Discrete Mathematics and Theoretical Computer Science DMTCS vol. **17:1**, 2015, 283–306

Parameterized Complexity of Synchronization and Road Coloring

Vojtěch Vorel[1]* and Adam Roman[2]†

[1]*Faculty of Mathematics and Physics, Charles University, Prague, Czech Republic*
[2]*Institute of Computer Science, Jagiellonian University, Krakow, Poland*

received 9th July 2014, revised 14th Oct. 1998, accepted 8th Apr. 2015.

First, we close the multi-parameter analysis of a canonical problem concerning short reset words (SYN) initiated by Fernau et al. (2013). Namely, we prove that the problem, parameterized by the number of states, does not admit a polynomial kernel unless the polynomial hierarchy collapses. Second, we consider a related canonical problem concerning synchronizing road colorings (SRCP). Here we give a similar complete multi-parameter analysis. Namely, we show that the problem, parameterized by the number of states, admits a polynomial kernel and we close the previous research of restrictions to particular values of both the alphabet size and the maximum length of a reset word.

Keywords: synchronizing word, reset word, Road Coloring Problem, synchronizing automata, parameterized complexity, Černý conjecture

1 Introduction

Questions about synchronization of finite automata have been studied since the early times of automata theory. The basic concept is very natural: For a given machine, we want to find an input sequence that gets the machine to some particular state, no matter in which state the machine was before. Such sequence is called a *reset word*[(i)]. If an automaton has a reset word, we call it a *synchronizing automaton*. A need for finding reset words appears in several fields of mathematics and engineering. Classical applications (see [21]) include model-based testing of sequential circuits, robotic manipulation, and symbolic dynamics, but there are also some important connections with information theory [20] and with formal models of biomolecular processes [3].

Two particular problems concerning synchronization have gained some publicity: the Road Coloring Problem and the Černý conjecture. The first one has been solved by Trahtman [17, 19] in 2008 by

*Corresponding author. Email: `vorel@ktiml.mff.cuni.cz`. Supported by the Czech Science Foundation grant GA14-10799S and the GAUK grant No. 52215.

†Email: `roman@ii.uj.edu.pl`. Supported by the Polish Ministry of Science and Higher Education Iuventus Plus grant IP2012 052272.

(i) Some authors use other terms such as *synchronizing word* or *directing word*.

Parameter	Parameterized Complexity of SYN		Polynomial Kernel of SYN	
k	W[2]-hard	[8]	—	
$\lvert I \rvert$	NP-complete for $\lvert I \rvert = 2, 3, \ldots$	[7]	—	
k and $\lvert I \rvert$	FPT, running time $\mathcal{O}^{\star}(\lvert I \rvert^{k})$	[triv.]	Not unless $\mathrm{NP} \subseteq \mathrm{coNP/poly}$	[8]
t	FPT, running time $\mathcal{O}^{\star}(2^{t})$	[triv.]	Not unless PH collapses	♦

Parameter	Parameterized Complexity of SRCP		Polynomial Kernel of SRCP	
k	NP-complete for $k = 4, 5, \ldots$	[13]	—	
$\lvert I \rvert$	NP-complete for $\lvert I \rvert = 2, 3, \ldots$	♦	—	
k and $\lvert I \rvert$	*See* Tab. *2*		—	
t	FPT, running time $\mathcal{O}^{\star}(2^{\lvert I \rvert})$	♦	Yes	♦

Tab. 1: Results of the complete multi-parameter analysis of SYN and SRCP. Diamonds mark the results of the present paper

proving that the edges of any strongly connected aperiodic directed multigraph with constant out-degree can be colored such that a synchronizing automaton arises. Motivation for this problem comes from symbolic dynamics [1]. The Černý conjecture remains open since 1971 [4, 5]. It claims that any t-state synchronizing automaton has a reset word of length at most $(t-1)^2$.

In the practical applications of synchronization one may need to compute a reset word for a given automaton and, moreover, the reset word should be as short as possible. The need to compute a synchronizing labeling for a suitable graph, possibly with a request for a short reset word, may arise as well. It turns out, as we describe below, that such computational problems are typically NP-hard, even under heavy restrictions.

Parameterized Complexity offers various notions and useful tools that has became standard in modern analysis of NP-complete problems. The problems are studied with respect to numerical attributes (*parameters*) of instances. A *multi-parameter analysis* considers multiple different parameters and their combinations. In the present paper we close the multi-parameter analyses of canonical problems SYN and SRCP related to synchronization and road coloring. The study of SYN was initiated in 2013 by Fernau, Heggernes, and Villanger [8, 9]. The results about SRCP give complete answers to the questions raised by the second author and Drewienkowski [13, 14].

Since the instances of our problems consist of automata, graphs and word lengths, the natural parameters are: the number t of states (or vertices), the alphabet size $\lvert I \rvert$, and the word length k (see the definitions in Sec. 287 and a summary of the results in Tab. 1 and Tab. 2).

	$k = 2$		$k = 3$		$k = 4, 5, \ldots$	
$\|I\| = 2$	P	[14]	P	♦	NPC	♦
$\|I\| = 3$	P	[14]	P	[13]	NPC	[14]
$\|I\| = 4, 5, \ldots$	P	[14]	P	[13]	NPC	[14]

Tab. 2: Complexity of SRCP restricted to particular values of k and $|I|$

2 Preliminaries

2.1 Automata and synchronization

A *deterministic finite automaton* is a triple $A = (Q, I, \delta)$, where Q and I are finite sets and δ is an arbitrary mapping $Q \times I \to Q$. Elements of Q are called *states*, I is the *alphabet*. The *transition function* δ is naturally extended to $Q \times I^\star \to Q$, still denoted by δ, slightly abusing the notation. We extend it also by defining

$$\delta(S, w) = \{\delta(s, w) \mid s \in S\}$$

for each $S \subseteq Q$ and $w \in I^\star$. If an automaton $A = (Q, I, \delta)$ is fixed, we write

$$r \xrightarrow{x} s$$

instead of $\delta(r, x) = s$.

For a given automaton $A = (Q, I, \delta)$ we call $w \in I^\star$ a *reset word* if

$$|\delta(Q, w)| = 1.$$

If such a word exists, we call the automaton *synchronizing*. Note that each word having a reset word as a factor is also a reset word.

The *Černý conjecture*, a longstanding open problem in automata theory, claims that each synchronizing automaton has a reset word of length at most $(|Q| - 1)^2$. There is a series of automata due to Černý whose shortest reset words reach this bound exactly [4], but all known upper bounds lie in $\Omega\left(|Q|^3\right)$. A tight bound has been established for various special classes of automata, see some of recent advances in [11, 16]. The best general upper bound of the length of shortest reset words is currently the following[(ii)]:

Theorem 1 ([12])**.** *Any t-state synchronizing automaton has a reset word of length $z(t)$, where*

$$z(t) \le \frac{t^3 - t}{6}.$$

It is convenient to analyze synchronization as a process in discrete time. Having an automaton $A = (Q, I, \delta)$ and a word

$$w = x_1 \ldots x_{|w|}$$

(ii) An improved bound published by Trahtman [18] in 2011 has turned out to be proved incorrectly, see [10].

with $x_1, \ldots, x_{|w|} \in I$ fixed, we say that a state $s \in Q$ is *active at time* $l \leq |w|$ if

$$s \in \delta(Q, x_1 \ldots x_l) .$$

At time 0, before the synchronization starts, all states are active. As we apply the letters, the number of active states may decrease.

It is also very intuitive to consider *activity markers*. At time 0 there is an activity marker in each state. Whenever a letter $x \in I$ is applied, each activity marker in state $s \in Q$ moves to $\delta(s, x)$. The state s may become unmarked or receive incoming markers. A state is active if an only if it has an activity marker.

When the number of active states decreases to 1 at a time l, the synchronization is complete and the word $x_1 \ldots x_l$ is a reset word of A.

