, we apply a variation of a standard binarization of a language.

We modify the construction of $\mathcal{A}$ from to obtain a binary $\mathcal{B} = (Q_{\mathcal{B}}, \{0,1\}, \delta_{\mathcal{B}}, q_0, F)$, where $Q_{\mathcal{B}}$ is $Q_{\mathcal{A}}$ with some states added, and q_0 and F are from the original $\mathcal{A}$.

The letter α is encoded by 0, every letter r_i is encoded by 1^i0 for $i \leq m-1$, and r_m is encoded by 1^m. Note that this binary encoding is a complete prefix code, thus the encoding of a word $w \in \Sigma^*$ is unambiguous and every binary word w', after removing at most $m-1$ symbols from the end, encodes some word w.

The construction of $\mathcal{B}$ is as follows. The transitions labeled by α are now labeled by 0. We introduce $m-1$ new states for each state of Q_{P} in the way that a word encoding r_i acts as r_i in the original automaton; these new states are not final. The transitions of R on $Q_{\mathcal{B}} \setminus Q_{\mathrm{P}}$, which are the same for every $r \in R$, are simply replaced with one transition labeled by 1.

The correctness of the binarization is observed through the following lemmas.

Lemma 3.8. *If a word w is f_0-omitting for a subset $C \subseteq Q_{\mathcal{A}^*} \setminus Q_{\mathrm{F}}$, then its binary encoding w' is f_0-omitting for C and such that $\delta_{\mathcal{A}^*}(C, w) = \delta_{\mathcal{B}^*}(C, w')$.*

Proof. This can be observed by analyzing the transitions from each state in $Q_{\mathcal{A}^*} \setminus Q_{\mathrm{F}}$ in both automata. □

Lemma 3.9. *If a word w is not f_0-omitting for a subset $C \subseteq Q_{\mathcal{A}^*} \setminus Q_{\mathrm{F}}$ for $\mathcal{A}^*$, then its binary encoding w' is not f_0-omitting for C in $\mathcal{B}^*$.*

Proof. Suppose that a prefix of w activates f_0; let ua be a shortest such prefix for $u \in \Sigma^*$ and $a \in \Sigma$. From (1), we know that $\delta_{\mathcal{A}^*}(C, u) = \delta_{\mathcal{B}^*}(C, u')$, where u' is the binary encoding of u. If $a = \alpha$, then $u'0$ activates f_0 in $\mathcal{B}^*$. If $a \in R$, then active q_0, an active state $s_k^{i,j}$, or an active state p_i is mapped to f_0 by the transition of a. In the first two cases, $u'1$ activates f_0, and in the third case, $u'a'$ activates f_0, where a' is the binary encoding of a. □

Lemma 3.10. *The language of $\mathcal{B}^*$ is cofinite if and only the language of $\mathcal{A}^*$ is cofinite.*

Proof. From Lemma 3.8 and by the fact that all not f_0-omitting words for $\{q_0\}$ are accepted, we know that if a word $w \in \Sigma^*$ is not accepted by $\mathcal{A}^*$, then its binary encoding $w' \in \{0,1\}^*$ is not accepted by $\mathcal{B}^*$. Thus, we get that 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 a $t \geq m$, let w' be a binary word not accepted by $\mathcal{B}^*$ and of length at least t. Let u' be the maximal prefix of w' that properly encodes a word $u \in \Sigma^*$; then u' is shorter by at most $m - 1$ than w'. We observe that Lemma 3.2 holds for $\mathcal{B}^*$. Hence, since w' is not accepted, u' must be f_0-omitting. From Lemma 3.9, we know that u also must be f_0-omitting. By applying the same argument as in the proof of Lemma 3.6 to u for $t \geq m((\ell+1)2^{|Q_{\mathrm{P}}|} + \ell)$, (this ensures that u is of length at least $\ell + 1 + (\ell+1)2^{|Q_P|}$, since the any original letter is encoded by at most m letters) we conclude that the set rewriting system is immortal, thus the language of $\mathcal{A}^*$ is not cofinite. □

3.3 List of words

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

Lemma 3.11. *The maximum length of words in the language of $\mathcal{B}$ is equal to $3\ell + m + 1$ and the number of words is at most $m\ell^2 + (1 + \ell m(1+\ell) + 1)(1+\ell)$.*

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: $q_0 \xrightarrow{0^\ell} p_\ell \xrightarrow{1^m} s_\ell^{\ell,m} \xrightarrow{0^{\ell-1}} s_1^{\ell,m} \xrightarrow{0} q_g \xrightarrow{1} f_0 \xrightarrow{0^\ell} f_\ell$.

