Subset Synchronization in Monotonic Automata

Andrew Ryzhikov[1,2], Anton Shemyakov[3]

[1] Université Grenoble Alpes, Laboratoire G-SCOP, 38031 Grenoble, France
[2] United Institute of Informatics Problems of NASB, 220012 Minsk, Belarus
[3] Belarusian State University, 220030 Minsk, Belarus
`ryzhikov.andrew@gmail.com, shemyanton@gmail.com`

Abstract

We study extremal and algorithmic questions of subset and careful synchronization in monotonic automata. We show that several synchronization problems that are hard in general automata can be solved in polynomial time in monotonic automata, even without knowing a linear order of the states preserved by the transitions. We provide asymptotically tight bounds on the maximum length of a shortest word synchronizing a subset of states in a monotonic automaton and a shortest word carefully synchronizing a partial monotonic automaton. We provide a complexity framework for dealing with problems for monotonic weakly acyclic automata over a three-letter alphabet, and use it to prove NP-completeness and inapproximability of problems such as FINITE AUTOMATA INTERSECTION and the problem of computing the rank of a subset of states in this class. We also show that checking whether a monotonic partial automaton over a four-letter alphabet is carefully synchronizing is NP-hard. Finally, we give a simple necessary and sufficient condition when a strongly connected digraph with a selected subset of vertices can be transformed into a deterministic automaton where the corresponding subset of states is synchronizing.

1 Introduction

Let $A = (Q, \Sigma, \delta)$ be a deterministic finite automaton (which we further simply call an *automaton*), where Q is the set of its states, Σ is a finite alphabet and $\delta : Q \times \Sigma \to Q$ is a transition function. Note that our definition of automata does not include initial and accepting states. The mapping δ can be inductively extended to the mapping $Q \times \Sigma^* \to Q$, which we also denote as δ: for each word xw, where x is a letter, take $\delta(q, xw) = \delta(\delta(q, x), w)$. An automaton is called *synchronizing* if there exists a word that maps every its state to some fixed state. Such word is also called *synchronizing*. Synchronizing automata play an important role in manufacturing, coding theory and biocomputing, and model systems that can be controlled without knowing their actual state [Vol08].

Synchronizing automata model devices that can be reset, by applying a synchronizing word, to some particular state without having any information about their current state. Automata with a synchronizing set of states model devices that can be reset to a particular state with some partial information about the current state, namely when it is known that the current state belongs to a synchronizing subset of states. A set $S \subseteq Q$ of states of an automaton $A = (Q, \Sigma, \delta)$ is called *synchronizing* if there exists a word $w \in \Sigma^*$ and a state $q \in Q$ such that the word w maps each state $s \in S$ to the state q. The word w is said to *synchronize* the set S. It follows from the definition that an automaton is synchronizing if and only if the set Q of all its states is synchronizing.

In this paper, we deal with monotonic and weakly acyclic automata. An automaton $A = (Q, \Sigma, \delta)$ is called *monotonic* if there is a linear order $\leq$ of its states such that for each $x \in \Sigma$ if $q_1 \leq q_2$ then $\delta(q_1, x) \leq \delta(q_2, x)$. In this case we say that the transitions of the automaton *preserve*, or *respect* this order. Monotonic automata play an important role in the part-orienting process in manufacturing [AV04], some connections of monotonic automata with infinite games are described in [Kop06] and [Kop08]. Once the order $q_1, \ldots, q_n$ of the states is fixed, we denote $[q_i, q_j] = \{q_\ell \mid i \leq \ell \leq j\}$, and $\min S$, $\max S$ as the minimum and maximum states of $S \subseteq Q$ with respect to the order. The following open problem is mentioned in [Shc06], showing that monotonic automata are not fully understood, and require more investigation.

Question 1. *Find a combinatorial characterization (for example, using regular expressions) of languages recognized by monotonic automata.*

An automaton $A = (Q, \Sigma, \delta)$ is called *weakly acyclic* if there exists an order of its states $q_1, \ldots, q_n$ such that if $\delta(q_i, x) = q_j$ for some $x \in \Sigma$, then $i \leq j$. Note that a monotonic automaton does not have to be weakly acyclic, and vice versa. Both weakly acyclic and monotonic automata present proper subclasses of a widely studied class of aperiodic automata [Vol08]. An automaton is called *orientable*, if there exists a cyclic order of its states that is preserved by all transitions of the automaton (see [Vol08] for the discussion of this definition). Each monotonic automaton is obviously orientable. An automaton is called *strongly connected* if any its state can be mapped to any other state by some word.

The two fundamental directions is studying synchronization of automata are extremal (bounding the length of a shortest synchronizing word) and algorithmic (exploring the complexity of deciding synchronizability and finding a shortest synchronizing word) questions.

From the extremal point of view, it is known that any synchronizing n-state automaton can be synchronized by a word of length at most $\frac{n^3-n}{6}$ [Pin83], and the famous Černý conjecture states that the length of such word is at most $(n-1)^2$ [Vol08]. A slightly better but still cubic bound is reported in [Szy17]. For words synchronizing a subset of states, the situation is quite different. It is known that the length of a shortest word synchronizing a subset of states in a binary strongly connected automaton can be exponential in the number of states of the automaton [Vor16]. In weakly acyclic automata, there is a quadratic upper bound on the length of such words [Ryz17]. For orientable n-state automata, a tight $(n-1)^2$ upper bound on the length of a shortest word synchronizing a subset of states is known [Epp90].

Checking whether an automaton is synchronizing can be performed in polynomial time [Vol08], but checking whether a given subset of states in an automaton is synchronizing (the Sync Set problem) is a PSPACE-complete problem in binary strongly connected automata [Vor16], and a NP-complete problem in binary weakly acyclic automata [Ryz17].

Eppstein [Epp90] provides a polynomial algorithm for the Sync Set problem, as well as for some other problems, in orientable automata. However, the proposed algorithms assume that a cyclic order of the states preserved by the transitions is known. Since the problems of recognizing monotonic and orientable automata are NP-complete [Szy15], a linear or cyclic order preserved by the transitions of an automaton cannot be computed in polynomial time unless P = NP. Thus, we should avoid using these orders explicitly in algorithms, so we have to investigate other structural properties of monotonic automata. As shown in this paper, several synchronization problems are still solvable in polynomial time in monotonic automata without knowing an order of states preserved by the transitions.

Approximating the length of a shortest word synchronizing a n-state automaton within a factor of $O(n^{1-\epsilon})$ for any $\epsilon > 0$ in polynomial time is impossible unless P = NP [GS15]. For finding the length of a shortest word synchronizing a subset of states in binary weakly acyclic automata a similar inapproximability bound holds [Ryz17].