2.2 Road coloring

In this paper directed multigraphs are referred to as *graphs*. A graph is:

1. *aperiodic* if the lengths of its cycles do not have any nontrivial common divisor,
2. *admissible* if it is aperiodic and all its out-degrees are equal,
3. *road colorable* if its edges can be labeled such that a synchronizing deterministic finite automaton arises.

It is not hard to observe that any road colorable graph is admissible. In 1977 Adler, Goodwyn and Weiss [1] conjectured that for strongly connected graphs the backward implication holds as well. Their question became known as the Road Coloring Problem and a positive answer was given in 2008 by Trahtman [17, 19]:

Theorem 2 (Road Coloring Theorem)**.** *Each strongly connected admissible graph is road colorable.*

In the literature it is common to consider only strongly connected graphs since many general claims can be easily reduced to the corresponding claims about strongly connected cases. We do not admit such restriction explicitly, because in the scope of computational problems it does not seem very natural. However, all the results hold with the restriction as well. Especially the NP-completeness proofs have been made slightly more complicated in order to use only strongly connected graphs.

2.3 Parameterized complexity

In most of the paper we do not need to work with a formal definition of a *parameterized problem*. We see it as a classical decision problem where we consider some special numerical property (*parameter*) of each input. Parameterized complexity studies the way in which the hardness of an NP-complete problem depends on the parameter. A problem may remain NP-hard even if restricted to instances with a particular value of the parameter, or there may be a polynomial-time algorithm for each such value. In the second case, if the algorithm is the same (*uniform*) for all the values, the problem is said to lie in the class *XP*. Moreover, if the time-bounding polynomials for different values are all of the same degree, we get into the class *FPT*:

A parameterized problem is *fixed-parameter tractable* (*FPT*) if there is an algorithm that solves it in time

$$f(P) \cdot r(|x|) ,$$

where x is the input string, $P \in \mathbb{N}$ is the parameter of x, r is an appropriate polynomial, and f is any computable function. If there is more than one possible parameter for a problem, one may consider *combinations* of parameters. A problem is FPT with respect to parameters P, Q if it is decidable in time

$$f(P,Q) \cdot r(|x|)\,.$$

This is typically much less restrictive condition than the previous one, where f depends only on P.

There is a hierarchy of problems (the *W-hierarchy*) lying in XP but possibly outside FPT. It consists of the classes $\mathrm{W}[1]\,, \mathrm{W}[2]\,, \ldots$:

$$\mathrm{FPT} \subseteq \mathrm{W}[1] \subseteq \mathrm{W}[2] \subseteq \cdots \subset \mathrm{XP}. \tag{1}$$

Since it has been conjectured that all the inclusions are proper, it is common to use $\mathrm{W}[i]$-hardness (with respect to an appropriate type of reduction) as the evidence of lying outside FPT. However, we do not need to define the W-hierarchy here since it is used only to formulate former results, not the new ones (see Tab. 1). See the textbook [6] for the definitions and many other great ideas of parameterized complexity.

A *kernel* of a parameterized problem is a polynomial-time procedure that transforms any input x with parameter P_x to another input y with parameter P_y such that both $|y|$ and P_y are bounded by $f(P_x)$, where f is an arbitrary function. Having a kernel is equivalent to lying in FPT. If f is a polynomial, we get a *polynomial kernel*.

2.4 Studied problems

In this paper we work with two canonical computational problems related to synchronization (SYN) and road coloring (SRCP). The problems are defined as follows:

SYN	
Input:	Automaton $A = (Q, I, \delta)$, $k \in \mathbb{N}$
Output:	Is there $w \in I^\star$ of length at most k such that $\lvert\delta(Q,w)\rvert = 1$?
Parameters:	k, $\lvert I\rvert$, $t = \lvert Q\rvert$

SRCP	
Input:	Alphabet I, admissible graph $G = (Q, E)$ with out-degrees $\lvert I\rvert$, $k \in \mathbb{N}$
Output:	Is there a coloring δ such that $\lvert\delta(Q,w)\rvert = 1$ for some $w \in I^\star$ of length at most k?
Parameters:	k, $\lvert I\rvert$, $t = \lvert Q\rvert$

We need the following basic facts related to SYN:

Theorem 3 ([4])**.** *There is a polynomial-time algorithm that decides whether a given automaton is synchronizing.*

Corollary 1. SYN, *if restricted to the instances with* $k \geq z(t) = \frac{t^3-t}{6}$*, is solvable in polynomial time.*

Theorem 4 ([7])**.** SYN *is NP-complete, even if restricted to automata with two-letter alphabets.*

The results of this paper, as well as the former results of Fernau, Heggernes, and Villanger [8, 9] and of the second author and Drewienkowski [13, 14] are summarized by Tables 1 and 2. We have filled all the gaps in the corresponding tables from [8, Sec. 3] and [13, Sec. 6]. This means that the multi-parameter analysis of SYN and SRCP is complete in the sense that NP-complete restrictions are identified and we know, under several standard assumptions, which parameterizations lie in FPT and which of them have polynomial kernels.

3 Parameterized Complexity of SYN

The following lemma, easy to prove using the construction of a power automaton, says that SYN lies in FPT if parameterized by the number of states:

Lemma 1 ([8, 15])**.** *There exists an algorithm that solves* SYN *in time* $r(t, |I|) \cdot 2^t$ *for an appropriate polynomial* r.

But does there exist a polynomial kernel? In this section we use methods developed by Bodlaender et al. [2] to prove the following:

Theorem 5. *If* SYN *parameterized by the number of states has a polynomial kernel, then* $\mathrm{PH} = \Sigma_{\mathrm{p}}^3$*, where* PH *denotes the union of the entire polynomial hierarchy.*

The claim $\mathrm{PH} = \Sigma_{\mathrm{p}}^3$ means that polynomial hierarchy collapses into the third level, which is widely assumed to be false. The key proof method relies on *composition algorithms*. In order to use them immediately we introduce the formalization of our parameterized problem as a set of string-integer pairs:

$$L_{\mathrm{SYN}} = \{(x, t) \mid x \in \Sigma^\star \text{ encodes an instance of SYN with } t \in \mathbb{N} \text{ states}\},$$

where Σ is an appropriate finite alphabet.

3.1 Composition algorithms

An *or-composition algorithm* for a parameterized problem $L \subseteq \Sigma^\star \times \mathbb{N}$ is an algorithm that

- receives as input a sequence $((x_1, t), \ldots, (x_m, t))$ with $(x_i, t) \in \Sigma^\star \times \mathbb{N}^+$ for each $1 \leq i \leq m$,
- uses time polynomial in $\sum_{i=1}^{m} |x_i| + t$
- outputs $(y, t') \subseteq \Sigma^\star \times \mathbb{N}^+$ with

  1. $(y, t') \in L \Leftrightarrow$ there is some $1 \leq i \leq m$ with $(x_i, t) \in L$,
  2. t' is polynomial in t.

Let $L \subseteq \Sigma^\star \times \mathbb{N}$ be a parameterized problem. Its *unparameterized version* is

$$\widehat{L} = \left\{x\#a^t \mid (x, t) \in L\right\},$$

where $\# \notin \Sigma$ is a special symbol.

Theorem 6 ([2]). *Let L be a parameterized problem having an or-composition algorithm. Assume that its unparameterized version $\widehat{L}$ is* NP*-complete. If L has a polynomial kernel, then* $\mathrm{PH} = \Sigma_{\mathrm{p}}^{3}$.

The unparameterized version of L_{SYN} is computationally as hard as the classical SYN, so it is NP-complete. It remains only to describe an or-composition algorithm for L_{SYN}, which is done in the remainder of this section.

