Theoretical Computer Science 450 (2012) 3–9

Contents lists available at SciVerse ScienceDirect

Theoretical Computer Science

journal homepage: www.elsevier.com/locate/tcs

Complexity of problems concerning reset words for cyclic and Eulerian automata

Pavel Martyugin

Ural Federal University, 620083 Ekaterinburg, Russia

ARTICLE INFO

Keywords:
Synchronization
Automata
Reset words
Computational complexity

ABSTRACT

A word is called a reset word for a deterministic finite automaton if it maps all states of this automaton to one state. We consider two classes of automata: cyclic automata and Eulerian automata. For these classes we study the computational complexity of the following problems: does there exist a reset word of given length for a given automaton? what is the minimal length of the reset words for a given automaton?

1. Introduction

A *deterministic finite automaton* (DFA) $\mathscr{A}$ is a triple $\langle Q, \Sigma, \delta\rangle$, where Q is a finite set of *states*, Σ is a finite *alphabet*, and $\delta : Q \times \Sigma \to Q$ is a totally defined *transition function*. The function δ extends in a unique way to an action $Q \times \Sigma^* \to Q$ of the free monoid Σ^* over Σ; this extension is also denoted by δ. We denote $\delta(q, w)$ by $q \cdot w$. For $S \subseteq Q$, $w \in \Sigma^*$, we also define $S \cdot w = \{q \cdot w \mid q \in S\}$.

A DFA $\mathscr{A}$ is called *synchronizing* if there exists a word $w \in \Sigma^*$ whose action synchronizes $\mathscr{A}$, that is, leaves the automaton in one particular state no matter at which state in Q it started: $\delta(q, w) = \delta(q', w)$ for all $q, q' \in Q$. Any word w with this property is said to be a *reset word* for the automaton.

Černý in [4] produced for each integer n a synchronizing automaton with n states, 2 input letters and the shortest reset word has length $(n-1)^2$. He conjectured that these automata represent the worst possible case, that is, every synchronizing automaton with n states can be reset by a word of length $(n-1)^2$. The conjecture is arguably the most longstanding open problem in the combinatorial theory of finite automata. Upper bounds within the confines of the Černý conjecture have been obtained for the maximum length of the shortest reset words for synchronizing automata in some special classes, see, e.g., [6,1,7,5,12]. Two of these classes are considered in the present paper. In the general case there is only a cubic upper bound $(n^3 - n)/6$, see [10].

It is natural to consider computational complexity of various problems arising from the study of automata synchronization. The most important questions are: is a given automaton synchronizing or not, and what is the minimal length of the reset words for a given automaton?

It follows from [4] that there exists an algorithm that checks whether a given DFA $\mathscr{A} = \langle Q, \Sigma, \delta\rangle$ is synchronizing. This algorithm works within $O(|\Sigma| \cdot |Q|^2)$ time bound. In [6], Eppstein presented another algorithm which works within $O(|\Sigma| \cdot |Q|^2) + O(|Q|^3)$ time bound and finds some reset word (which need not to be the shortest reset word for $\mathscr{A}$). In [6,11] it was shown that the following problem SYN is NP-complete: given a DFA $\mathscr{A}$ and a positive integer L, is there a synchronizing word of length at most L. This problem remains NP-complete even if restricted to automata on a 2-letter alphabet. Moreover, Berlinkov in [3] proved that no polynomial time algorithm approximates the length of the shortest synchronizing word within a constant factor for a given DFA.

In [9], Olschewski and Ummels considered a problem MIN-SYN: given a DFA $\mathscr{A}$ and a positive integer L, is the minimum length of reset words for the automaton $\mathscr{A}$ equal to L? They proved that this problem is DP-complete, where DP is a class of

E-mail address: martugin@mail.ru.

0304-3975/$ – see front matter © 2012 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2012.04.022

all languages of the form $L = L_1 \setminus L_2$ with $L_1, L_2 \in$NP. The canonical DP-complete problem is SAT-UNSAT: given two Boolean formulae ϕ and ψ (in CNF), the problem is to decide whether ϕ is satisfiable and ψ is unsatisfiable. The problem MIN-SYN remains DP-complete even for 2-letter automata.

Since the problems SYN and MIN-SYN turn out to be computationally difficult in general, it is reasonable to consider their restrictions to some natural classes of automata. For any class C of automata, we define the restricted versions SYN(C) and MIN-SYN(C) of SYN and respectively MIN-SYN.

Instance: A DFA $\mathscr{A} \in C$ and an integer $L > 0$.
Question of SYN(C): Is there a reset word of length L for the automaton $\mathscr{A}$?
Question of MIN-SYN(C): Is the minimum length of reset words for the automaton $\mathscr{A}$ equal to L?