A problem closely connected to subset synchronization is careful synchronization of partial automata. A *partial automaton* A is a triple (Q, Σ, δ), where Q and Σ are the same as in the definition of a finite deterministic automaton, and δ is a partial transition function (i.e., a transition function which may be undefined for some argument values). A word w is said to *carefully synchronize* a partial automaton A if it maps all its states to the same state, and each mapping corresponding to a prefix of w is defined for each state. The automaton A is then called *carefully synchronizing*.

The length of a shortest word carefully synchronizing a n-state partial automaton is also a subject of research. Rystsov [Rys80], Martyugin [Mar10b], Vorel [Vor16] and de Bondt et al. [dBDZ17] propose consecutive improvements of (exponential) lower bounds for this value, both in the case of constant and non-constant alphabets. Rystsov [Rys80] provides an upper bound of $O(3^{\frac{n}{3}})$ on this value. A simple relation between careful synchronization and subset synchronization is provided by Lemma 1 of [Vor16].

Deciding whether a partial automaton is carefully synchronizing is PSPACE-complete for binary partial automata [Mar10a], and moreover for binary strongly connected partial automata [Vor16]. It is also NP-hard for aperiodic partial automata over a three-letter alphabet [Ryz17].

A synchronizing set of states can be considered as a set compressible to one element. A more general case of a set compressible to a set of size r is defined by the notion of the rank of a subset. Given an automaton $A = (Q, \Sigma, \delta)$, the *rank* of a word $w \in \Sigma^*$ with respect to a set $S \subseteq Q$ is the size of the image of S under the mapping defined by w in A, i.e., the number $|\{\delta(s, w) \mid s \in S\}|$. The *rank* of an automaton (respectively, of a subset of states) is the minimum among the ranks of all words $w \in \Sigma^*$ with respect to the whole set Q of states of the automaton (respectively, to the subset of states). It follows from the definition that a set of states has rank 1 if and only if it is synchronizing. A state in an automaton is a *sink state* if all letters map this state to itself.

For n-state monotonic automata of rank at most r, Ananichev and Volkov [AV04] show an upper bound of $n-r$ on the length of a shortest word of rank at most r, and also provide bounds on the length of a shortest word of interval rank at most r. Shcherbak [Shc06] continues the investigation of words of bounded interval rank in monotonic automata. Ananichev [Ana05] provides bounds on the length of a shortest word of rank 0 in partial monotonic automata of rank 0 (a partial automaton is called *monotonic* if there exists a linear order of its states preserved by all defined transitions).

In this paper, we study both extremal and algorithmic questions of subset synchronization in monotonic automata. In Section 2, we provide structural results about synchronizing sets of states in monotonic automata and give algorithmic consequences of this results. In Section 3, we provide lower and upper bounds on the maximum length of shortest words synchronizing a subset of states in monotonic automata, and show some lower bounds for related problems. In Section 4, we provide NP-hardness and inapproximability of several problems related to subset synchronization and careful synchronization of monotonic automata. In Section 5 we give necessary and sufficient conditions when a strongly connected digraph can be colored resulting in an automaton with a pre-defined synchronizing set.

A conference version of this paper was published in [RS17]. Besides presenting new results, this paper corrects errors of the conference version.

2 Structure of Synchronizing Sets

Let A be an automaton, and S be a subset of its states. In general, if any two states in S can be synchronized (i.e., form a synchronizing set), S does not necessarily have to be synchronizing, as it is shown by the following theorem.

Theorem 1. *For any positive integer k_0, there exists a binary weakly acyclic automaton A and a subset S of its states such that $|S| \geq k_0$, each pair of states in S in synchronizing, but the rank of S equals $|S| - 1$.*

Proof. Consider the following automaton $A = (Q, \{0, 1\}, \delta)$. Let $S = \{s_0, \dots, s_{k-1}\}$. Let $k = 2^\ell$ for an integer number ℓ, and let $\text{bin}(i)$ be a word which is equal to the binary representation of i of length ℓ (possibly with zeros at the beginning). We introduce new states t_i, p_i for $0 \leq i \leq k - 1$, a state f, and new intermediate states in Q as follows. For each s_i, $0 \leq i \leq k-1$, consider a construction sending s_i to f for a word $\text{bin}(i)$, and to t_i by any other word of length ℓ.

For each t_i, consider the same construction sending t_i to f for a word $\text{bin}(i)$, and to p_i otherwise. For each i, define both transitions from p_i as self-loops. Define both transitions from f as self-loops.

In this construction, each word applied after a word of length 2ℓ obviously has no effect. Consider a word w of length 2ℓ, $w = w_1 w_2$, where both w_1 and w_2 have length ℓ. If $w_1 = w_2$, then the image of S under the mapping defined by w has size k. Otherwise, w synchronizes two states s_i and s_j with $\text{bin}(i) = w_1$ and $\text{bin}(j) = w_2$ and maps all other states to different states. Thus, the rank of S equals $k - 1$. □

The size of the whole automaton is $O(|S| \log |S|)$, thus S can be large comparing to the size of the whole set of states in the automaton.

Since the Sync Set problem is PSPACE-complete in strongly connected automata, pairwise synchronization of states in a subset does not imply that this subset is synchronizing for this class of automata unless P = PSPACE. Thus, it is reasonable to ask the following question.

Question 2. *How large can be the rank of a subset of states in a strongly connected automaton such that each pair of states in this subset can be synchronized?*

For the rest of section, fix a monotonic automaton $A = (Q, \Sigma, \delta)$ and an order $q_1, \ldots, q_n$ of its states preserved by all transitions. As shown by the next theorem, the situation in monotonic automata is in some sense opposite to the situation described by Theorem 1.

Theorem 2. *Let $S \subseteq Q$ be a subset of states of A. Then S is synchronizing if and only if any two states in S can be synchronized.*

Proof. Obviously, any subset of a synchronizing set is synchronizing.