For the number of words in the recognized language, we consider all final states. The first type of final states is the setting states. Each such state is reachable from q_0 by a unique path, thus each of them induces one word in the language, which gives at most $m\ell^2$ words. The second type is forcing states. A state f_i may be reached through different paths, but all such paths consist of a path to f_0, whose number is bounded by the number of states, and a unique path from f_0 to f_i. In this case, we have at most $(1 + \ell m(1 + \ell) + 1)(1 + \ell)$ words. $\square$

We conclude with

Theorem 3.12. *Problem 1.1 is PSPACE-hard if L is a finite list of binary words.*

Using the construction, we can also infer the hardness for every fixed size larger than one of the alphabet. For this, it is enough to add a suitable number of additional letters to $\mathcal{B}$ with the action mapping $Q_{\mathcal{B}} \setminus (F \cup \{q_{\mathrm{s}}\})$ to f_0 and mapping $F \cup \{q_{\mathrm{s}}\}$ to q_{s}.

Chapter 4

The factor universality problem

We follow similarly as in Section 3. In a few steps, we reduce from Problem 2.3 (Emptying Set Rewriting) to Problem 1.2 (Factor Universality for a Finite Set of Words) when L is a given as a finite list of binary words.

4.1 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. To do this, we slightly modify the DFA construction $\mathcal{A}$ from Subsection 3.1 as follows. We remove the last state f_ℓ and end the chain of the forcing states with $f_{\ell-1}$. Thus, the set Q_F becomes $\{f_x \mid x \in \{0, 1, \ldots, \ell-1\}\}$, and we redefine the transition $\delta(f_{\ell-1}, \alpha) = q_\mathrm{s}$. As before, we build the standard NFA $\mathcal{A}^*$ recognizing the language L^*, where L is the language of $\mathcal{A}$.

The idea of the modified construction is as follows. In the NFA $\mathcal{A}^*$, all states are reachable from the initial state q_0. Since we also remove the sink state q_s, the NFA meets the mentioned criterion for factor universality (Subsection 2.2). 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 rule sequences in the set rewriting system in the same way as in Subsection 3.1. The construction ensures that to map the whole set Q_P to the empty set, there must exist a P-emptying sequence of rules in the set rewriting system. The forcing states have the property that whenever f_0 is activated, the only way to get rid of all forcing states is to make the whole Q_P active again. When f_0 is active, which is also the case at the beginning, this is done by applying the word α^ℓ.

Correctness. The correctness is observed through the following lemmas.

Lemma 4.1. *We have:*

1. $\delta_{\mathcal{A}^*}(Q_{\mathcal{A}^*}, r_1^2) = \{f_0, q_0\}$, *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 4.2. *Let $C \subseteq Q_{\mathcal{A}^*}$, let $f_0 \in C$, 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 with the property. Hence, there is no non-empty prefix u of w such that $f_0 \in \delta_{\mathcal{A}^*}(\{f_0\}, u)$. Consider a prefix α^i of w for an $i < \ell$. Then $\delta_{\mathcal{A}^*}(\{f_0\}, \alpha^i) = \{f_i, q_0, p_1, \ldots, p_i\}$. Thus w must have length at least ℓ. If w would start with $\alpha^i r_j$ for an $i < \ell$ and some rule letter r_j, then active state q_0 would be mapped to f_0 by the transition of r_j. Thus, w must start with the prefix $u = \alpha^\ell$, which is that $\delta_{\mathcal{A}^*}(\{f_0\}, u) = Q_\mathrm{P}$. □

We show the properties of a simulating word.

Lemma 4.3. *Let $C \subseteq Q_\mathrm{P} \cup \{q_\mathrm{g}\}$, let $S = C \cap Q_\mathrm{P}$ be non-empty, and let $w = r_{i_1}\alpha^\ell \cdots r_{i_k}\alpha^\ell$ be a simulating word for S. Then:*

$$C' = \begin{cases} (S \cdot r_{i_1} \cdot \cdots \cdot r_{i_k}) \cup \{q_\mathrm{g}\}, & \text{if } S \cdot r_{i_1} \cdot \cdots \cdot r_{i_{k-1}} \neq \emptyset \\ \emptyset = (S \cdot r_{i_1} \cdot \cdots \cdot r_{i_k}), & \text{otherwise.} \end{cases}$$

Proof. In the case of $S \cdot r_{i_1} \cdot \cdots \cdot r_{i_{k-1}} \neq \emptyset$, the proof is the same as that of Lemma 3.3, since for all $0 \leq j \leq k-1$, we have $S \cdot r_{i_1} \cdot \cdots \cdot r_j \neq \emptyset$, thus all preconditions apply.