These problems have been considered in the literature for cyclically monotonic (see [6]), monotonic, commutative, aperiodic, $\mathscr{D}$-trivial automata, for automata with simple idempotents and for automata with a zero state (see [8]). In some cases they become solvable in polynomial time, in some other cases they remain computationally hard. In the present paper we consider these problems for two further natural classes of automata: the class *CYCLE* of cyclic automata and the class *EULER* of Eulerian automata. Let us define these classes and comment on their synchronization properties.

Let $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ be a DFA and $|Q| = n$. The letter $b \in \Sigma$ is said to be *cyclic* if it acts on the set Q as a cyclic permutation of order n. This means that for any $q \in Q$ and $i \in \{1, \dots, n-1\}$, we have $\delta(q, b^n) = q \neq \delta(q, b^i)$. A DFA with a cyclic letter is called *cyclic*. Dubuc [5] has proved that every n-state synchronizing cyclic DFA has a reset word of length $(n-1)^2$, thus, the Černý conjecture holds true for cyclic automata. Furthermore, this upper bound of the length of the shortest reset words is tight, because automata from the Černý series [4] are cyclic.

A DFA $\mathscr{A} = \langle Q, \Sigma, \delta \rangle$ is said to be *Eulerian* if its underlying digraph is Eulerian. It is well-known that the underlying digraph of $\mathscr{A}$ is Eulerian if and only if for every state $q \in Q$ there are exactly $|\Sigma|$ pairs $(p, a) \in Q \times \Sigma$ such that $p \cdot a = q$. For any n-state synchronizing Eulerian DFA there exists a reset word of length $n^2 - 3n + 3$ (see [2,7]). It means that the Černý conjecture is true for Eulerian automata.

For a class C of automata and a positive integer k, we denote by C_k the class of all automata in C with k input letters. Here we prove that each of the problems SYN(*CYCLE*), SYN($CYCLE_k$) with $k \geq 2$, SYN(*EULER*), SYN($EULER_k$) with $k \geq 3$ is NP-complete, and each of the problems MIN-SYN(*CYCLE*), MIN-SYN($CYCLE_k$) with $k \geq 2$, MIN-SYN(*EULER*), MIN-SYN($EULER_k$) with $k \geq 3$ is both NP-hard and co-NP-hard. The question about the complexity of the problems SYN($EULER_2$) and MIN-SYN($EULER_2$) remains open.

For the sequel, we need some notation. For a set Q, let $|Q|$ denote the cardinality of Q and let 2^Q stands for the set of all subsets of Q. For a word $w \in \Sigma^*$, we denote by $|w|$ the length of w and by $w[i]$, where $1 \leq i \leq |w|$, the i-th letter in w from the left. If $1 \leq i \leq j \leq |w|$, we denote by $w[i, j]$ the word $w[i] \cdots w[j]$.

2. Cyclic automata

Theorem 1. *The problem SYN(CYCLE$_2$) is NP-complete.*

Proof. It is easy to see that the general problem SYN belongs to NP, because any synchronizing automaton can be synchronizing by a word of polynomial length (of length at most $(n^3 - n)/6$, see [10]). Now we reduce the problem SAT to SYN($CYCLE_2$). Take an instance of SAT consisting of the clauses $c_1(x_1, \dots, x_n), \dots, c_p(x_1, \dots, x_n)$ over the Boolean variables $x_1, \dots, x_n \in \{0, 1\}$. We may (and will) assume that no clause contains both x_m and $\neg x_m$ for any $m \in \{1, \dots, n\}$. We are going to construct a 2-letter automaton $\mathscr{A}_{cycle} = \langle Q, \Sigma, \delta \rangle$ and a number L such that there exists a reset word of length L for $\mathscr{A}_{cycle}$ if and only if $c_1 \wedge c_2 \wedge \cdots \wedge c_p$ is satisfiable.

Let $G = \{(i, m) \mid c_i \text{ contains } \neg x_m\}$. We put

$$\Sigma = \{a, b\}, \qquad Q = \left(\bigcup_{i=1}^{p} Q_i\right) \cup \left(\bigcup_{i=1}^{p} D_i\right) \cup \left(\bigcup_{(i,m)\in G} S_i^m\right), \quad \text{where}$$
$$Q_i = \{q(i, 0), \dots, q(i, n+2)\}, \qquad D_i = \{d(i, 1), \dots, d(i, n+4)\},$$
$$S_i^m = \{s(i, m, 1), \dots, s(i, m, n+4)\}.$$