In the other direction, if any two states in S can be synchronized, then the minimal state $q_\ell = \min S$ and the maximal state $q_r = \max S$ in S can be synchronized by a word $w \in \Sigma^*$. Let $q = \delta(q_\ell, w) = \delta(q_r, w)$. Then the interval $[q_\ell, q_r] = \{q_\ell, \ldots, q_r\}$ is synchronized by w, because each state of $[q_\ell, q_r]$ is mapped to the interval $[\delta(q_\ell, w), \delta(q_r, w)]$ $= \{q\}$, since A is monotonic. Thus, $S \subseteq [q_\ell, q_r]$ is synchronizing. □

Corollary 1. *The problem of checking whether a given set S is synchronizing can be solved in $O(|Q|^2 \cdot |\Sigma|)$ time and space for monotonic automata.*

Proof. By Theorem 2 it is enough to check that each pair of states in S can be synchronized, which can be done by solving the reachability problem in the subautomaton of the power automaton, built on all 2-element and 1-element subsets of Q [Vol08]. There are $\frac{|Q|(|Q|+1)}{2}$ states in this subautomaton A^2. We need to check that from each state $\{q_i, q_j\}$, $q_i, q_j \in S$, in A^2 some singleton set is reachable. Consider the underlying digraph of A^2 and reverse all arcs in it. Then we need to check that in this new digraph each vertex $\{q_i, q_j\}$, $q_i, q_j \in S$, is reachable from some singleton. To check it, run breadth-first search simultaneously from all singletons [CLRS09].

To construct the subautomaton we need $O(|Q|^2 \cdot |\Sigma|)$ time and space, and breadth-first search requires time and space linear in the number of arcs of the digraph. □

Corollary 2. *A shortest word synchronizing a given subset S of states can be found in $O(|Q|^4 \cdot |\Sigma|)$ time and $O(|Q|^2 \cdot |\Sigma|)$ space for monotonic automata.*

Proof. Consider the following algorithm. For each pair of states, find a shortest word synchronizing this pair. This can be done by solving the shortest path problem in the subautomaton of the power automaton, build on all 2-element and 1-element subsets of Q [Vol08]. Let W be the set of all such words that synchronize S. Output the shortest word in W.

By an argument similar to the proof of Theorem 2, any shortest word synchronizing $\{\min S, \max S\}$ is a shortest word synchronizing S, thus the algorithm finds a shortest word synchronizing S. Since finding a shortest synchronizing word for a pair of states requires $O(|Q|^2 \cdot |\Sigma|)$ time and there are $O(|Q|^2)$ such words, the set W can be found in $O(|Q|^4 \cdot |\Sigma|)$ time. Finding a shortest word in W synchronizing S requres additional $O(|Q|^4)$ time, since we have to check each word. Thus, the total time required by the algorithm is $O(|Q|^4 \cdot |\Sigma|)$.

The algorithm requires $O(|Q|^2 \cdot |\Sigma|)$ space for the subautomaton construction. We don't have to store W, since we can check the words in W one by one and store only the shortest one, so we need $O(|Q|^2 \cdot |\Sigma|)$ space. □

Corollary 3. *A synchronizing subset of states of maximum size can be found in $O(|Q|^4 \cdot |\Sigma| + |Q|^5)$ time and $O(|Q|^2 \cdot |\Sigma|)$ space in monotonic automata.*

Proof. For each synchronizing pair q_i, q_j of states, find a word synchronizing this pair (in the same way as described in Corollary 1), and the find all states that are mapped by this word to the same state as q_i and q_j. Output the pair with the largest synchronizing set constructed in such a way.

To prove that this algorithm is correct, observe that each word synchronizing a pair q_i, q_j of states synchronizes also all the states q_k, $q_i < q_k < q_j$. On the other hand, if $q_i = \min S$ and $q_j = \max S$ for a synchronizing set S of maximum size (which is an interval), any word synchronizing q_i and q_j synchronizes only S.

The described algorithm requires $O(|Q|^2 \cdot (|Q|^2 \cdot |\Sigma| + |Q|^3))$ (for each pair of states, we need to find a synchronizing word which requires $O(|Q|^2 \cdot |\Sigma|)$ time, and then apply this word to each state, which requires $O(|Q|^3))$ time. Since we need to store only the set of maximum size, the algorithm requires $O(|Q|^2 \cdot |\Sigma|)$ space. □

The problem of finding a synchronizing subset of states of maximum size in general automata is PSPACE-complete [Ryz17]. Türker and Yenigün [TY15] study a variation of this problem, which is to find a set of states of maximum size that can be mapped by some word to a subset of a given set of states in a given monotonic automaton. They reduce the N-Queens Puzzle problem [BS09] to this problem to prove its NP-hardness. However, their proof is unclear, since in the presented reduction the input has size $O(\log N)$, and the output size is polynomial in N.

The algorithms proposed in this section run polynomial time, but the degrees of these polynomials are quite high. A natural question is to find faster algorithms for the described problems. Another interesting quiestion is whether the results can be generalized to oriented automata.

3 Lower Bounds for Synchronizing Words

The length of a shortest word synchronizing a n-state monotonic automaton is at most $n-1$ [AV04]. In this section we investigate a more general question of bounding the length of a shortest word synchronizing a subset of states in a n-state synchronizing automaton. For a more general class of oriented automata a bound of $(n-2)^2$ is known [Epp90], but for monotonic automata a smaller upper bound can be proved.

Theorem 3. *Let S be a synchronizing set of states in a monotonic n-state automaton A. Then for $n \geq 8$ the length of a shortest word synchronizing S is at most $\frac{(n-2)^2}{4}$.*

Proof. Let $A = (Q, \Sigma, \delta)$, and $\{q_1, \dots, q_n\}$ be an order of the states preserved by all transitions of A. Define $q_\ell = \min S$, $q_r = \max S$. We can assume that S can be mapped only to the states in $[q_\ell, q_r]$. Indeed, assume without loss of generality that q_i, $i < \ell$, is a state such that S can be mapped to q_i, and it is the smallest such state. Then by monotonicity there exists a word mapping q_r to q_i by taking only transitions going to smaller states. This word then synchronizes S and has length at most $n-1 \leq \frac{(n-2)^2}{4}$ for $n \geq 8$.

Now we can assume that S can be mapped only to states in $[q_\ell, q_r]$. This means that S can be mapped to the states q_i, q_j in $[q_\ell, q_r]$, and no state outside $[q_i, q_j]$ is reachable from any state of $[q_i, q_j]$. Indeed, the set of states reachable from both q_ℓ, q_r contains q_i and q_j, and if some state outside $[q_i, q_j]$ is reachable from $[q_i, q_j]$ we can synchronize S to a state outside $[q_i, q_j]$, which contradicts the definition of the interval $[q_i, q_j]$. If both q_ℓ and q_r are mapped to states inside $[q_i, q_j]$, they can be then synchronized by applying a word of length at most $j-i$, for example by applying a word w' composed of only letters mapping the consecutive images of q_j to states with smaller indexes by the same reasoning as below. By our assumptions, q_i is reachable from each state in $[q_i, q_j]$, thus such a word w' exists.