3.2 Preprocessing

Let the or-composition algorithm receive an input

$$((A_1, k_1), t), \ldots, ((A_m, k_m), t)$$

consisting of t-state automata $A_1, \ldots, A_m$, each of them equipped with a number k_i. Assume that the following easy procedures have been already applied:

- For each $i = 1, \ldots, m$ such that $k_i \geq z(t)$, use the polynomial-time synchronizability algorithm from Corollary 1 to decide whether $((A_i, k_i), t) \in L_{\mathrm{SYN}}$. If so, return a trivial true instance immediately. Otherwise just delete the i-th member from the sequence.
- For each $i = 1, \ldots, m$, add an additional letter κ to A_i such that κ acts as the identical mapping: $\delta_i(s, \kappa) = s$.
- For each $i = 1, \ldots, m$,rename the states and letters of A_i such that

$$\begin{aligned} A_i &= (Q_i, I_i, \delta_i), \\ Q_i &= \{1, \ldots, t\}, \\ I_i &= \left\{\kappa, a_{i,1}, \ldots, a_{i,|I_i|-1}\right\}. \end{aligned}$$

After that, our algorithm chooses one of the following procedures according to the length m of the input sequence:

- If $m \geq 2^t$, use the exponential-time algorithm from Lemma 1: denote $D = \sum_{i=1}^{m} |(A_i, k_i)| + t$, where we add lengths of descriptions of the pairs. Note that $D \geq m \geq 2^t$ and that D is the quantity used to restrict the running time of or-composition algorithms. By the lemma, in time

  $$\sum_{i=1}^{m} r(t, |I_i|) \cdot 2^t \leq m \cdot r(D, D) \cdot 2^t \leq D^2 \cdot r(D, D)$$

  we are able to analyze all the m automata and decide if some of them have a reset word of the corresponding length. It remains to output some appropriate trivial instance $((A', k'), t')$.
- If $m < 2^t$, put $q(m) = \lfloor \log(m+1) \rfloor$. It follows that $q(m) \leq t + 2$. On the output of the or-composition algorithm we put $((A', k'), t')$, where A' is the automaton described in the following paragraphs and

  $$d' = z(t) + 1$$

  is our choice of the maximal length of reset words to be found in A'.

3.3 Construction of A' and its ideas

Here we describe the automaton A' that appears in the output of our or-composition algorithm. We set

$$\begin{aligned} A' &= (Q', I', \delta')\,, \\ Q' &= \{1,\ldots,t\} \cup \{\mathrm{D}\} \cup (\{0,\ldots,z(t)\} \times \{0,\ldots,q(m)\} \times \{\mathrm{T},\mathrm{F}\})\,, \\ I' &= \left(\bigcup_{i=1}^{m} I_i\right) \cup \{\alpha_1,\ldots,\alpha_m\} \cup \{\omega_1,\ldots,\omega_t\}\,. \end{aligned}$$

The letters from $\bigcup_{i=1}^{m} I_i$ act on the states $1,\ldots,t$ in the following way:

$$s \xrightarrow{x_{i,j}} \delta_i(s, a_{i,j})$$

for each $s \in \{1,\ldots,t\}$, $i \in \{1,\ldots,m\}$, $a_{i,j} \in I_i$. In other words, we let all the letters from all the automata $A_1,\ldots,A_m$ act on the states $1,\ldots,t$ just as they did in the original automata. The additional letters act on $1,\ldots,t$ simply as well:

$$s \xrightarrow{\alpha_i} s, \qquad \overline{s} \xrightarrow{\omega_s} \begin{cases} \mathrm{D} & \text{if } \overline{s} = s, \\ \overline{s} & \text{otherwise} \end{cases}$$

for each $s, \overline{s} \in \{1,\ldots,t\}$, $i \in \{1,\ldots,m\}$. The state D is a *sink state*, which means that

$$\mathrm{D} \xrightarrow{y} \mathrm{D}$$

for each $y \in I'$. Note that any reset word of A' has to map all the states of Q' to D.

The remaining $2\cdot(z(t)+1)\cdot(q(m)+1)$ states form what we call a *guard table*. Its purpose is to guarantee that:

(C1) Any reset word of A' has to be of length at least $k' = z(t)+1$.

(C2) Any reset word w of A' having length exactly $z(t)+1$ is of the form

$$w = \alpha_i y_1 \ldots y_{k_i} \kappa^{z(t)-1-k_i} \omega_s \tag{2}$$

for some $i \in \{1,\ldots,m\}$, $y_1,\ldots,y_{k_i} \in I_i$, and $s \in \{1,\ldots,t\}$ such that $y_1 \ldots y_{k_i}$ is a reset word of A_i.

(C3) Any word w

- of length $k' = z(t)+1$,
- of the form (2),
- and satisfying $\delta_i(Q_i, y_1 \ldots y_{k_i}) = \{s\}$,

is a reset word of A'.

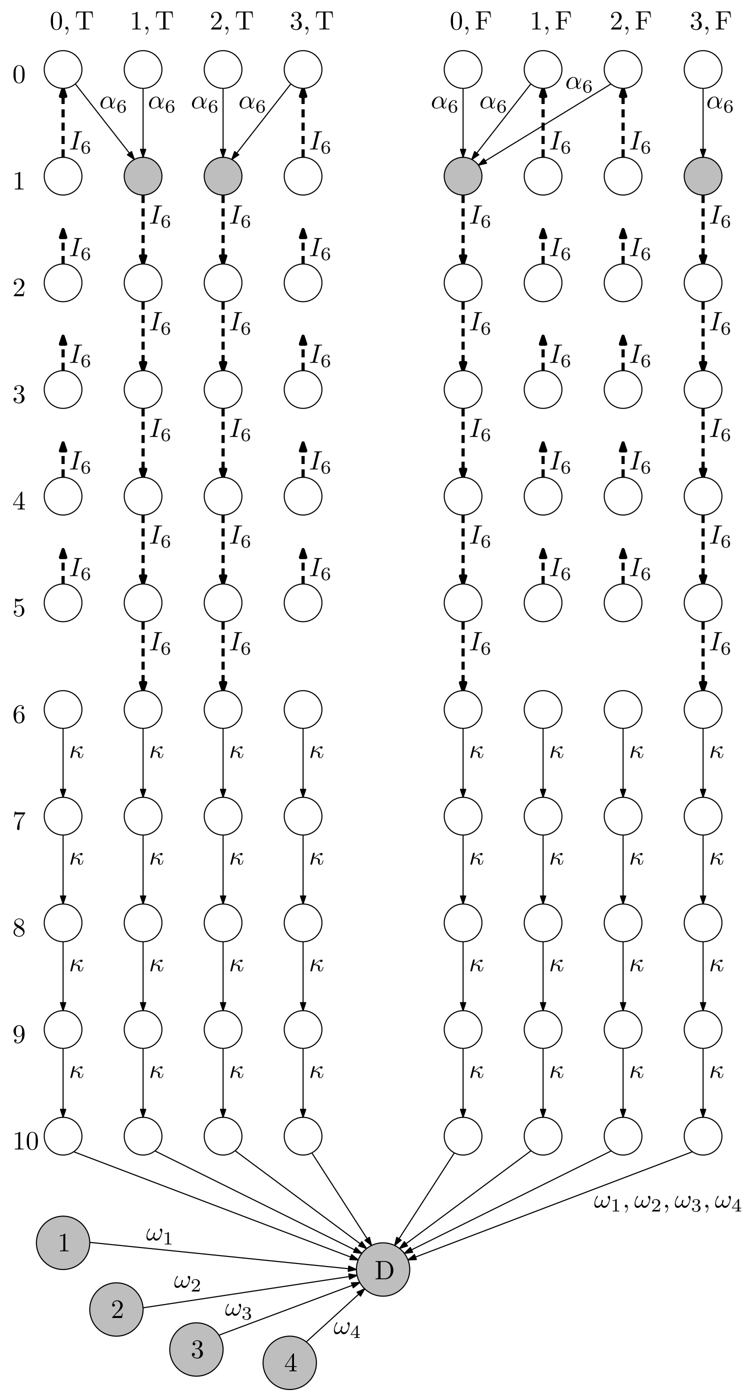

Fig. 1: Some transitions of the example automaton described in Sec. 3.4. Grey states remain active after applying α_6.