Now we define the action of the letters a and b. For all $i \in \{1, \dots, p\}$ and $m \in \{1, \dots, n\}$, we put

$$q(i, 0) \cdot a = q(i, 1); \; q(i, 0) \cdot b = \begin{cases} q(i-1, 0) & \text{if } i > 1, \\ q(1, 1) & \text{if } i = 1; \end{cases}$$

$$q(i, m) \cdot b = \begin{cases} s(i, m, 1) & \text{if } \neg x_m \text{ occurs in } c_i, \\ q(i, m+1) & \text{otherwise;} \end{cases}$$

$$q(i, m) \cdot a = \begin{cases} d(1, 1) & \text{if } x_m \text{ occurs in } c_i, \\ q(i, m+1) & \text{otherwise;} \end{cases}$$

$$q(i, n+1) \cdot a = q(i, n+1) \cdot b = q(i, n+2);$$
$$q(i, n+2) \cdot a = d(1, 1); \; q(i, n+2) \cdot b = d(i, 1).$$

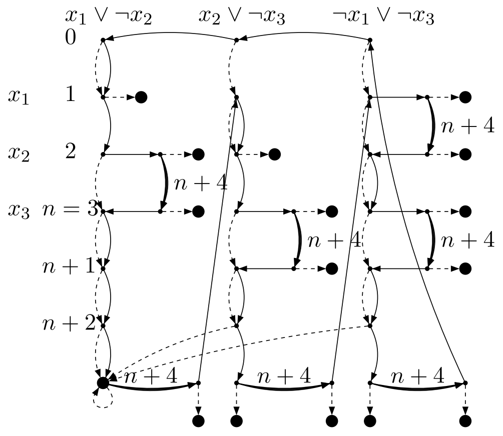

Fig. 1. The automaton $\mathscr{A}_{cycle}$ for the clauses $x_1 \vee \neg x_2, x_2 \vee \neg x_3, \neg x_1 \vee \neg x_3$.

For all $i \in \{1, \dots, p\}, m \in \{1, \dots, n\}$ such that $(i, m) \in G$, and for all $j \in \{1, \dots, n+4\}$, we put

$$s(i, m, j) \cdot a = d(1, 1);\ s(i, m, j) \cdot b = \begin{cases} s(i, m, j+1) & \text{if } j < n+4, \\ q(i, m+1) & \text{if } j = n+4. \end{cases}$$

For all $i \in \{1, \dots, p\}$ and $j \in \{1, \dots, n+4\}$, we put

$$d(i, j) \cdot a = d(1, 1);\ d(i, j) \cdot b = \begin{cases} d(i, j+1) & \text{if } j < n+4, \\ q(i+1, 1) & \text{if } j = n+4, i < p, \\ q(p, 0) & \text{if } j = n+4, i = p. \end{cases}$$

We put $L = n + 2$. An example of the automaton $\mathscr{A}_{cycle}$ is presented in Fig. 1. The action of the letter b is shown with solid lines. The action of the letter a is shown with dashed lines. All large black circles represent the same state $d(1, 1)$ (this way we try to improve the readability of the picture). Every bold arrow labeled by $n + 4$ represents one of the sets S_i^m or D_i. Every set D_i, $i \in \{1, \dots, p\}$, and every set S_i^m, $(i, m) \in G$, contains $n + 4$ states. Fig. 1 contains three columns of states. The i-th column contains the states $q(0, i), \dots, q(n + 2, i)$ for fixed i. Every horizontal row contains the states $q(m, 1), q(m, 2), q(m, 3)$ for fixed m. There are some right arrows labeled by b from columns to the sets S_i^m. The set D_i is drawn under the corresponding set Q_i.

The size of the automaton $\mathscr{A}_{cycle}$ is a polynomial function of the size of the clauses $c_1, \dots, c_p$. It easy to prove that the automaton $\mathscr{A}_{cycle}$ is cyclic, namely, the letter b acts on the set Q as a cyclic permutation of order $|Q|$.

We notice that the word a^{n+3} is a reset word for $\mathscr{A}_{cycle}$. We will prove that there is a reset word of length less than $n + 3$ if and only if the formula $c_1 \wedge c_2 \wedge \dots \wedge c_p$ is satisfiable.

Lemma 1. *If $q \in Q$, $w \in \Sigma^*$, $w[n + 2] = a$ and there is an integer s such that $s \in \{1, \dots, n + 1\}$ and $q \cdot w[1, s] = d(1, 1)$, then $q \cdot w[1, n + 2] = d(1, 1)$.*

Proof. We have

$$q \cdot w[1, n+1] = q \cdot w[1, s]w[s+1, n+1] = d(1, 1) \cdot w[s+1, n+1] \in D_1.$$

Therefore, $q \cdot w[1, n + 2] = d(1, 1)$. □

Lemma 2. *If $w \in \Sigma^*$, $|w| = n + 2$ and $w[1] = w[n + 2] = a$, then*

$$\delta(Q \backslash \{q(1, 0), \dots, q(p, 0)\}, w) = \{d(1, 1)\}.$$

Proof. The letter a maps all sets D_i and S_i^m to the state $d(1, 1)$. Therefore, if $q \in (\bigcup_{(i,j)\in G} S_i^j) \cup (\bigcup_{i=1}^p D_i)$, then $q \cdot w[1] = d(1, 1)$, and we obtain from Lemma 1 that $q \cdot w[1, n + 2] = d(1, 1)$.