Otherwise, let $j < k$ be the smallest index such that the set $S \cdot r_{i_1} \cdot \cdots \cdot r_{i_j}$ is empty. By the argument for the first case, we know that $\delta_{\mathcal{A}^*}(S, r_{i_1}\alpha^\ell \cdots r_{i_j}\alpha^\ell) = \{q_\mathrm{g}\}$. 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 are related with simulating words.

Lemma 4.4. *Let $S \subseteq Q_\mathrm{P}$ be non-empty, and let $C = S \cup \{q_\mathrm{g}\}$. If 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 3.4, we observe that a word w must start with $r_j\alpha^\ell$, unless it ends prematurely. Then, by Lemma 4.3, we have $C' = \delta_{\mathcal{A}^*}(C, r_j\alpha^\ell) = (S \cdot r_j) \cup \{q_\mathrm{g}\}$. We apply this argument iteratively, until either w ends, in which case (1) holds, or C' becomes $\{q_\mathrm{g}\}$, in which case (2) holds. □

Lemma 4.5. *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 $\delta_{\mathcal{A}^*}(Q_{\mathrm{P}}, u) = Q_{\mathrm{P}}$; otherwise, we can search for a factor v in w with u removed. Hence, by Lemma 4.2, w must be f_0-omitting. By Lemma 4.4, 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 not a simulating word, write $w = vr_{i_{k+1}}\alpha^i$ for a simulating word $v = r_{i_1}\alpha^\ell \cdots r_{i_k}\alpha^\ell$ for Q_{P} and some $0 \le i < \ell$; otherwise let $v = w$. Let $C' = \delta(Q_{\mathrm{P}}, v)$. By Lemma 4.3, $C' \subseteq Q_{\mathrm{P}} \cup \{q_{\mathrm{g}}\}$ and $S' = C' \cap Q_{\mathrm{P}} = P \cdot r_{i_1} \cdot \cdots \cdot r_{i_k}$. If $S' \neq \emptyset$, then the transitions 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 show the equivalence between the reduced problems.

Lemma 4.6. *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 for Q_{P} word for $\mathcal{A}^*$.*
3. *There exists a $Q_{\mathcal{A}^*}$-emptying word for $\mathcal{A}^*$.*

Proof. (1) $\Rightarrow$ (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 $w = r_1\alpha^\ell \cdots r_k\alpha^\ell$, which is a simulating word for Q_{P}. By Lemma 4.3, we conclude that $\delta_{\mathcal{A}^*}(Q_{\mathrm{P}}, w) \subseteq \{q_{\mathrm{g}}\}$. Thus, wr_1 is Q_{P}-emptying and f_0-omitting for Q_{P}.

(2) $\Rightarrow$ (3): If w is a Q_{P}-emptying word, then, by Lemma 4.1, $\delta_{\mathcal{A}^*}(Q_{\mathcal{A}^*}, r_1^2\alpha^\ell w) = \emptyset$.

(3) $\Rightarrow$ (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 4.5, 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 4.7. *Problem 1.2 is PSPACE-hard if L is specified by a DFA over a given (growing) alphabet.*

4.2 Binarization and list of words

We reduce to a binary DFA $\mathcal{B}$ using the same construction as in Subsection 3.2.

We observe that Lemma 3.8 and Lemma 3.9 hold also in this case. It is because both constructions differ only on the set Q_{F}, whose transitions are irrelevant for the observations.

Lemma 4.8. *For $\mathcal{B}^*$, there is a Q_{P}-emptying and f_0-omitting for Q_{P} word if and only if there is 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 for Q_{P}.*

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 let $w = uv$. Observe that Lemma 4.2 holds for $\mathcal{B}^*$; for this, it is enough to change in its proof α to 0 and r_j to 1. By this lemma, v has to be f_0-omitting for $Q_{\mathcal{P}}$, as otherwise u could be longer. Hence, v is $Q_{\mathcal{P}}$-emptying and f_0-omitting for $Q_{\mathcal{P}}$. □

Lemma 4.9. *There is a Q_{P}-emptying and f_0-omitting for Q_{P} word for $\mathcal{A}^*$ if and only if there is such a word for $\mathcal{B}^*$. In particular, if w' is such a word for $\mathcal{B}^*$, then $w'0$ encodes a word with this property for $\mathcal{A}^*$.*

Proof. Let w be a Q_{P}-emptying and f_0-omitting for Q_{P} word for $\mathcal{A}^*$. From Lemma 3.8, we know that its binary encoding w' is f_0-omitting for Q_{P} and such that $\delta_{\mathcal{B}^*}(Q_{\mathrm{P}}, w') = \delta_{\mathcal{A}^*}(Q_{\mathrm{P}}, w) = \emptyset$.