Suppose now that $w = w_1 \dots w_m$ is a shortest word synchronizing S. Consider the sequence of pairs $(t_k, s_k) = (\delta(q_\ell, w_1 \dots w_k), \delta(q_r, w_1 \dots w_k))$, $k = 1, 2, \dots, m$. As w is a shortest word synchronizing S, and synchronization of S is equivalent to synchronization of $\{q_\ell, q_r\}$, no pair appears in this sequence twice, and the only pair with equal components is (s_m, t_m). Because of monotonicity, $t_k \leq s_k$ for each $1 \leq k \leq m$. Thus, the maximum length of w is reached when $q_i = q_j$, since after both images are in $[q_i, q_j]$ the remaining length of a synchronizing word is at most $j-i$. Observe that if q_1 or q_n is in S, S again can be synchronized by a word of length $n-1$. Thus, we can assume that $|S| \leq n-2$, and thus the length of w is at most $(i-1)(n-3-i) \leq \frac{(n-2)^2}{4}$. □

The bound is almost tight for monotonic automata over a three-letter alphabet as shown by the following example.

Theorem 4. *For each $m \geq 1$, there exist a $(2m+3)$-state monotonic automaton A over a three-letter alphabet, which has a subset S of states, such that the length of a shortest word synchronizing S is $m^2 + m$.*

Proof. Consider the following monotonic automaton $A = (Q, \Sigma, \delta)$, $Q = \{q_1, \dots, q_{2m+3}\}$. Let $\Sigma = \{0, 1, 2\}$. Let states q_1, q_{m+2} and q_{2m+3} be sink states. For every state q_i, $2 \leq i \leq m+1$, we set $\delta(q_i, 0) = q_{i+1}$, $\delta(q_i, 1) = q_i$, $\delta(q_i, 2) = q_1$. For every state q_i, $m+4 \leq i \leq 2m+2$, we set $\delta(q_i, 0) = q_{2m+3}$, $\delta(q_i, 1) = q_{i-1}$, $\delta(q_i, 2) = q_i$. Finally we define $\delta(q_{m+3}, 0) = q_{2m+2}$, $\delta(q_{m+3}, 1) = q_{m+3}$, $\delta(q_{m+3}, 2) = q_{m+2}$. See Figure 1 for an illustration of the construction.

All transitions of A respect the order $q_1, \dots, q_{2m+3}$, so A is monotonic. Let us show that the shortest word synchronizing the set $S = \{q_2, q_{2m+2}\}$ is $w = 1^{m-1}(01^{m-1})^m 2$. Let S' be a set of states such that $q_i, q_j \in S'$, $2 \leq i \leq m+1$, $m+3 \leq j \leq 2m+2$. The set S' can be mapped only to q_{m+2}, because A is monotonic. Hence if any state of S' is mapped by a word to q_1 or to q_{2m+3}, then this word cannot synchronize S'.

We start with the set $S = \{q_2, q_{2m+2}\}$. There is only one letter 1 that does not map the state q_2 to q_1 or the state q_{2m+2} to q_{2m+3}, and maps S not to itself. Indeed, 0 maps q_{2m+2} to q_{2m+3} and 2 maps q_2 to q_1. Thus, any shortest synchronizing word can start only

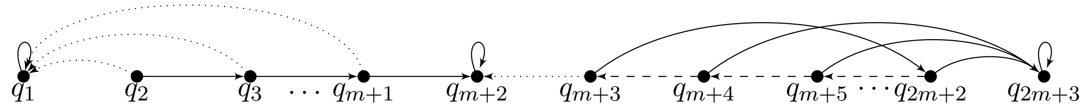

Figure 1: The automaton providing a lower bound for subset synchronization in monotonic automata over a three-letter alphabet. Solid arrows represent transitions for the letter 0, dashed – for the letter 1, dotted – for the letter 2. The states q_1, q_{m+2}, q_{2m+3} are sink states, self-loops are omitted.

with 1. Consider now the set $\{\delta(q_2, 1), \delta(q_{2m+2}, 1)\} = \{q_2, q_{2m+1}\}$. There is only letter 1 that does not map the state q_3 to q_1, or q_{2m+1} to q_{2m+3} and maps this set not to itself. Indeed, 0 maps q_{2m+1} to q_{2m+3} and 2 maps q_3 to q_1. So the second letter of the shortest synchronizing word can only be 1. By a similar reasoning (at each step there is exactly one letter that maps a pair of states not to itself and does not map the states to the sink states q_1 and q_{2m+3}), we deduce that any shortest synchronizing word has to begin with $1^{m-1}(01^{m-1})^m$ and it is easy too see that $1^{m-1}(01^{m-1})^m 2$ synchronizes S. Thus, w is a shortest word synchronizing S, and its length is $m^2 + m$. □

For a n-state automaton, the lower bound on the length of a shortest word in this theorem is $\frac{(n-2)^2-1}{4}$, which is very close to the lower bound $\frac{(n-2)^2}{4}$ from Theorem 3.

By taking q_2 and q_{2m+2} as initial states in two equal copies of the automaton in the proof of Theorem 4, and taking q_{m+2} as the only accepting state in both copies, we obtain the following result.

Corollary 4. *A shortest word accepted by two $(2m+3)$-state monotonic automata which differ only by their initial states can have length $m^2 + m$.*

For binary monotonic automata, our lower bound is slightly smaller, but still quadratic.

Theorem 5. *For each $m \geq 1$, there exist a $(4m+3)$-state binary monotonic automaton A, which has a subset S of states such that the length of a shortest word synchronizing S is at least m^2.*

Proof. Consider the following automaton $A = (Q, \Sigma, \delta)$ with $Q = \{q_1, \ldots, q_{4m+3}\}$, $\Sigma = \{0, 1\}$. Define δ as follows. Set q_1, q_{2m+2}, q_{4m+3} to be sink states. Define $\delta(q_i, 1) = q_{i-1}$ for all $i \neq 1, 2m+2, 4m+3$. For each i, $2 \leq i \leq m+1$, define $\delta(q_i, 0) = q_{i+m}$, and for each i, $m+2 \leq i \leq 2m+1$, define $\delta(q_i, 0) = q_{2m+2}$. For each i, $2m+3 \leq i \leq 3m+3$, define $\delta(q_i, 0) = q_{m+i-1}$, and for each i, $3m+4 \leq i \leq 4m+2$, define $\delta(q_i, 0) = q_{4m+3}$. The defined binary automaton is monotonic, since all its transitions respect the order $q_1, \ldots, q_{4m+3}$. See Figure 2 for an example of the construction.

Define $S = \{q_{m+2}, q_{4m+2}\}$. Let us prove that a shortest word synchronizing S has length at least m^2.