Let $q \in Q_i$ for some $i \in \{1, \dots, p\}$. If for some $s \in \{1, \dots, n + 1\}$ the state $q \cdot w[1, s]$ belongs to one of the sets S_i^m or D_i, then either $q \cdot w[n+1]$ belongs to the same set S_i^m or D_i, or there is some $s' \in \{s+1, \dots, n+1\}$ such that $q \cdot w[1, s'] = d(1, 1)$ and we can apply Lemma 1. In any case, if for some $s \in \{1, \dots, n + 1\}$, one has $q \cdot w[1, s] \in (\bigcup_{(i,j)\in G} S_i^j) \cup (\bigcup_{i=1}^p D_i)$, then $q \cdot w[1, n + 2] = d(1, 1)$, because $w[n + 2] = a$.

Let $q \in Q_i \setminus \{q(0, i)\}$ and suppose that, for all $s \in \{1, \dots, n + 1\}$, one has

$$q \cdot w[1, s] \notin \left(\bigcup_{(i,j)\in G} S_i^j\right) \cup \left(\bigcup_{i=1}^p D_i\right).$$

It means that $q \cdot w[1, s] \in Q_i$ for $s \in \{1, \dots, n + 1\}$. Hence, $q = q(1, i)$. Therefore, $q \cdot w[1, n + 2] = d(1, 1)$. Thus, $\delta(Q \backslash \{q(1, 0), \dots, q(p, 0)\}, w) = \{d(1, 1)\}$. □

It follows from Lemma 2 that a word w with $|w| = n + 2$ and $w[1] = w[n + 2] = a$ is a reset word for $\mathscr{A}_{cycle}$ if and only if $q(1, 0) \cdot w = \cdots = q(p, 0) \cdot w = d(1, 1)$.

Lemma 3. *If $c_1 \wedge c_2 \wedge \cdots \wedge c_p$ is satisfiable, then there exists a reset word w of length $n + 2$ for the automaton $\mathscr{A}_{cycle}$.*

Proof. Let $w = a\alpha_1 \ldots \alpha_n a$, where for $i \in \{1, \ldots, n\}$, $\alpha_i = \begin{cases} a & \text{if } x_i = 1, \\ b & \text{if } x_i = 0. \end{cases}$ We are going to prove that $\{q(1, 0), \ldots, q(p, 0)\} \cdot w = \{d(1, 1)\}$. Let $i \in \{1, \ldots, p\}$. We have $q(i, 0) \cdot a = q(i, 1)$. We also have $c_i(x_1, \ldots, x_n) = 1$ (1 means true). Hence, there is $m \in \{1, \ldots, n\}$ such that $x_m = 1$ (in this case $w[m + 1] = \alpha_m = a$) and the variable x_m occurs in c_i; or $x_m = 0$ (in this case $w[m + 1] = \alpha_m = b$) and $\neg x_m$ occurs in c_i. Let m be the least number with such property. Then we have $q(i, 1) \cdot \alpha_1 \ldots \alpha_{m-1} = q(i, m)$ and $q(i, m) \cdot \alpha_m \in \{s(i, m, 1), d(1, 1)\}$. If $q(i, m) \cdot \alpha_m = s(i, m, 1)$ and $\alpha_{m+1} = \cdots = \alpha_n = b$, then $q(i, 1) \cdot \alpha_1 \ldots \alpha_n \in S_i^m$ and $q(i, 0) \cdot w = d(1, 1)$. Otherwise, there is a number $m' \in \{m, \ldots, n + 1\}$ such that $q(i, 1) \cdot \alpha_1 \ldots \alpha_{m'} = d(1, 1)$. In this case we obtain from Lemma 1 that $q(i, 0) \cdot w = d(1, 1)$. Therefore, by Lemma 2 we obtain $Q \cdot w = \{d(1, 1)\}$. □

Lemma 4. *If there is a reset word $w \in \{a, b\}^*$ of length $n + 2$ for the automaton $\mathscr{A}_{cycle}$, then $c_1 \wedge c_2 \wedge \cdots \wedge c_p$ is satisfiable.*

Proof. For any letter $w[1] \in \{a, b\}$, we have $\{q(1, 1), \ldots, q(1, p)\} \subseteq Q \cdot w[1]$.