Conversely, assume that there is a Q_{P}-emptying and f_0-omitting for Q_{P} binary word w' for $\mathcal{B}^*$. We know that $w'0$ has the same properties, and it must be an encoding of some word $w \in \Sigma_{\mathcal{A}}^*$. Then, from Lemma 3.9, w must be also f_0-omitting for Q_{P}. From Lemma 3.8, we conclude that w has to be also Q_{P}-emptying. □

4.3 List of words

Lemma 4.10. *The maximum length of words in the language of $\mathcal{B}$ is equal to $3\ell+m$ and the number of words is at most $m\ell^2 + (1 + \ell m(1 + \ell) + 1)\ell$.*

Proof. We count words as in the proof of Lemma 3.11, taking into account that the chain of forcing states is shorter by 1. □

We conclude with

Theorem 4.11. *Problem 1.2 is PSPACE-hard when the alphabet is binary.*

As for the previous problem, in the same way, by adding a suitable number of letters, it is possible to show the hardness for every fixed size larger than one of the alphabet.

Chapter 5

Lower bounds

By $||L||_{\max}$ we denote the length of the longest words in L and by $||L||_{\text{sum}}$ we denote the sum of the lengths of the words in L. Thus, $||L||_{\text{sum}}$ can be treated as the size of the input L.

5.1 The longest omitted words

It is know that for each odd integer $n \geq 5$, there exists a set of binary words L of length at most n such that L^* is cofinite and the longest words not in L^* are of length $\Omega(n^2 2^{\frac{n}{2}})$ [13]. However, the constructed L contains exponentially many words, thus an exponential lower bound in terms of the size of L could not be inferred.

We show an exponential in $||L||_{\text{sum}}$ lower bound 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 2.1).

Theorem 5.1. *There exists an infinite family whose elements L are finite sets of binary words and are such that L^* is cofinite and the longest words not in L^* are of length at least $2^{\frac{||L||_{\max}-1}{4}} \cdot \frac{||L||_{\max}-1}{4}$ and this length is $2^{\Omega(\sqrt[5]{||L||_{\text{sum}}})}$.*

Proof. For an $n \geq 2$, we take the set rewriting system (P, R) and a subset S from Theorem 2.1 meeting the bound $2^n - 2$, and we use the construction from Section 3 to create a list of binary words L. Since the set rewriting system is mortal, L^* is cofinite.

The length of the longest words in this list is equal to $||L||_{\max} = 4n + 1$ and there are at most $n^3 + (1 + n^2(1 + n) + 1)(1 + n) = n^4 + 3n^3 + n^2 + 2n + 2$ words (Lemma 3.11), thus $||L||_{\text{sum}} \leq (n^4 + 3n^3 + n^2 + 2n + 2)(4n + 1)$.

We take a binary simulating word w' for the longest possible legal sequence of rules in this set rewriting system for some singleton S. From Lemma 3.3 and

Lemma 3.8, we know that $0^i w' \notin L^*$, for some $i \geq 1$. The number i corresponds to the initial singleton set S in the construction. For $n \geq 2$, we can lower bound the length of the 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 (the encoding of the rule letter and 0^n) 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 $n = \frac{||L||_{\max}-1}{4}$ and $n = \Omega(\sqrt[5]{||L||_{\text{sum}}})$, the length of the word $0^i w'$ is at least $2^{\frac{||L||_{\max}-1}{4}} \cdot \frac{||L||_{\max}-1}{4}$ when written in terms of $||L||_{\max}$, and it is $2^{\Omega(\sqrt[5]{||L||_{\text{sum}}})}$ in terms of $||L||_{\text{sum}}$. □

5.2 The shortest incompletable words

We show that when L^* is not factor universal, the length of the shortest words that are not completable can be exponential in either $||L||_{\max}$ or $||L||_{\text{sum}}$.

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 2.5).