If the guard table manages to guarantee these three properties of A', we are done: It is easy to check that they imply all the conditions given in the definition of a composition algorithm. So, let us define the action of the letters from I' on the states from $\{0,\ldots,z(t)\}\times\{0,\ldots,q(m)\}\times\{\mathrm{T},\mathrm{F}\}$. After that A' will be complete and we will check the properties C1, C2, C3.

The actions of $\alpha_1,\ldots,\alpha_m$ should meet the following two conditions:

- Any reset word w of length $z(t)+1$ has to start with some α_i.
- After applying the first letter α_i there must occur at least $z(t)-1$ consecutive letters from I_i. Informally, by applying α_i we *choose* the automaton A_i.

How to do that? The number m may be quite large and each of $\alpha_1,\ldots,\alpha_m$ needs to have a unique effect. The key tool is what we call *activity patterns*. Let us work with the set

$$R=\{0,\ldots,q(m)\},$$

which matches „half of a row" of the guard table. Subsets of R correspond in a canonical way to binary representations of numbers $0,\ldots,2^{q(m)+1}-1$. We will actually represent only the numbers $1,\ldots,m$. These does not include any of the extreme values corresponding to the empty set and whole R, because we have $m<2^{q(m)+1}-1$. Let the mapping

$$\mathfrak{b}:\{1,\ldots,m\}\to 2^R$$

assign the corresponding subset of R to a number. For instance, it holds that

$$\mathfrak{b}(11)=\{0,1,3\}$$

because $11=2^0+2^1+2^3$. For each $i\in\{1,\ldots,m\}$ we define specific *pattern functions*

$$\pi_i^{\mathrm{T}},\pi_i^{\mathrm{F}}:R\to R$$

such that

$$\begin{aligned}\operatorname{rng}\pi_i^{\mathrm{T}} &= \mathfrak{b}(i),\\ \operatorname{rng}\pi_i^{\mathrm{F}} &= R\backslash\mathfrak{b}(i)\end{aligned}$$

for each i. It is irrelevant how exactly π_i^{T} and π_i^{F} are defined. It is sure that they exist, because the range is never expected to be empty. The action of the letters $\alpha_1,\ldots,\alpha_m$ is

$$\begin{aligned}(h,p,\mathrm{T}) &\xrightarrow{\alpha_i} \left(1,\pi_i^{\mathrm{T}}(p),\mathrm{T}\right),\\ (h,p,\mathrm{F}) &\xrightarrow{\alpha_i} \left(1,\pi_i^{\mathrm{F}}(p),\mathrm{F}\right),\end{aligned}$$

for each $i\in\{1,\ldots,m\}$, $h\in\{0,\ldots,z(t)\}$, and $p\in R$.

Note that each α_i maps the entire guard table, including row 0, into row 1. In fact, all „downward" transitions within the guard table will lead only one row down and the only transitions escaping from the guard table will lead from the bottom row. Thus, any reset word will have length at least $k'=z(t)+1$. Moreover, during its application, at time l the rows $0,\ldots,l-1$ will have to be all inactive. This is a key mechanism that the guard table uses for enforcing necessary properties of short reset words.

Let us define how the letters $a_{i,j}$ act on the guard table. Choose any $i\in\{1,\ldots,m\}$. The action of $a_{i,j}$ within the guard table does not depend on j. All the letters coming from a single automaton act identically here:

- for the rows $h \in \{1, \dots, k_i\}$ we set

$$(h, p, \mathrm{T}) \xrightarrow{a_{i,j}} \begin{cases} (h+1, p, \mathrm{T}) & \text{if } p \in \mathfrak{b}(i)\,, \\ (0, p, \mathrm{T}) & \text{otherwise,} \end{cases}$$

$$(h, p, \mathrm{F}) \xrightarrow{a_{i,j}} \begin{cases} (h+1, p, \mathrm{F}) & \text{if } p \notin \mathfrak{b}(i)\,, \\ (0, p, \mathrm{F}) & \text{otherwise,} \end{cases}$$

- for the remaining rows $h \in \{0\} \cup \{k_i + 1, \dots, z(t)\}$ we set

$$(h, p, \mathrm{T}) \xrightarrow{a_{i,j}} (0, p, \mathrm{T})\,,$$
$$(h, p, \mathrm{F}) \xrightarrow{a_{i,j}} (0, p, \mathrm{F})\,.$$

Recall that sending an activity marker along any transition ending in the row 0 is a „suicide”. A word that does this cannot be a short reset word. So, if we restrict ourselves to the letters from some I_i, the transitions defined above imply that the forthcoming letter can be $a_{i,j}$ only at times $1, \dots, k_i$. In the following $z(t) - k_i - 1$ steps the only letter from I_i that can be applied is κ.

The letter κ maps all the states of the guard table simply one state down, except for the rows 0 and $z(t)$. Set

$$(h, p, \mathrm{T}) \xrightarrow{\kappa} (h+1, p, \mathrm{T})\,,$$
$$(h, p, \mathrm{F}) \xrightarrow{\kappa} (h+1, p, \mathrm{F})$$

for each $h \in \{1, \dots, z(t) - 1\}$, and

$$(0, p, \mathrm{T}) \xrightarrow{\kappa} (0, p, \mathrm{T})\,,$$
$$(0, p, \mathrm{F}) \xrightarrow{\kappa} (0, p, \mathrm{F})\,,$$
$$(z(t)\,, p, \mathrm{T}) \xrightarrow{\kappa} (0, p, \mathrm{T})\,,$$
$$(z(t)\,, p, \mathrm{F}) \xrightarrow{\kappa} (0, p, \mathrm{F})\,.$$

It remains to describe the actions of the letters $\omega_1, \dots, \omega_t$ on the guard table. Set

$$(z(t)\,, p, \mathrm{T}) \xrightarrow{\omega_s} \mathrm{D},$$
$$(z(t)\,, p, \mathrm{F}) \xrightarrow{\omega_s} \mathrm{D}$$

for each $p \in R$, $s \in \{1, \dots, t\}$, and

$$(h, p, \mathrm{T}) \xrightarrow{\omega_s} (0, p, \mathrm{T})\,,$$
$$(h, p, \mathrm{F}) \xrightarrow{\omega_s} (0, p, \mathrm{F})$$

for each $h \in \{0, \dots, z(t) - 1\}$, $p \in R$, and $s \in \{1, \dots, t\}$. Now the automaton A' is complete.

3.4 An example

Consider an input consisting of $m = 12$ automata $A_1, \ldots, A_{12}$, each of them having $t = 4$ states. Because $z(4) = 10$ and $q(12) = 3$, the output automaton A' has 93 states in total. In Fig. 1 all the states are depicted, together with some of the transitions. We focus on the transitions corresponding to the automaton A_6, assuming that $k_6 = 5$.

The action of α_6 is determined by the fact that $6 = 2^1 + 2^2$ and thus

$$\begin{aligned} \operatorname{rng} \pi_6^{\mathrm{T}} &= \mathfrak{b}(6) = \{1, 2\}, \\ \operatorname{rng} \pi_6^{\mathrm{F}} &= R \backslash \mathfrak{b}(6) = \{0, 3\}. \end{aligned}$$

If the first letter of a reset word is α_6, when applied, only the states

$$(1, 1, \mathrm{T}), (1, 2, \mathrm{T}), (1, 0, \mathrm{F}), (1, 3, \mathrm{F})$$

remain active within the guard table. Now we need to move their activity markers one row down in each of the following $z(t) - 1 = 9$ steps. The only way to do this is to apply $k_6 = 5$ letters of I_6 and then $z(t) - 1 - k_6 = 4$ occurrences of κ. Then we are allowed to apply one of the letters $\omega_1, \ldots, \omega_t$. But before that time there should remain only one active state $s \in \{1, \ldots, t\}$, so that we could use ω_s. The letter κ does not affect the activity within $\{1, \ldots, t\}$ so we need to synchronize these states using $k_6 = 5$ letters from I_6.

So, any short reset word of A' starting with α_6 has to contain a short reset word of A_6.