The set S can only be mapped to q_{2m+2}, since it is a sink state between $\min S$ and $\max S$. Thus, no word synchronizing S maps any its state to q_1 or q_{4m+3}. Consider now an interval $[q_i, q_j]$ for $2 \leq i \leq m+1$, $2m+3 \leq j \leq 4m+2$ and note that applying 0 reduces

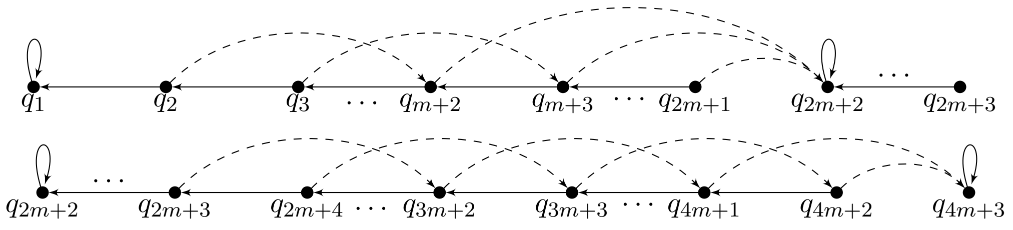

Figure 2: The automaton providing a lower bound for subset synchronization in binary monotonic automata. Dashed arrows represent transitions for the letter 0, solid – for the letter 1. The states q_1, q_{2m+2}, q_{4m+3} are sink states. The picture is divided into two parts because of its width.

its length by 1 (or maps its right end to q_{4m+3}), and applying 1 maps its ends to the ends of another interval of this form with the same length (or maps its left end to q_1). The maximal length of a segment of this form that allows its left end to be mapped to q_{2m+2} is $2m+1$, so before any end of the interval is mapped to q_{2m+2}, the letter 0 has to be applied at least m times. Each application of 0 moves the right end of the intervals $m-1$ states to the right, so each application of 0 requires $m-1$ applications of 1 so that 0 can be applied one more time. Thus, the word mapping S to q_{2m+2} has length at least m^2. Note that S can be synchronized by a word $w = (1^{m-1}0)^m 1^{2m}$ of length $|w| = m^2 + 2m$. $\square$

For a n-state binary monotonic automaton we get a lower bound of $\frac{(n-3)^2}{16}$ from this theorem.

By removing the first and the last state (and leaving all transitions to the removed states undefined) in the automata in the both series, we get the following results.

Corollary 5. *For infinitely many n, there exists a n-state monotonic partial automaton over a three-letter alphabet with shortest carefully synchronizing word of length at least* $\frac{(n-1)^2}{4} + 1$.

Corollary 6. *For infinitely many n, there exist a n-state monotonic binary partial automaton with shortest carefully synchronizing word of length at least* $\frac{(n-1)^2}{16}$.

Question 3. *Find upper bounds on the length of a shortest carefully synchronizing words for monotonic partial automata.*

A related question is to measure the shortest length of a word accepted simultaneously by k monotonic automata. This question is important, for example, for investigation of the computational complexity of the FINITE AUTOMATA INTERSECTION problem, see Section 4 for the details. For alphabet of unbounded size, a partial answer is provided by the following theorem.

Theorem 6. *For any $k > 0$, there exist k 3-state monotonic automata over a k-letter alphabet, such that the length of a shortest word accepted by all automata is at least* $2^k - 1$.

Proof. To prove the theorem, we show how to imitate a simple binary counter with k monotonic automata. Consider the following family $A_i = (Q_i, \Sigma, \delta_i)$, $1 \leq i \leq k$. Here $Q_i = \{f_i, s_i, t_i\}$, $\Sigma = \{a_1, \ldots, a_k\}$ and δ_i are defined as follows. We define $\delta_i(s_i, a_i) = t_i$, and $\delta_i(s_i, a_j) = f_i$, $\delta_i(t_i, a_j) = s_i$ for $i < j$. All yet undefined transitions are self-loops. In each A_i, we take s_i as the initial state, and t_i as the only accepting state.

By induction, the length of a shortest word accepted by all automata is $2^k - 1$, since each next automaton can be mapped to its accepting state only when all previous automata are already in their accepting states, and this operation maps all previous automata to their initinal states. □

Hence, we get a lower bound of $2^{\frac{n}{3}} - 1$, where n is the total number of states in the automata. Thus the most obvious candidate for a certificate to show that Finite Automata Intersection is in NP for monotonic automata fails.

The proof of Theorem 6 implies the following interesting result.

Corollary 7. *For each k there exists a monotonic partial automaton with $3k$ states, alphabet of size $k + 1$ and a shortest carefully synchronizing word of length at least 3^k.*

Proof. Remove the states f_i in all automata from the construction of Theorem 6 and leave all transtions to this states undefined. Then add a letter a such that $\delta(t_i, a) = t_k$ for $1 \leq i \leq k$. Thus we get a carefully synchronizing automaton imitating a binary counter. The last step is to note that $2^{\frac{1}{2}} < 3^{\frac{1}{3}}$, and to imitate a ternary counter in the exactly same way instead. □

We note that the ternary counter is optimal since $2^{\frac{1}{2}} < 3^{\frac{1}{3}}$ and $3^{\frac{1}{3}} > \ell^{\frac{1}{\ell}}$ for $\ell \geq 4$.

The bound in this corollary is of the same order as in the result of Rystsov [Rys80] and Martyugin [Mar10a], but our example is monotonic and has simpler structure. Since each carefully synchronizing partial automaton has a carefully synchronizing word of length $O(3^{\frac{n}{3}})$ [Rys80], the bound is asymptotically tight.

Some additional optimization can be done for the described construction. In particular, the after countng to $3^k - 1$, only two states in the set of reached states can be synchronized by a new letter. Then counting is repeated (to $3^{k-1} - 1$) and again two states are synchronized, and so on. Thus, the bound will become $3^k + 3^{k-1} + \ldots + 3 + 1$. However, the bound remains of the same order and is still smaller than the bounds of Rystsov and Martyugin for general automata.

For a constant number of letters the considered problems remain open.

Question 4. *What is the length of a shortest word accepted simultaneously by k monotonic automata over an alphabet of constant size? What is the length of a shortest word carefully synchronizing a monotonic partial automaton with n states and alphabet of constant size?*

4 Complexity Results

In this section, we obtain computational complexity results for several problems related to subset synchronization in monotonic automata. We improve Eppstein's construction

[Epp90] to make it suitable for monotonic automata. We shall need the following NP-complete SAT problem [Sip12].

SAT
Input: A set X of n boolean variables and a set C of m clauses;
Output: Yes if there exists an assignment of values to the variables in X such that all clauses in C are satisfied, No otherwise.