We consider the word $w[2, n + 1]$. For $m \in \{1, \ldots, n\}$, we put

$$x_m = \begin{cases} 0 & \text{if } w[m + 1] = b, \\ 1 & \text{if } w[m + 1] = a. \end{cases}$$

Arguing by contradiction, suppose that $c_i(x_1, \ldots, x_n) = 0$ for some clause c_i. If for some $s \in \{2, \ldots, n + 1\}$, we have $q(i, 1) \cdot w[2, s] \in S_i^m$, then $c_i(x_1, \ldots, x_n) = 1$. Therefore, $q(i, 1) \cdot w[2, n + 1] = q(i, n + 1)$. In both cases $w[n + 2] = a$ and $w[n + 2] = b$ we obtain $q(i, 0) \cdot w = q(i, n + 1) \cdot w[n + 2] = q(i, n + 2)$. Therefore, the word w resets the automaton $\mathscr{A}_{cycle}$ to the state $q(i, n + 2)$. On the other hand, the state $q(i, n + 2)$ cannot be reached from the state $q(j, 1)$, $i \neq j$ by using the word of length $n + 2$. We obtain a contradiction. Therefore, for any $i \in \{1, \ldots, p\}$, $c_i(x_1, \ldots, x_n) = 1$. The lemma and the theorem is proved. □ □

Corollary 1. 1. *The problems SYN(CYCLE) and SYN($CYCLE_k$) for $k \geq 2$ are NP-complete.*
2. *The problems MIN-SYN(CYCLE) and MIN-SYN($CYCLE_k$) for $k \geq 2$ are NP-hard and co-NP-hard.*

Proof. 1. The problem SYN($CYCLE_2$) is a special case of SYN($CYCLE$). Hence, the latter problem is NP-complete. To reduce the problem SYN($CYCLE_2$) to SYN($CYCLE_k$) for any $k \geq 2$, we add $k - 2$ letters that act as identical transformations to the construction in the proof above.

2. The NP-hardness of the problem MIN-SYN($CYCLE_2$) for $k \geq 2$ follows from the same reduction as in the proof of Theorem 1. To prove the co-NP-hardness, we use the same automaton $\mathscr{A}_{cycle}$ constructed from given clauses $c_1, \ldots, c_p$ but put $L = n + 3$. Then the shortest reset word for the automaton $\mathscr{A}_{cycle}$ has length L if and only if there are no values for the variables $x_1, \ldots, x_n$ such that $c_1(x_1, \ldots, x_n) = \cdots = c_p(x_1, \ldots, x_n) = 1$. Therefore, the problem is co-NP-hard. The result extends to the problems MIN-SYN($CYCLE$) and MIN-SYN($CYCLE_k$) for $k \geq 2$ in an obvious way. □

3. Eulerian automata

Theorem 2. *The problem SYN($EULER_3$) is NP-complete.*

Proof. The problem SYN($EULER_3$) belongs to NP because general problem SYN belongs to NP. We use a reduction from SAT again. Take an instance of SAT consisting of the clauses $c_1(x_1, \ldots, x_n), \ldots, c_p(x_1, \ldots, x_n)$ over the Boolean variables $x_1, \ldots, x_n$. We assume that no clause contains both x_m and $\neg x_m$ for any $m \in \{1, \ldots, n\}$. We are going to construct a 3-letter automaton $\mathscr{A}_{euler} = \langle Q, \Sigma, \delta \rangle$ and an integer $L > 0$ such that there exists a reset word of length L for $\mathscr{A}_{euler}$ if and only if $c_1 \wedge c_2 \wedge \cdots \wedge c_p$ is satisfiable.

Let $\mathscr{A}_{euler} = \langle Q, \Sigma, \delta \rangle$, where

$$\Sigma = \{a, b, c\},\ Q = Z \cup \left(\bigcup_{i=1}^{p} Q_i\right) \cup \left(\bigcup_{i=1}^{p} R_i\right) \cup \left(\bigcup_{i=1}^{p} S_i\right),$$

$$Z = \{z(m) \mid m \in \{2, \ldots, n + p + 5\}\} \quad \text{and, for } i \in \{1, \ldots, p\},$$

$$Q_i = \{q(i, m) \mid m \in \{1, \ldots, n + 3\}\}, \qquad R_i = \{r(i, m) \mid m \in \{2, \ldots, n + 3\}\},$$

$$S_i = \{s(i, m) \mid m \in \{1, \ldots, p - i + 1\}\}.$$

Now we define the action of the letters a and b. Let $i \in \{1, \ldots, p\}$. For all $m \in \{1, \ldots, n\}$, we put

$$q(i,m)\cdot a = \begin{cases} r(i,m+2) & \text{if } x_m \text{ occurs in } c_i, \\ q(i,m+1) & \text{otherwise;} \end{cases}$$

$$q(i,m)\cdot b = \begin{cases} r(i,m+2) & \text{if } \neg x_m \text{ occurs in } c_i, \\ q(i,m+1) & \text{otherwise.} \end{cases}$$

$$q(i,n+1)\cdot a = q(i,n+1)\cdot b = q(i,n+2);\ q(i,n+2)\cdot a = q(i,n+2)\cdot b = q(i,n+3);$$
$$q(i,n+3)\cdot a = r(i,n+3)\cdot a = s(i,1);\ q(i,n+3)\cdot b = q(i,n+3);$$
$$r(i,n+3)\cdot b = r(i,n+3).$$