3.5 The guard table works

It remains to use the ideas informally outlined in Sec. 3.3 to prove that A' has the properties C1, C2, and C3 from Sec. 3.3.

Proof C1: As it has been said, for each $x \in I'$ and each state (h, p, Q), where $\mathrm{Q} \in \{\mathrm{T}, \mathrm{F}\}$ and $h \in \{0, \ldots, z(t) - 1\}$, it holds that

$$(h, p, \mathrm{Q}) \xrightarrow{x} (h', p', \mathrm{Q}),$$

where $h' < h$ or $h' = h + 1$. So the shortest paths from the row 0 to the state D have lengths at least $z(t) + 1$. $\square$

Proof C2: We should prove that any reset word w having length exactly $z(t) + 1$ is of the form

$$w = \alpha_i y_1 \ldots y_{k_i} \kappa^{z(t) - 1 - k_i} \omega_s,$$

where $y_1 \ldots y_{k_i}$ is a reset word of A_i. The starting α_i is necessary, because $\alpha_1, \ldots, \alpha_m$ are the only letters that map states from the row 0 to other rows. Denote the remaining $z(t)$ letters of w by $y_1, \ldots, y_{z(t)}$.

Once an α_i is applied, only $|R| = q(m) + 1$ active states in the guard table remain. All of them are in the row 1, depending on i. The active states are exactly from

$$\{1\} \times \mathfrak{b}(i) \times \{\mathrm{T}\} \text{ and } \{1\} \times R \backslash \mathfrak{b}(i) \times \{\mathrm{F}\},$$

because this is exactly the range of α_i within the guard table. Let us continue by an induction. We claim that for $0 \leq \tau < k_i$ it holds what we have already proved for $\tau = 0$:

1. If $\tau \geq 1$, the letter y_τ lies in I_i. Moreover, if $\tau > k_i$, it holds that $w_\tau = \kappa$.
2. After the application of y_τ the active states within the guard table are exactly from

$$\{\tau+1\} \times \mathfrak{b}(i) \times \{\mathrm{T}\} \text{ and } \{\tau+1\} \times R \backslash \mathfrak{b}(i) \times \{\mathrm{F}\}\,.$$

For $\tau = 0$ both claims hold. Take some $1 \leq \tau < k_i$ and suppose that the claims hold for $\tau - 1$. Let us use the second claim for $\tau - 1$ to prove the first claim for τ. All the states from

$$\{\tau\} \times \mathfrak{b}(i) \times \{\mathrm{T}\} \text{ and } \{\tau\} \times R \backslash \mathfrak{b}(i) \times \{\mathrm{F}\}$$

are active. Which letter may appear as y_τ? The letters $\omega_1, \ldots, \omega_t$ and $\alpha_1, \ldots, \alpha_m$ would map all the active states to rows 0 and 1, which is a contradiction. Consider any letter $a_{l,j}$ for $l \neq i$. It holds that $\mathfrak{b}(i) \neq \mathfrak{b}(l)$, so there is some $p \in R$ lying in their symmetrical difference. For such p it holds that

$$(\tau, p, \mathrm{T}) \xrightarrow{a_{l,j}} (0, p, \mathrm{T}) \text{ if } p \in \mathfrak{b}(i) \backslash \mathfrak{b}(l)$$

or

$$(\tau, p, \mathrm{F}) \xrightarrow{a_{l,j}} (0, p, \mathrm{F}) \text{ if } p \in \mathfrak{b}(l) \backslash \mathfrak{b}(i)\,.$$

This necessarily activates some state in the row 0, which is a contradiction again. So, $y_\tau \in I_i$. Moreover, if $\tau > k_i$, the letters from $I_i \backslash \{\kappa\}$ map the entire row τ into the row 0, so the only possibility is $y_\tau = \kappa$.

The letter y_τ maps all the active states right down to the row $\tau + 1$, so the second claim for τ holds as well. □

Proof C3: It is easy to verify that no „suicidal" transitions within the guard table are used, so during the application of

$$y_1 \ldots y_{k_i} \kappa^{z(t)-1-k_i}$$

the activity markers just flow down from the row 1 to the row $z(t)$. Since $y_1 \ldots y_{k_i}$ is a reset word of A_i, there remains only one particular state s within $\{1, \ldots, t\}$. Finally, the letter ω_s is applied that maps s and the entire row $z(t)$ directly to D. □

4 Parameterized Complexity of SRCP

4.1 Parameterization by the number of states

We point out that SRCP parameterized by the number of states has a polynomial kernel, so it necessarily lies in FPT.

Theorem 7. *There is a polynomial kernel for* SRCP *parameterized by* $t = |Q|$.

Proof: The algorithm takes an instance of SRCP, i.e., an alphabet I, an admissible graph $G = (Q, E)$ with out-degrees $|I|$, and a number $k \in \mathbb{N}$. It produces another instance of size depending only on $t = |Q|$. If

$k \geq z(t)$, we just solve the problem using Corollary 1 and output a trivial instance. Otherwise the output instance is denoted by $I', G' = (Q', E'), k'$, where

$$\begin{aligned} Q' &= Q, \\ k' &= k, \\ |I'| &= \min\{|I|, t \cdot (z(t)-1)\}, \end{aligned}$$

and the algorithm just deletes appropriate edges in order to reduce the out-degree to $|I'|$. Let us use a procedure that:

- takes an admissible graph with out-degree $d > t \cdot (z(t) - 1)$
- for each of its vertices:
  - finds an outgoing multiedge with the largest multiplicity (which is at least $z(t)$),
  - deletes one edge from the multiedge.

Clearly the resulting graph has out-degree $d - 1$. We create the graph G' by repeating this procedure (starting with G) until the out-degree is at most $t \cdot (z(t) - 1)$.

Now we claim that

$$\begin{gathered} (I, G, k) \in \mathrm{SRCP} \\ \Updownarrow \\ (I', G', k') \in \mathrm{SRCP}. \end{gathered}$$

The upward implication is trivial since any coloring of G' can be extended to G and the appropriate reset word can be still used. On the other hand, let us have a coloring δ of G such that $|\delta(Q, w)| = 1$ for a word w of length at most $k < z(t)$, so it uses at most $z(t) - 1$ letters from I. If we delete from G all the edges labeled by non-used letters, we get a subgraph of G' because during the reduction of edges we have reduced only multiedges having more than $z(t) - 1$ edges. So we are able to color G' according to the used letters of G and synchronize it by the word w. □

Corollary 2. SRCP *parameterized by* $t = |Q|$ *lies in* FPT.

4.2 Restriction to $|I| = 2$ and $k = 3$

Here we prove that SRCP restricted to $|I| = 2$ and $k = 3$ is decidable in polynomial time. If $G = (Q, E)$ is a graph, by $V_i(q)$ we denote the set of vertices from which there is a path of length i leading to q and there is no shorter one. For any $w \in I^*$ by $\mathbb{G}_w$ we denote the set of graphs with outdegree 2 that admit a coloring δ such that $\delta(Q, w) = \{q\}$ for some $q \in Q$.

Lemma 2. *A graph* $G = (Q, E)$ *lies in* $\mathbb{G}_{aaa}$ *if and only if there is a state* $q \in Q$ *such that* $(q, q) \in E$ *and there is a path of length at most* 3 *from each* $r \in Q$ *to* q*.*

Proof: If there exists a coloring δ with $\delta(Q, aaa) = \{q\}$, both claims follow immediately. On the other hand, if the loop (q, q) and all the paths leading to q are colored by a, we obtain a suitable coloring. □

Lemma 3. *Let $G = (Q, E) \in \mathbb{G}_{abb} \backslash \mathbb{G}_{aaa}$. Then at least one of the following two conditions holds:*

1. *There is a vertex $q \in Q$ such that each vertex has an outgoing edge leading to $V_2(q)$.*
2. *$G \in \mathbb{G}_{aba}$.*

Proof: Let $G = (Q, E) \in \mathbb{G}_{abb} \backslash \mathbb{G}_{aaa}$. Thus G admits a coloring δ such that

$$\delta(Q, abb) = \{q\}$$

for $q \in Q$.