Theorem 5.2. *There exists an infinite family whose elements L are finite sets of binary words such that the shortest incompletable binary words are of length at least $2^{\frac{||L||_{\max}}{4}} \cdot \frac{||L||_{\max}}{4}$ and this length is $2^{\Omega(\sqrt[5]{||L||_{\text{sum}}})}$.*

Proof. For $n \geq 2$, we take the set rewriting system (P, R) from Theorem 2.5. Then we apply the construction from Subsection 4 to create a list of binary words L. Since there exists a P-emptying sequence of rules, by Lemmas 4.6, 4.9, and 4.8, we conclude that there is a $Q_{\mathcal{B}^*}$-emptying word in $\mathcal{B}^*$, thus L^* is not factor universal.

We show a lower bound on the length of such words. If some word is not a factor of any word from L^*, then this word must be $Q_{\mathcal{B}^*}$-emptying. From Lemma 4.8, we know that it contains a factor w' that is Q_{P}-emptying and f_0-omitting for Q_{P}. Then, from Lemma 4.9, we know that the word w encoded by binary word $w'0$ is Q_{P}-emptying for $\mathcal{A}^*$. By Lemma 4.5, w contains as a factor a simulating word v whose sequence of rules is P-emptying. Since the shortest such sequence of rules has length $2^n - 1$, word v and also w have length at least $(2^n - 1) \cdot (n+2)$. Moreover, they contain at least $(2^n - 1)$ rule letters. Since, for $n \geq 2$, each rule letter is encoded by at least two binary symbols, we conclude that w', where $w'0$ is the encoding of w, has length at least $(2^n - 1) \cdot (n+2) - 1$. For $n \geq 2$, $(2^n - 1) \cdot (n+2) - 1 \geq 2^n \cdot n$.

By setting $n = \frac{||L||_{\max}}{4}$ and $n = \Omega(\sqrt[5]{||L||_{\text{sum}}})$, the length of every Q_{P}-emptying word is at least $2^{\frac{||L||_{\max}}{4}} \cdot \frac{||L||_{\max}}{4}$ when written in terms of $||L||_{\max}$, and it is $2^{\Omega(\sqrt[5]{||L||_{\text{sum}}})}$ in terms of $||L||_{\text{sum}}$. □

Chapter 6

Upper bounds

We show algorithms and upper bounds on the related length for both problems, which are exponential only in $||L||_{\max}$ while remains polynomial in $||L||_{\text{sum}}$.

For the Frobenius monoid problem, there was shown upper bound $\frac{2}{2|\Sigma|-1}(2^{||L||_{\max}}|\Sigma|^{||L||_{\max}}-1)$ on the length of the longest words not in L^* when L^* is cofinite [13]. We show an upper bound that involves both $||L||_{\max}$ and $||L||_{\text{sum}}$.

Theorem 6.1. *Problem 1.1 can be solved in time exponential only in $||L||_{\max}$ while polynomial in $||L||_{\text{sum}}$. If L^* is cofinite, then the longest words not in L^* have length at most $1 + (||L||_{\text{sum}} + 1)2^{||L||_{\max}}$.*

Proof. We construct a DFA $\mathcal{A}$ recognizing L in the way that it forms a radix trie. Then every distinct word w maps the initial state q_0 to a different state, unless it is the unique non-final sink state q_{s}. By a standard construction for the Kleene star, we construct an NFA $\mathcal{A}^* = (Q_{\mathcal{A}^*}, \Sigma, \delta_{\mathcal{A}^*}, q_0, F_{\mathcal{A}^*})$ recognizing L^*. We can assume that L does not contain the empty word, so $\mathcal{A}^*$ contains an ε-transition from every final state to the initial state q_0. The final states $F_{\mathcal{A}^*}$ is the set of final states of $\mathcal{A}$ with q_0 added. We can remove the sink state from $\mathcal{A}^*$, hence from every state, a final state is reachable in $\mathcal{A}^*$.