For $m \in \{2, \ldots, n+2\}$, we put $r(i,m)\cdot a = r(i,m)\cdot b = r(i,m+1)$.
For $m \in \{1, \ldots, p-i+1\}$, we put

$$s(i,m)\cdot b = s(i,m); \qquad s(i,m)\cdot a = \begin{cases} s(i,m+1) & \text{if } m < p+i-1, \\ s(i-1,p-i+2) & \text{if } m = p+i-1, i > 1, \\ z(n+p+4) & \text{if } m = p, i = 1. \end{cases}$$

For $m \in \{2, \ldots, n+p+2\}$, we put $z(m)\cdot a = z(m)\cdot b = z(m+1)$. We also put

$$z(n+p+3)\cdot a = z(n+p+4); \qquad z(n+p+3)\cdot b = z(n+p+3);$$
$$z(n+p+4)\cdot a = z(n+p+5); \qquad z(n+p+4)\cdot b = z(n+p+4);$$
$$z(n+p+5)\cdot a = z(n+p+5)\cdot b = z(n+p+5).$$

The letters a and b encode satisfiability of the clauses. Now we define the action of the letter c such that the automaton $\mathscr{A}$ becomes Eulerian. For $q \in Q$, we put $q \cdot c = q$ except the following cases. Let $i \in \{1, \ldots, p\}$, $m \in \{1, \ldots, n\}$. If either x_i or $\neg x_i$ occurs in c_m, then we put $r(i,m+2)\cdot c = q(i,m+1)$. Besides that, we put

$$q(i,n+3)\cdot c = s(i,1)\cdot c = q(i,1); \qquad r(i,n+3)\cdot c = r(i,2) \quad \text{for } i \neq p;$$
$$s(i,p-i+1)\cdot c = r(i+1,2); \qquad z(n+p+3)\cdot c = z(n+p+4)\cdot c = z(2);$$
$$z(n+p+5)\cdot c = r(1,2).$$

An example of the automaton $\mathscr{A}_{euler}$ is presented in Fig. 2. The action of the letters a, b, c is shown by solid, dashed and dotted lines respectively. The states in Fig. 2 are organized in several columns containing respectively the sets Z, $Q_1 \cup S_1$, R_1, $Q_2 \cup S_2$, R_2, and so on. We put $L = n + p + 3$.

In general, the states of the automaton $\mathscr{A}_{euler}$ can be partitioned in $n+p+5$ "rows" $T_1, \ldots, T_{n+p+5}$. We put $T_1 = \{q(1,1), \ldots, q(p,1)\}$; for $m \in \{2, \ldots, n+3\}$ we put $T_m = \{z(m), q(1,m), r(1,m), \ldots, q(p,m), r(p,m)\}$ and for $m \in \{n+4, \ldots, n+p+4\}$ we put $T_m = \{z(m), s(m-n-3,1), \ldots, s(m-n-3, p-m+n+4)\}$; we also put $T_{n+p+5} = \{z(n+p+5)\}$. Clearly, the size of the automaton $\mathscr{A}_{euler}$ is a polynomial function of the size of the clauses $c_1, \ldots, c_p$.

Lemma 5. *The DFA $\mathscr{A}_{euler}$ is Eulerian.*

Proof. It is easy to check that for any state $q \in Q$ there exist exactly 3 pairs $(r, \alpha) \in Q \times \Sigma$ such that $r \cdot \alpha = q$. □

Let $U \subseteq Q$, $\Theta \subseteq \Sigma$. We denote by $d_\Theta(U)$ the minimum length of words $w \in \Theta^*$ such that $|U \cdot w| = 1$. Thus, the minimum length of reset words for $\mathscr{A}_{euler}$ is equal to $d_\Sigma(Q)$.

Lemma 6. $d_\Sigma(Q) = d_{\{a,b\}}(Q) \in \{n+p+3, n+p+4\}$.

Proof. It is immediate to check that the word a^{n+p+4} is a reset word for DFA $\mathscr{A}_{euler}$. Therefore $d_\Sigma(Q) \leq n+p+4$. We notice that $d_{\{a,b\}}(\{z(2), z(n+p+5)\}) = n+p+3$. Therefore $d_{\{a,b\}}(Q) \geq n+p+3$.