- If δ satisfies $q \notin \delta(Q, a)$, each edge labeled by a has to lead to $V_2(q)$. Indeed:
  - It cannot lead to q due to $q \notin \delta(Q, a)$.
  - It cannot lead to any $r \in V_1(q)$ because in such case, using $q \notin \delta(Q, a)$, it would hold that $\delta(r, b) = q$ and thus $q \in \delta\,(Q, ab)$. Hence, it would be necessary that $\delta\,(q, b) = q$. According to Lemma 2, a loop on q guarantees that $G \in \mathbb{G}_{aaa}$, which is a contradiction.
  - It cannot lead to $V_3(q)$, because there is no path of length 2 from $V_3(q)$ to q.

  Thus, the condition (1) holds.

- Otherwise the coloring δ satisfies $q \in \delta(Q, a)$. Put

  $$W = \{s \in Q \mid \delta(s, b) = q\}\,.$$

  Now define another coloring δ' by switching the colors of the two edges leaving each state of W. Elsewhere, δ' and δ coincide. We claim that

  $$\delta'(Q, aba) = \{q\}$$

  and so the condition (2) holds. Indeed:

  - Take $s \in V_3(q)$. In δ there is a path

    $$s \xrightarrow{a} t \xrightarrow{b} u \xrightarrow{b} q. \tag{3}$$

    Because $s \in V_3(q)$, it holds that $t \in V_2(q)$ and $u \in V_1(q)$. It follows that $t \notin W, u \in W$ and thus in δ' there is a path

    $$s \xrightarrow{a} t \xrightarrow{b} u \xrightarrow{a} q. \tag{4}$$

  - Take $s \in V_2(q)$. In δ there is a path (3).
    - If $t \in V_2(q)$, we get again that $t \notin W, u \in W$ and thus there is a path (4) in δ'.
    - Otherwise, we have $t \in V_1(q)$. Because $G \notin \mathbb{G}_{aaa}$, there is no loop on q, thus $u \neq q$ and $t \notin W$. But $u \in W$, so we get a path (4) again.

– Take $s \in V_1(q)$. In δ' there is always an edge $s \xrightarrow{a} q$, so we need $\delta'(q, ba) = q$. Because we assume that $q \in \delta(Q, a)$, there has to be a cycle $q \xrightarrow{b} r \xrightarrow{b} q$ in δ for some $r \in V_1(q)$. In δ' we have $q \xrightarrow{b} r \xrightarrow{a} q$.

– For $s = q$ we apply the same reasoning as for $s \in V_2(q)$.

$\square$

Lemma 4. *For each G with outdegree 2 it holds that*

$$G \in \mathbb{G}_{abb} \setminus (\mathbb{G}_{aba} \cup \mathbb{G}_{aaa})$$

if and only if

- it holds that $G \notin \mathbb{G}_{aba} \cup \mathbb{G}_{aaa}$, and
- there is a vertex $q \in Q$ such that each vertex has an outgoing edge leading to $V_2(q)$.

Proof: The downward implication follows easily from Lemma 3. For the upward one we only need to deduce that $G \in \mathbb{G}_{abb}$. We construct the following coloring δ:

- The edges leading to $V_2(q)$ are labeled by a. If two such edges start in a common vertex, they are labeled arbitrarily.
- The other edges are labeled by b.

This works because from any state $s \in V_2(q)$ there is an edge leading to some $t \in V_1(q)$ and from t there is an edge leading to q. We have labeled both these edges by b. It follows that wherever we start, the path labeled by abb leads to q. $\square$

Theorem 8. *SRCP with $|I| = 2$ and $k = 3$ lies in P.*

Proof: Let the algorithm test the membership of a given graph G for the following sets:

1. $\mathbb{G}_{aaa}$,
2. $\mathbb{G}_{aab} \setminus \mathbb{G}_{aaa}$,
3. $\mathbb{G}_{aba} \setminus \mathbb{G}_{aaa}$,
4. $\mathbb{G}_{abb} \setminus (\mathbb{G}_{aba} \cup \mathbb{G}_{aaa})$.

For the sets 1, 2, 3 the membership is polynomially testable due to the results from [13]. Lemma 4 provides a polynomially testable characterization of the set 4. It is easy to see that a graph G should be accepted if and only if it lies in some of the sets. $\square$

4.3 Restriction to $|I| = 2$ and $k = 4$

Theorem 9. *SRCP remains NP-complete if restricted to $|I| = 2$ and $k = 4$.*

Let us perform a reduction from 3-SAT. Consider a propositional formula of the form

$$\Phi = \bigwedge_{j=1}^{m} \mathcal{C}_j \,,$$

where

$$\mathcal{C}_j = l_{i,1} \vee l_{i,2} \vee l_{i,3}$$

and

$$l_{j,k} \in \{x_1, \ldots, x_n, \overline{x_1}, \ldots, \overline{x_n}\}$$

for each $j \in \{1, \ldots, m\}$ and $k \in \{1, 2, 3\}$.

We construct a directed multigraph $G_\Phi = (Q, E)$ with

$$|Q| = 5m + 3n + 8$$

states, each of them having exactly two outgoing edges. We describe Q as a disjoint union of the following sets:

$$Q = \mathbf{C}_1 \cup \cdots \cup \mathbf{C}_m \cup \mathbf{V}_1 \cup \cdots \cup \mathbf{V}_n \cup \mathbf{D},$$

where

$$\begin{aligned} \mathbf{C}_j &= \{\mathrm{C}_{j,0}, \mathrm{C}_{j,1}, \mathrm{C}_{j,2}, \mathrm{C}_{j,3}, \mathrm{C}_{j,4}\}\,, \\ \mathbf{V}_i &= \{x_i, \overline{x_i}, \mathrm{W}_i\}\,, \\ \mathbf{D} &= \{\mathrm{D}_0, \ldots, \mathrm{D}_7\}\,, \end{aligned}$$

for each $j \in \{1, \ldots, m\}$ and $i \in \{1, \ldots, n\}$. The parts $\mathbf{C}_j$ correspond to clauses, the parts $\mathbf{V}_i$ correspond to variables. In each $\mathbf{V}_i$ there are two special states labeled by literals x_i and $\overline{x_i}$. All the edges of G_Φ are defined by Figures 2, 3, 4.

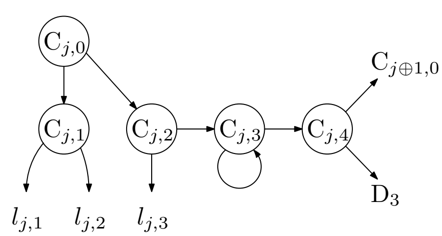

Fig. 2: The part $\mathbf{C}_j$. Note the edges that depend on Φ: they end in vertices labeled by literals from $\mathcal{C}_j$.