Provided a set X of boolean variables $x_1, \dots, x_n$ and a clause c_j, construct the following automaton $A_j = (Q, \Sigma, \delta)$. Take

$$Q = \{q_1, \dots, q_{n+1}\} \cup \{q'_2, \dots, q'_n\} \cup \{s, t\}.$$

Let $\Sigma = \{0, 1, r\}$. Define the transition function δ as follows. For each i, $1 \le i \le n$, map a state q_i to q'_{i+1} (or to t if $i = n$) by a letter $x \in \{0, 1\}$ if the assignment $x_i = x$ satisfies c_j, and to q_{i+1} otherwise. For each i, $2 \le i \le n-1$, set $\delta(q'_i, x) = \delta(q'_{i+1}, x)$ for $x \in \{0, 1\}$. Set $\delta(q'_n, x) = t$ for $x \in \{0, 1\}$. Define transitions from t for letters $0, 1$ as self-loops. Finally, define $\delta(q, r) = s$ for $q \in Q \setminus \{t\}$, $\delta(t, r) = t$. See Figure 3 for an example.

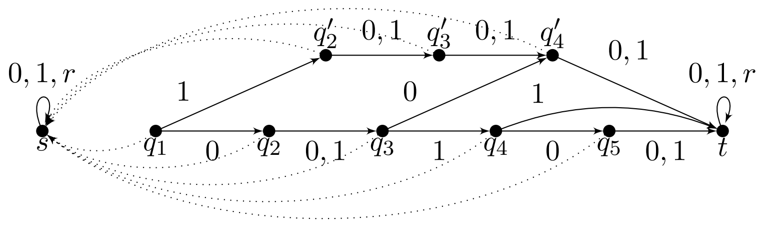

Figure 3: The automaton A_j for a clause $c_j = (x_1 \vee \overline{x}_3 \vee x_4)$. Dotted arrows represent transitions for the letter r.

Note that A_j is monotonic, since it respects the order

$$s, q_1, q_2, q'_2, q_3, q'_3, \dots, q_n, q'_n, q_{n+1}, t.$$

It is also weakly acyclic, since its underlying digraph has no simple cycles of length at least 2.

Also, provided the number of variables n, construct an automaton $T = (Q_T, \Sigma, \delta_T)$ as follows. Take $Q_T = \{a, p_1, \dots, p_{n+1}, b\}$, $\Sigma = \{0, 1, r\}$. Define $\delta(p_i, x) = p_{i+1}$ for each i, $1 \le i \le n$, and $x \in \{0, 1\}$, and $\delta(p_{n+1}, x) = b$ for $x \in \{0, 1\}$. Define also $\delta(a, x) = a$ and $\delta(b, x) = b$ for each $x \in \Sigma$, and $\delta(p_i, r) = a$ for $1 \le i \le n+1$. See Figure 4 for an example. This automaton is monotonic, since it respects the order $a, p_1, \dots, p_{n+1}, b$, and it is obviously weakly acyclic.

First, we prove NP-completeness of the following problem.

Finite Automata Intersection
Input: Automata $A_1, \dots, A_k$ (with initial and accepting states);
Output: Yes if there is a word which is accepted by all automata, No otherwise.

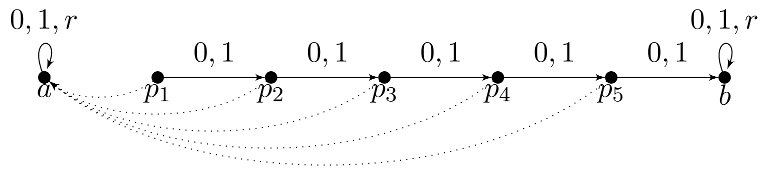

Figure 4: The automaton T for $n = 4$ variables. Dotted arrows represent transitions for the letter r.

This problem is PSPACE-complete for general automata [Koz77], and NP-complete for binary weakly acyclic automata [Ryz17]. Some results on this problem are surveyed in [HK11]. Blondin et al. [BKM16] provide further results on the problem. Some complexity lower bounds are presented in [Weh14] and [FK16].

Theorem 7. *The* Finite Automata Intersection *problem is NP-complete for monotonic weakly acyclic automata over a three-letter alphabet.*

Proof. The fact that the problem is in NP follows from the fact that Finite Automata Intersection for weakly acyclic automata is in NP [Ryz17].

To prove hardness, we reduce the SAT problem. For each clause $c_j \in C$, construct an automaton A_j, and set q_1 as its initial state and t as its only accepting state. Construct also the automaton T with the initial state p_1 and accepting state a.

We claim that C is satisfiable if and only if all automata in $\{A_j \mid c_j \in C\} \cup \{T\}$ accept a common word w. Indeed, assume that there is a common word accepted by all these automata. Then none of the first n letters of this word can be r, otherwise all automata A_j are mapped to s, which is a non-accepting sink state. The next letter has to be r, otherwise T is mapped to b, which is a non-accepting sink state. But that means that in each A_j, the set q_1 is mapped by a n-letter word $z_1 \dots z_n$ to the accepting state t. Thus, by construction, the assignment $x_i = z_i$ satisfies all clauses in C.

By the same reasoning, if the assignment $x_i = z_i$, $1 \le i \le n$, satisfies all clauses in C, then $z_1 \dots z_n r$ is a word accepted by all automata. □

Now we switch to a related Set Rank problem.

Set Rank
Input: An automaton A and a set S of its states;
Output: The rank of S.

This problem is hard to approximate for binary weakly acyclic automata [Ryz17]. To get inapproximability results for monotonic automata, we use the following problem.

Max-3SAT
Input: A set X of n boolean variables and a set C of m 3-term clauses;
Output: The maximum number of clauses that can be simultaneously satisfied by some assignment of values to the variables.

This problem cannot be approximated in polynomial time within a factor of $\frac{7}{8} + \epsilon$ for any $\epsilon > 0$ unless P = NP [Hås01].

Theorem 8. *The* Set Rank *problem cannot be approximated in polynomial time within a factor of* $\frac{9}{8} - \epsilon$ *for any* $\epsilon > 0$ *in monotonic weakly acyclic automata over a three-letter alphabet unless* $P = NP$.

Proof. We reduce the Max-3SAT problem. For each clause $c_j \in C$, construct an automaton A_j. Construct also m copies of the automaton T, denoted T_j, $1 \leq j \leq m$. Define an automaton A with the set of states which is the union of all sets of states of $\{A_j, T_j \mid 1 \leq j \leq m\}$, alphabet Σ and transition functions defined in all constructed automata. For each j, identify the state t in A_j with the state a in T_j. Take S to be the set of states q_1 from each automaton A_j, and p_1 from each T_j. The constructed automaton is monotonic and weakly acyclic.