We observe that in $\mathcal{A}^*$, after reading any word w, there are no more than $|Q_{\mathcal{A}^*}| \cdot 2^{||L||_{\max}}+1$ active states. We define the *level* of a state $q \in Q_{\mathcal{A}^*} \setminus \{q_{\text{s}}\}$ to be the length of the (unique) shortest word mapping q_0 to q. Every state by the action of every letter is mapped to at most one state, which has the level larger by 1, and possibly to q_0 by following ε-transition. Hence, for a subset with at most one state at each level, the action of every letter preserves this property. Since the initial subset is $\{q_0\}$, after reading any word, for every level at most one state can be active. Moreover, if q is the active state with the largest level i, the set of possible active states with smaller levels is determined, because if w is the unique shortest word of length i such that $\{q\} \subseteq \delta(q_0, w)$, then the only possible active state at a level $j < i$ is that in $\delta(q_0, w')$ (if it contains a state of level j), where w' is the suffix of w of length j. The largest possible level is ℓ. State q_0 is active if and only if a final state other than q_0 is active, with the exception of the initial active subset $\{q_0\}$. Hence, we can choose one of the $|Q_{\mathcal{A}^*}|$ states to be that with the largest level, and then any subset of the ℓ states that are determined by the chosen state.

Having the number of reachable active subsets of states bounded, we can determinize $\mathcal{A}^*$ to a minimal DFA $\mathcal{D}_{\mathcal{A}^*}$ with at most $|Q_{\mathcal{A}^*}| \cdot 2^{||L||_{\max}} + 1$ states. Finally, the problem of whether a minimal DFA recognizes a cofinite language is equivalent to whether there exists a cycle containing a non-final state.

Since $|Q_{\mathcal{A}^*}| \leq ||L||_{\text{sum}} + 1$, the upper bound on the length follows. □

For the factor universality problem, only trivial upper bound $2^{||L||_{\text{sum}} - ||L||_{\max} + 1}$ was known [10].

Theorem 6.2. *Problem 1.2 can be solved in time exponential only in $||L||_{\max}$ while polynomial in $||L||_{\text{sum}}$. If the set is not complete, then the shortest incompletable words have length at most $||L||_{\max} + 1 + (||L||_{\text{sum}} + 1)2^{||L||_{\max}}$.*

Proof. We construct an NFA $\mathcal{A}^*$ for L^* as in the proof of Theorem 6.1. We remove its sink state and make all states initial and final, hence it recognizes the language of all factors of L^*. The language is universal if and only if there exists a word w such that $\delta(Q_{\mathcal{A}^*}, w) = \emptyset$ [17].

Similarly as before, we observe that in $\mathcal{A}^*$, after reading any word w of length at least $||L||_{\max}$, there are no more than $|Q_{\mathcal{A}^*}| \cdot 2^{||L||_{\max}} + 1$ active states. Since we start with the set of all states $Q_{\mathcal{A}^*}$, at the beginning there could be more reachable subsets.

If there exists a word w such that $\delta_{\mathcal{A}^*}(Q^*_{\mathcal{A}}, w) = \emptyset$, then for every word u we also have $\delta_{\mathcal{A}^*}(Q_{\mathcal{A}^*}, uw) = \emptyset$. Hence, we can start from an arbitrary word u of length $||L||_{\max}$, and then check the reachability of $\emptyset$ visiting at most $|Q_{\mathcal{A}^*}| \cdot 2^{||L||_{\max}} + 1$ states. □

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

Bibliography

[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, 2009.

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

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

[8] 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.

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

[10] 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.

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

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

[13] 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.

[14] S. Kiefer and C. Mascle. On Finite Monoids over Nonnegative Integer Matrices and Short Killing Words. In *STACS*, LIPIcs, 2019.

[15] 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.

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

[17] 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.

[18] 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.

[19] J. Sandrine, 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.

[20] 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`.

[21] S. H. Smita. A Review of Concatenative text To Speech Synthesis. *International Journal of Latest Engineering Research*, 3, 2014.

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

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

[24] 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 5.1 of sets of words L for which the shortest incompletable words in L^* are of exponential length $2^{\frac{||L||_{\max}}{4}} \cdot \frac{||L||_{\max}}{4}$ in terms of $||L||_{\max}$ and $2^{\Omega(\sqrt[5]{||L||_{\text{sum}}})}$ in terms of $||L||_{\text{sum}}$.

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 $x \in \{0, \ldots, n-1\}$; $(q_0 \xrightarrow{0^i} p_i \xrightarrow{1^j0} s_n^{i,j} \xrightarrow{0^n} q_g \xrightarrow{0} f_0 \xrightarrow{0^x} f_x)$
- $0^i1^n0^n00^x$ for $i \in \{1, \ldots, n\}$ and $x \in \{0, \ldots, n-1\}$; $(q_0 \xrightarrow{0^i} p_i \xrightarrow{1^n} s_n^{i,n} \xrightarrow{0^n} q_g \xrightarrow{0} f_0 \xrightarrow{0^x} f_x)$

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 [15] as a source file.