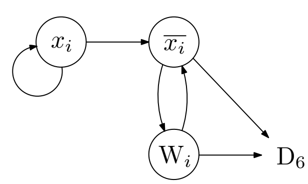

Fig. 3: The part $\mathbf{V}_i$

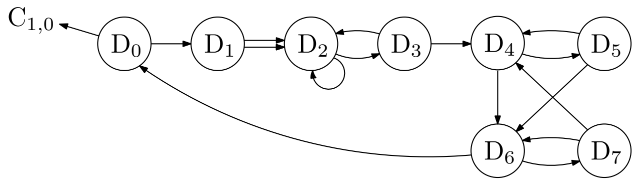

Fig. 4: The part $\mathbf{D}$

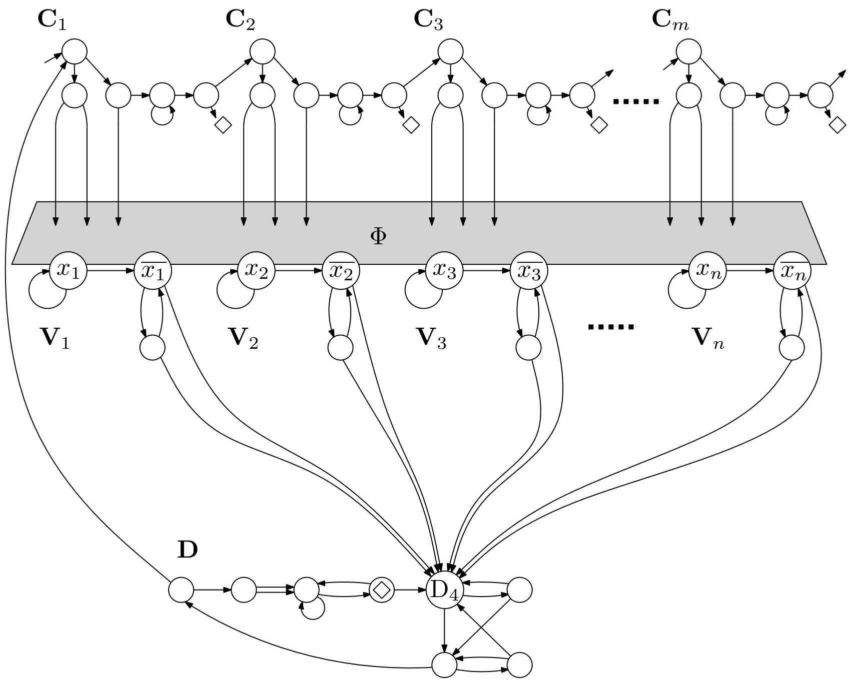

Fig. 5: The entire G_Φ

Figure 5 gives the overall picture of G_Φ. Let us prove that

$$\Phi \text{ is satisfiable}$$
$$\Updownarrow$$
$$\text{some labeling of } G_\Phi \text{ has a reset word of length } 4.$$

The upward implication

Suppose that there is a labeling δ by letters a (solid) and b (dotted) such that

$$w = y_1 \ldots y_4 \in \{a, b\}^4$$

satisfies

$$|\delta(Q, w)| = 1.$$

Let a be the first letter of w. By a *k-path* (resp. *k-reachable*) we understand a path of length exactly k (resp. reachable by a path of length exactly k).

Lemma 5. *The synchronization takes place in* D_4.

Proof: From D_1 only states from $\mathbf{D}$ are 4-reachable. From D_0 the only states within $\mathbf{D}$ that are 4-reachable are $\mathrm{D}_2, \mathrm{D}_3, \mathrm{D}_4$. From $\mathrm{C}_{1,0}$ only D_4 is 4-reachable. □

Lemma 6. *All edges outgoing from states of* $\mathbf{D}$ *are labeled as in Figure 6.*

Proof: Since D_4 is not 3-reachable from D_0 nor D_6, all the edges incoming to D_0 and D_6 are labeled by b. The remaining labeling follows easily. □

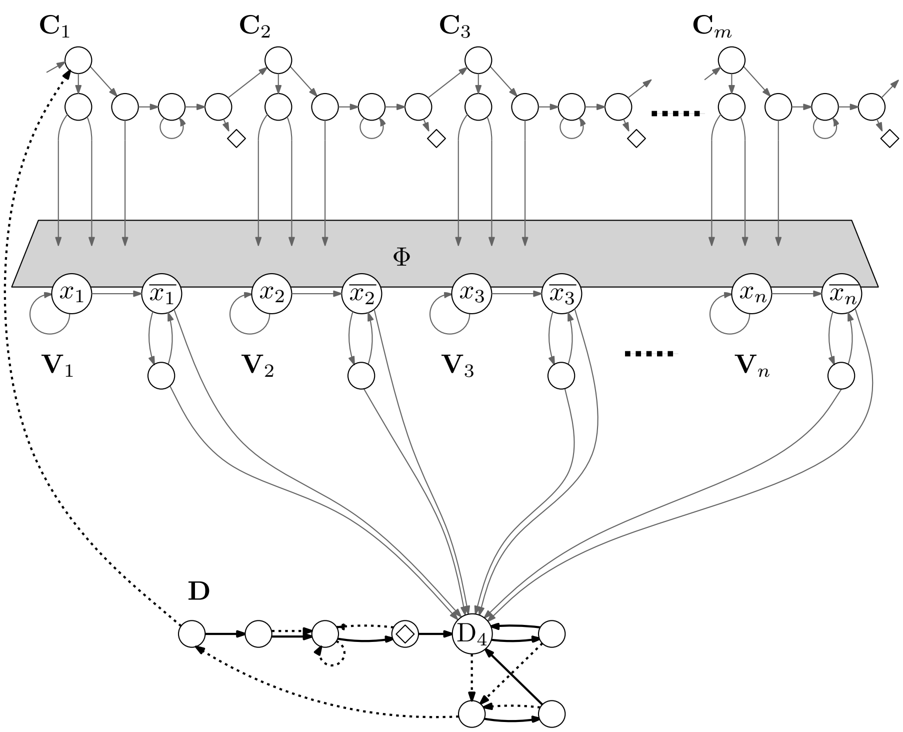

Fig. 6: The entire G_Φ with the edges outgoing from D colored. Bold arrows: a, dotted arrows: b.

Corollary 3. *It holds that*

$$w = aba^2.$$

Lemma 7. *For each $j = 1, \dots, m$ we have*

$$\delta(\mathrm{C}_{j,0}, ab) \in \{x_1, \dots, x_n, \overline{x_1}, \dots, \overline{x_n}\}.$$

Proof: None of the other states 2-reachable from $\mathrm{C}_{j,0}$ offers a 2-path leading to D_4. □

Lemma 8. *There are no $j, l \in \{1, \dots, m\}$ and $i \in \{1, \dots, n\}$ such that*

$$\delta(\mathrm{C}_{j,0}, ab) = x_i$$

and

$$\delta(C_{l,0}, ab) = \overline{x_i}.$$

Proof: If both x_i and $\overline{x_i}$ are active after applying $y_1 y_2 = ab$, there have to be 2-paths labeled by a^2 from both $x_i, \overline{x_i}$ to D_4. It is easy to see that it is not possible to find such labeling. □

Corollary 4. *There is a partial assignment making all the literals*

$$\delta(\mathrm{C}_{1,0}, ab), \delta(\mathrm{C}_{2,0}, ab), \dots, \delta(\mathrm{C}_{m,0}, ab)$$

satisfied, because none of them is the negation of another. Each clause contains some of these literals.

We are done - the existence of a satisfying assignment is guaranteed.

The downward implication

For a given satisfying assignment we construct a coloring based on the above-mentioned ideas and the example given by Fig. 7.

- For each j, the coloring of edges outgoing from $\mathrm{C}_{j,0}, \mathrm{C}_{j,1}, \mathrm{C}_{j,2}$ depends on which of the three literals of $\mathcal{C}_j$ are satisfied by the assignment (the example assigns $x_1 = \mathbf{1}, x_2 = \mathbf{0}, x_3 = \mathbf{0}, x_4 = \mathbf{1}$). The 2-path from $\mathrm{C}_{j,0}$ labeled by ab should lead to a state labeled by a satisfied literal. The edges outgoing from $\mathrm{C}_{j,3}$ and $\mathrm{C}_{j,4}$ are always colored in the same way.
- For each i, all the edges outgoing from the states of the $\mathbf{V}_i$ part are colored in one of two ways depending on the truth value assigned to x_i.
- The edges outgoing from the states of $\mathbf{D}$ admit the only possible coloring.

Note that in our example the edges outgoing from the states of $\mathbf{V}_3$ could be colored in the opposite way as well. None of the literals $x_3, \overline{x_3}$ is chosen by the coloring to satisfy a clause.