If h is the minimum number of clauses in C that are not satisfied by an assignment, the set S has rank $m+h$. Indeed, consider an assignment $x_i = z_i$, $1 \leq i \leq n$, not satisfying exactly h clauses in C. Then the word $z_1 \ldots z_n r$ has rank $m + h$ with respect to S.

In the other direction, let w be a word of minimum rank with respect to the set S. If any of the first n letters of w is r, then q_1 in each A_i is mapped to s in the corresponding automaton, and thus w has rank $2m$ with respect to S. The same is true if $(n+1)$st letter of w is not r, because then p_1 in each T_i is mapped to b in the corresponding automaton. If first n letters $z_1, \ldots, z_n$ of w are not r, and the next letter is r, then the assignment $x_i = z_i$ does not satisfy exactly h' clauses, where $m + h'$ is the rank of the word w with respect to S. For the word of minimum rank, we get the required equality.

It is NP-hard to decide between (i) all clauses in C are satisfiable and (ii) at most $(\frac{7}{8} + \epsilon)m$ clauses in C can be satisfied by an assignment [Hås01]. In the case (i), the rank of S is m, in the case (ii) it is at least $m + (\frac{1}{8} - \epsilon)m$. Since it is NP-hard to decide between this two options, we get $(\frac{9}{8} - \epsilon)$-inapproximability for any $\epsilon > 0$. □

By using an argument similar to the proof of Theorem 8, we can show inapproximability of the maximization version of Finite Automata Intersection (where we are asked to find a maximum number of automata accepting a common word). Indeed, take m copies of T together with the set $\{A_j \mid c_j \in C\}$ as the input of Finite Automata Intersection and reduce Max-3SAT to it (input and accepting states are assigned according to the construction in Theorem 7). Then the maximum number of automata accepting a common word is $m + g$, where g is the maximum number of simultaneously satisfied clauses in C, since all copies of T have to accept this word. Thus it is NP-hard to decide between (i) all $2m$ automata accept a common word and (ii) at most $m + (\frac{7}{8} + \epsilon)m$ automata accept a common word, and we get the following result.

Corollary 8. *The maximization version of the* Finite Automata Intersection *problem cannot be approximated in polynomial time within a factor of* $\frac{15}{16} + \epsilon$ *for any* $\epsilon > 0$ *in monotonic weakly acyclic automata over a three-letter alphabet unless* $P = NP$.

Consider now the following problem briefly discussed in the introduction.

Careful Synchronization
Input: A partial automaton A;
Output: Yes if A is carefully synchronizing, No otherwise.

Theorem 9. *The* Careful Synchronization *problem is NP-hard for monotonic automata over a four-letter alphabet.*

Proof. We reduce the SAT problem. Let A'_j be A_j with alphabet restricted to $\{0, 1\}$ and without the state s, and T' be T with alphabet restricted to $\{0, 1\}$ and without the states a, b. Let $\Sigma' = \{0, 1, y, z\}$. Provided X and C, construct A'_j for each clause c_j, and also construct T'. Let Q' be the union of all states of each A'_j, all states of T' and a new state f. We expand already defined (for the letters $0, 1$) transition function δ as follows. Define y to map all states in each A'_j to q_1 in this gadget, and all states in T' to p_1. Finally, define z to map the state t in each A_j, the state p_{n+1} in T and the state f to f. Leave all other transition undefined. We denote thus obtained automaton as $A' = (Q', \Sigma', \delta')$.

Any word w carefully synchronizing A' begins with y, since it is the only letter defined for all states. After applying it, the set S of reached states consists of q_1 in each A_j, p_1 in T, and f. To reach f from this set, we need to apply z at least once. Right before applying z, the set of reached states must consist of exactly n letters of the set $\{0, 1\}$, because application of y takes us back to S, and applying $0, 1$ more than n times is not defined for the state p_1. Thus, in each A_j the state q_1 must be mapped to t by n 0s and 1s which is possible if and only if C is satisfiable. □

Question 5. *What is the complexity of the mentioned problems for binary monotonic automata? Do the problems discussed in this section belong to NP for monotonic automata?*

We note that it does not matter in the provided reductions whether a linear order preserved by all transitions is known or not.

5 Subset Road Coloring

The famous Road Coloring problem is formulated as follows. Given a strongly connected digraph with all vertices of equal out-degree k, is it possible to find a coloring of its arcs with letters of alphabet Σ, $|\Sigma| = k$, resulting in a synchronizing deterministic automaton. This problem was stated in 1977 by Adler, Goodwyn and Weiss [AGW77] and solved in 2007 by Trahtman [Tra09]. A natural generalization of this problem is to find a coloring of a strongly connected digraph turning it into a deterministic automaton where a given subset of states is synchronizing. We introduce the problem formally and show that its solution is a corollary of a result of Béal and Perrin [BP14]. In particular, the problem of deciding whether such a coloring exists is solvable in polynomial time.

Let $G = (V, E)$ be a strongly connected digraph such that each its vertex has out-degree k. A *coloring* of G with letters from alphabet Σ, $|\Sigma| = k$, is a function assigning each arc of G a letter from Σ, such that for each vertex, each pair of arcs outgoing from it achieves different letters. We say that a coloring *synchronizes* $S \subseteq V$ in G if S is a synchronizing set in the resulting automaton.

If the greatest common divisor of the lengths of all cycles of G is ℓ, the set V can be partitioned into sets $V_1, \ldots, V_\ell$ in such a way that if (v, u) is an arc of G, then $v \in V_i, u \in V_{i+1}$ or $v \in V_\ell, u \in V_1$ [Fri90]. Moreover, such partition is unique.

Theorem 10. *A strongly connected digraph G with vertices of equal out-degree has a coloring synchronizing a set $S \subseteq V$ if and only if $S \subseteq V_i$ for some i.*

Proof. Obviously, if two vertices of S belong to distinct sets V_i and V_j, S can not be synchronized. Assume that $S \subseteq V_i$ for some i. As proved in [BP14], there exists a coloring of G such that the resulting automaton A has rank ℓ. In this coloring each V_j, $1 \leq j \leq \ell$, is a synchronizing set, since no two states from two different sets V_p, V_t, $p \neq t$, can be synchronized and A has rank ℓ. Hence, $S \subseteq V_i$ is also a synchronizing set. □

According to this theorem, checking whether there exists such a coloring can be performed in polynomial time. This coloring can be constructed in polynomial time using the algorithm from [BP14].