Now let w be a shortest reset word for the $\mathscr{A}_{euler}$ and suppose that the letter c occurs in w. We aim to prove that $|w| \geq n+p+4$. It is not difficult to verify that $d_\Sigma(\{z(2), r(1,2)\}) = n+p+2$ and there is no word $u \in \Sigma^* \setminus \{a,b\}^*$ of length $n+p+2$ such that $z(2)\cdot u = r(1,2)\cdot u$. Let i be the position of the leftmost occurrence of the letter c in w. If $i \geq n+p+4$, then $|w| \geq n+p+4$. Let $i \leq n+p+3$, then we have $z(n+p+4), z(n+p+5) \in Q \cdot w[1, i-1]$. Hence $z(2), r(1,2) \in Q \cdot w[1,i]$. Therefore $|w| \geq i + d_\Sigma(\{z(2), r(1,2)\}) = i+n+p+2$. If $i \geq 2$, then $|w| \geq n+p+4$. If $i = 1$, then $w[1] = c$ and $\{z(2), z(3), r(1,2)\} \subseteq Q \cdot w[1]$. It is not difficult to prove that $d_\Sigma(\{z(2), z(3), r(1,2)\}) \geq n+p+3$. Hence, $|w| \geq n+p+4$ in any case. Therefore $d_\Sigma(Q) = d_{\{a,b\}}(Q)$. □

In particular, the lemma claims that there exists a reset word of the minimum length in which the letter c does not occur. We notice that $T_{n+p+5}\cdot a = T_{n+p+5}\cdot b = T_{n+p+5}$, and $T_m \cdot a, T_m \cdot b \subseteq (T_m \cup T_{m+1})$ for $m \in \{n+1, \ldots, n+p+4\}$ while $T_m \cdot a, T_m \cdot b \subseteq T_{m+1} \cup T_{m+2}$ for $m \in \{1, \ldots, n\}$. The following lemma is an immediate corollary of this property.

Lemma 7. 1. *Every reset word from $\{a,b\}^*$ resets the DFA $\mathscr{A}_{euler}$ to the state $z(n+p+5)$.*
2. $d_{\{a,b\}}(T_{n+2} \cup \cdots \cup T_{n+p+5}) = p+3$.

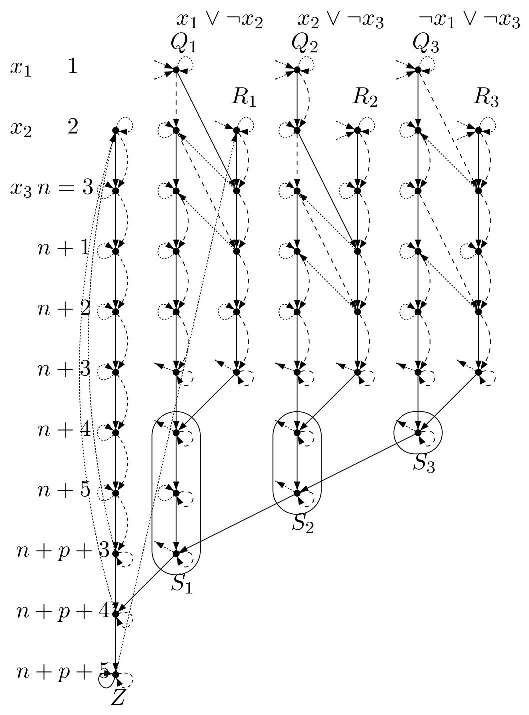

Fig. 2. The automaton $\mathscr{A}_{euler}$ for the clauses $x_1 \vee \neg x_2, x_2 \vee \neg x_3, \neg x_1 \vee \neg x_3$.

3. *If a word $w \in \{a, b\}^*$ of length $n + p + 3$ is a reset word for $\mathscr{A}_{euler}$, then $Q \cdot w[1, n] \subseteq T_{n+2} \cup \cdots \cup T_{n+p+5}$.*
4. *If a word $w \in \{a, b\}^*$ of length $n + p + 3$ is a reset word for $\mathscr{A}_{euler}$, then $T_1 \cdot w[1, n] \subseteq T_{n+2}$.*
5. *If there is a word w of length n such that $Q \cdot w \subseteq T_{n+2} \cup \cdots \cup T_{n+p+5}$, then the word wa^{p+3} is a reset word of length $n + p + 3$ for $\mathscr{A}_{euler}$.*
6. *If there is a word w of length n such that $T_1 \cdot w \subseteq T_{n+2}$, then there is a reset word of length $n + p + 3$ for $\mathscr{A}_{euler}$.*
7. $d_{\{a,b\}}(Q) = d_{\{a,b\}}(T_1)$.