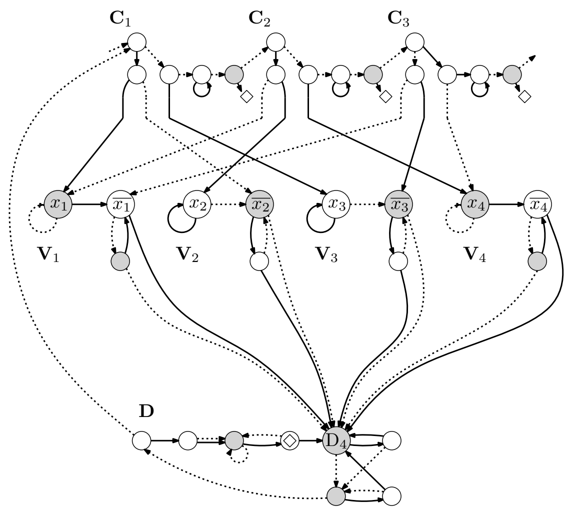

Fig. 7: An example of G_Φ for $\Phi = (x_1 \vee \overline{x_2} \vee x_3) \wedge (x_1 \vee x_2 \vee x_4) \wedge (\overline{x_1} \vee \overline{x_3} \vee x_4)$. The filling marks the states that are active after applying $y_1 y_2 = ab$.

Strong connectivity

If there is a non-negated occurrence of each x_i in Φ, the graph G_Φ is strongly connected. This assumption can be easily guaranteed by adding tautological clauses like $x_i \vee \overline{x_i} \vee \overline{x_i}$.

5 Further Research: SRCW

Part of the RSCP input consists of a prescribed length of a reset word that should be used in the road coloring. But what if an exact reset word or a set of possible reset words were prescribed? This problem is called SRCW:

SRCW	
Input:	Alphabet I, admissible graph $G = (Q, E)$ with out-degrees $\lvert I \rvert$, $W \subseteq I^\star$
Output:	Is there a coloring δ such that $\lvert\delta(Q, w)\rvert = 1$ for some $w \in W$?

In [22] we study SRCW with respect to fixed values of both $|I|$ and W. For $|I| = 2$, singleton sets W that make the problem NP-complete are characterized. We also study the situation with the additional restriction to strongly connected graphs. We show that the complexity differs in the case $|I| = 2$ and $W = \{abb\}$. However, with such restriction the characterization remains incomplete. There are also no results about non-binary alphabets and non-singleton sets W. The current state of art is mostly depicted in Tab. 3.

<table>
<tr><th></th><th colspan="2">$|I| = 2$</th><th colspan="2">$|I| = 3, 4, \ldots$</th></tr>
<tr><th></th><th>general</th><th>S.C.</th><th>general</th><th>S.C.</th></tr>
<tr><td>$W = \{a^k\}$ $(k \geq 1)$</td><td>P</td><td>P</td><td>P</td><td>P</td></tr>
<tr><td>$W = \{a^k b\}$ $(k \geq 1)$</td><td>P</td><td>P</td><td>P</td><td>P</td></tr>
<tr><td>$W = \{abb\}$</td><td>NPC</td><td>P</td><td rowspan="3">?</td><td rowspan="3">?</td></tr>
<tr><td>$W = \{ava\}$ $(v \notin a^\star)$</td><td>NPC</td><td>NPC</td></tr>
<tr><td>other forms of W</td><td>NPC</td><td>?</td></tr>
</table>

Tab. 3: Complexity of SRCW restricted to fixed $|I|$, fixed singleton W, and with possible assumption of strong connectivity (S.C.). The results come from [13] and [22].

References

[1] Adler, R., Goodwyn, L., Weiss, B.: Equivalence of topological Markov shifts. Israel Journal of Mathematics 27(1), 49–63 (1977)

[2] Bodlaender, H.L., Downey, R.G., Fellows, M.R., Hermelin, D.: On problems without polynomial kernels. Journal of Computer and System Sciences 75(8), 423 – 434 (2009)

[3] Bonizzoni, P., Jonoska, N.: Regular splicing languages must have a constant. In: Mauri, G., Leporati, A. (eds.) Developments in Language Theory, Lecture Notes in Computer Science, vol. 6795, pp. 82–92. Springer Berlin Heidelberg (2011)

[4] Černý, J.: Poznámka k homogénnym experimentom s konečnými automatmi. Matematicko-fyzikálny časopis 14(3), 208–216 (1964)

[5] Černý, J., Pirická, A., Rosenauerová, B.: On directable automata. Kybernetica 7, 289–298 (1971)

[6] Downey, R.G., Fellows, M.R.: Fundamentals of Parameterized Complexity. Springer-Verlag (2013), 763 pp.

[7] Eppstein, D.: Reset sequences for monotonic automata. SIAM J. Comput. 19(3), 500–510 (1990)

[8] Fernau, H., Heggernes, P., Villanger, Y.: A multivariate analysis of some DFA problems. In: Dediu, A.H., Martín-Vide, C., Truthe, B. (eds.) Language and Automata Theory and Applications, Lecture Notes in Computer Science, vol. 7810, pp. 275–286. Springer Berlin Heidelberg (2013)

[9] Fernau, H., Heggernes, P., Villanger, Y.: A multi-parameter analysis of hard problems on deterministic finite automata. Journal of Computer and System Sciences 81(4), 747 – 765 (2015)

[10] Gonze, F., Jungers, R.M., Trahtman, A.N.: A note on a recent attempt to improve the pin-frankl bound. CoRR abs/1412.0975 (2014)

[11] Grech, M., Kisielewicz, A.: The Černý conjecture for automata respecting intervals of a directed graph. Discrete Mathematics & Theoretical Computer Science 15(3), 61–72 (2013)

[12] Pin, J.E.: On two combinatorial problems arising from automata theory. Annals of Discrete Mathematics 17, 535–548 (1983)

[13] Roman, A., Drewienkowski, M.: A complete solution to the complexity of synchronizing road coloring for non-binary alphabets. Information and Computation, in print (2015)

[14] Roman, A.: P-NP threshold for synchronizing road coloring. In: Dediu, A.H., Martín-Vide, C. (eds.) Language and Automata Theory and Applications, Lecture Notes in Computer Science, vol. 7183, pp. 480–489. Springer Berlin Heidelberg (2012)

[15] Sandberg, S.: Homing and synchronizing sequences. In: Broy, M., Jonsson, B., Katoen, J.P., Leucker, M., Pretschner, A. (eds.) Model-Based Testing of Reactive Systems, Lecture Notes in Computer Science, vol. 3472, pp. 5–33. Springer Berlin Heidelberg (2005)

[16] Steinberg, B.: The Černý conjecture for one-cluster automata with prime length cycle. Theoret. Comput. Sci. 412(39), 5487 – 5491 (2011)

[17] Trahtman, A.: The road coloring problem. Israel Journal of Mathematics 172(1), 51–60 (2009)

[18] Trahtman, A.: Modifying the upper bound on the length of minimal synchronizing word. In: Owe, O., Steffen, M., Telle, J. (eds.) Fundamentals of Computation Theory, Lecture Notes in Computer Science, vol. 6914, pp. 173–180. Springer Berlin Heidelberg (2011)

[19] Trahtman, A.N.: The road coloring and Černy conjecture. In: Holub, J., Žďárek, J. (eds.) Proceedings of the Prague Stringology Conference 2008. pp. 1–12. Czech Technical University in Prague, Czech Republic (2008)

[20] Travers, N., Crutchfield, J.: Exact synchronization for finite-state sources. Journal of Statistical Physics 145(5), 1181–1201 (2011)

[21] Volkov, M.: Synchronizing automata and the Černý conjecture. In: Martín-Vide, C., Otto, F., Fernau, H. (eds.) Language and Automata Theory and Applications, Lecture Notes in Computer Science, vol. 5196, pp. 11–27. Springer Berlin Heidelberg (2008)

[22] Vorel, V., Roman, A.: Complexity of road coloring with prescribed reset words. In: Dediu, A.H., Formenti, E., Martín-Vide, C., Truthe, B. (eds.) Language and Automata Theory and Applications, pp. 161–172. Lecture Notes in Computer Science, Springer International Publishing (2015)