Acknowledgments

We thank Ilia Fridman for useful comments on the presentation of this paper, and anonymous reviewers for multiple useful comments and suggestions that improved the content and presentation of the preliminary version of this paper.

References

[AGW77] Roy L. Adler, L. Wayne Goodwyn, and Benjamin Weiss. Equivalence of topological markov shifts. *Israel J. Math*, 27(1):49–63, 1977.

[Ana05] Dmitry S. Ananichev. The mortality threshold for partially monotonic automata. In Clelia De Felice and Antonio Restivo, editors, *DLT 2005. LNCS, vol. 3572*, pages 112–121. Springer, Berlin, Heidelberg, 2005.

[AV04] Dimitry S. Ananichev and Mikhail V. Volkov. Synchronizing monotonic automata. *Theor. Comput. Sci.*, 327(3):225–239, 2004.

[BKM16] Michael Blondin, Andreas Krebs, and Pierre McKenzie. The complexity of intersecting finite automata having few final states. *computational complexity*, 25(4):775–814, 2016.

[BP14] Marie-Pierre Béal and Dominique Perrin. A quadratic algorithm for road coloring. *Discrete Appl. Math*, 169:15–29, 2014.

[BS09] Jordan Bell and Brett Stevens. A survey of known results and research areas for N-queens. *Discrete Math.*, 309(1):1 – 31, 2009.

[CLRS09] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms.* MIT Press, 3rd edition, 2009.

[dBDZ17] Michiel de Bondt, Henk Don, and Hans Zantema. DFAs and PFAs with long shortest synchronizing word length. In Émilie Charlier, Julien Leroy, and Michel Rigo, editors, *DLT 2017, Liège, Belgium, August 7-11, 2017, Proceedings, LNCS vol. 10396*, pages 122–133, Cham, 2017. Springer International Publishing.

[Epp90] David Eppstein. Reset sequences for monotonic automata. *SIAM J. Comput*, 19(3):500–510, 1990.

[FK16] Henning Fernau and Andreas Krebs. *Problems on Finite Automata and the Exponential Time Hypothesis*, pages 89–100. Springer International Publishing, Cham, 2016.

[Fri90] Joel Friedman. On the road coloring problem. *Proc. Amer. Math. Soc*, 110:1133–1135, 1990.

[GS15] Paweł Gawrychowski and Damian Straszak. Strong inapproximability of the shortest reset word. In F. Giuseppe Italiano, Giovanni Pighizzini, and T. Donald Sannella, editors, *MFCS 2015. LNCS, vol. 9234*, pages 243–255. Springer, Heidelberg, 2015.

[Hås01] Johan Håstad. Some optimal inapproximability results. *J. ACM*, 48(4):798–859, 2001.

[HK11] Markus Holzer and Martin Kutrib. Descriptional and computational complexity of finite automata – a survey. *Information and Computation*, 209(3):456–470, 2011.

[Kop06] Eryk Kopczyński. Half-positional determinacy of infinite games. In Michele Bugliesi, Bart Preneel, Vladimiro Sassone, and Ingo Wegener, editors, *ICALP 2006. LNCS, vol. 4052*, pages 336–347. Springer, Berlin, Heidelberg, 2006.

[Kop08] Eryk Kopczyński. *Half-Positional Determinacy of Infinite Games.* PhD thesis, Warsaw University, Poland, 2008.

[Koz77] Dexter Kozen. Lower bounds for natural proof systems. In *Proceedings of the 18th Annual Symposium on Foundations of Computer Science*, pages 254–266. IEEE Computer Society, 1977.

[Mar10a] Pavel V. Martyugin. Complexity of problems concerning carefully synchronizing words for PFA and directing words for NFA. In Farid Ablayev and Ernst W. Mayr, editors, *CSR 2010. LNCS, vol. 6072*, pages 288–302. Springer, Heidelberg, 2010.

[Mar10b] Pavel V. Martyugin. A lower bound for the length of the shortest carefully synchronizing words. *Russian Mathematics*, 54(1):46–54, 2010.

[Pin83] Jean-Éric Pin. On two combinatorial problems arising from automata theory. *Ann. Discrete Math.*, 17:535–548, 1983.

[RS17] Andrew Ryzhikov and Anton Shemyakov. Subset synchronization in monotonic automata. In Juhani Karhumäki and Aleksi Saarela, editors, *Proceedings of the Fourth Russian Finnish Symposium on Discrete Mathematics, TUCS Lecture Notes 26*, pages 154–164. 2017.

[Rys80] Igor K. Rystsov. Asymptotic estimate of the length of a diagnostic word for a finite automaton. *Cybernetics*, 16(2):194–198, 1980.

[Ryz17] Andrew Ryzhikov. Synchronization problems in automata without non-trivial cycles. In Arnaud Carayol and Cyril Nicaud, editors, *CIAA 2017, LNCS, vol. 10329*, pages 188–200. Springer, Cham, 2017.

[Shc06] Tamara Shcherbak. The interval rank of monotonic automata. In Jacques Farré, Igor Litovsky, and Sylvain Schmitz, editors, *CIAA 2005. LNCS, vol. 3845*, pages 273–281. Springer, Berlin, Heidelberg, 2006.

[Sip12] Michael Sipser. *Introduction to the Theory of Computation.* Cengage Learning, 3rd edition, 2012.

[Szy15] Marek Szykuła. Checking whether an automaton is monotonic is NP-complete. In Frank Drewes, editor, *CIAA 2015. LNCS, vol. 9223*, pages 279–291. Springer, Cham, 2015.

[Szy17] Marek Szykuła. Improving the upper bound the length of the shortest reset words. *CoRR*, abs/1702.05455, 2017.

[Tra09] Avraham N. Trahtman. The road coloring problem. *Israel J. Math*, 172(1):51–60, 2009.

[TY15] Uraz Cengiz Türker and Hüsnü Yenigün. Complexities of some problems related to synchronizing, non-synchronizing and monotonic automata. *Int. J. Found. Comput. Sci.*, 26(01):99–121, 2015.

[Vol08] Mikhail V. Volkov. Synchronizing automata and the Černý conjecture. In Carlos Martín-Vide, Friedrich Otto, and Henning Fernau, editors, *LATA 2008. LNCS, vol. 5196*, pages 11–27. Springer, Heidelberg, 2008.

[Vor16] Vojtěch Vorel. Subset synchronization and careful synchronization of binary finite automata. *Int. J. Found. Comput. Sci.*, 27(5):557–578, 2016.

[Weh14] Michael Wehar. *Hardness Results for Intersection Non-Emptiness*, pages 354–362. Springer Berlin Heidelberg, Berlin, Heidelberg, 2014.