Lemma 8. *If $c_1 \wedge c_2 \wedge \cdots \wedge c_p$ is satisfiable, then there exists a reset word w of length $n + p + 3$ for the automaton $\mathscr{A}_{euler}$.*

Proof. Let $w = \alpha_1 \ldots \alpha_n a^{p+3}$, where for $i \in \{1, \ldots, n\}$, $\alpha_i = \begin{cases} a & \text{if } x_i = 1, \\ b & \text{if } x_i = 0. \end{cases}$ We are going to prove that $T_1 \cdot w = \{z(n + p + 5)\}$. Let $i \in \{1, \ldots, p\}$. We have $c_i(x_1, \ldots, x_n) = 1$. Hence, there is $m \in \{1, \ldots, n\}$ such that $x_m = 1$ (in this case $w[m] = a$) and x_m occurs in c_i or $x_m = 0$ (in this case $w[m] = b$) and $\neg x_m$ occurs in c_i. Let m be the least number with this property. We obtain from the definition of the action of a and b that

$$q(i, 1) \cdot w[1, m-1] = q(i, m) \quad \text{and} \quad q(i, m) \cdot w[m] = r(i, m+2).$$

For any word $w[m + 1, n] \in \{a, b\}^*$ we have $r(i, m + 2) \cdot w[m + 1, n] = r(i, n + 2) \in T_{n+2}$. Hence, from Lemma 7, w is a reset word for the automaton $\mathscr{A}_{euler}$. □

Lemma 9. *If there is a reset word $w \in \{a, b\}^*$ of length $p + n + 3$ for the automaton $\mathscr{A}_{euler}$, then $c_1 \wedge c_2 \wedge \cdots \wedge c_p$ is satisfiable.*

Proof. The word w resets DFA $\mathscr{A}_{euler}$. Therefore, we have $T_1 \cdot w[1, n] \in T_{n+2}$. For $m \in \{1, \ldots, n\}$ we put

$$x_m = \begin{cases} 0 & \text{if } w[m] = b, \\ 1 & \text{if } w[m] = a. \end{cases}$$

Arguing by contradiction, suppose that $c_i(x_1, \ldots, x_n) = 0$ for some clause c_i. If, for some $m \in \{1, \ldots, n\}$, we would have $q(i, 1) \cdot w[1, m] = r(\imath, m + 2)$, then $c_i(x_1, \ldots, x_n) = 1$. Therefore, $q(i, 1) \cdot w[1, n] = q(i, n + 1) \in T_{n+1}$. By Lemma 7, the word w is not a reset word for $\mathscr{A}_{euler}$. We obtain a contradiction. Therefore, for any $i \in \{1, \ldots, p\}$, we have $c_i(x_1, \ldots, x_n) = 1$. □ □

Corollary 2. 1. *The problems SYN(EULER) and SYN(EULER$_k$) for $k \geq 3$ are NP-complete.*
2. *The problems MIN-SYN(EULER) and MIN-SYN(EULER$_k$) for $k \geq 3$ are NP-hard and co-NP-hard.*

Proof. The proof is the same as the proof of Corollary 1. □

4. Conclusion and conjectures

We proved that problems SYN(*EULER*) and SYN(*CYCLE*) are NP-complete. This means that these problems have the same complexity as the general problem SYN stated for the class of all DFA. At the same time we proved that the problems MIN-SYN(*EULER*) and MIN-SYN(*CYCLE*) are NP-complete and co-NP-complete. But it is only the lower bound and it does not seem to be tight. The general problem MIN-SYN is DP-complete for a class of all DFA (see [9]). It is natural to conjecture the following.

Conjecture 1. *For any integer $k \geq 2$ the problems MIN-SYN(EULER), MIN-SYN(CYCLE), MIN-SYN($EULER_k$), MIN-SYN($CYCLE_k$) are DP-complete.*

The NP-completeness of the problem SYN($EULER_2$) now is also unproved. It may happen that if problem SYN($EULER_2$) can be solved in a polynomial time, then the problem MIN-SYN($EULER_2$) can be also solved in polynomial time.

It follows from [3] that no polynomial time algorithm approximates the length of the shortest synchronizing word within a constant factor for a given DFA. There is no such algorithm, even for automata over the binary alphabet. So we can conjecture the following.

Conjecture 2. *No polynomial time algorithm approximates the length of the shortest synchronizing word within constant factor for a given cyclical or Eulerian DFA. There is no such algorithm, even for DFA over the binary alphabet.*

Acknowledgments

The author acknowledges support from the Presidential Program for young researchers, grant MK-266.2012.1.
The work was supported by the Federal Education Agency of Russia, project 2.1.1/3537.

References

[1] D.S. Ananichev, M.V. Volkov, Synchronizing monotonic automata, Theoret. Comput. Sci. 327 (2004) 225–239.
[2] M.-P. Beal, A note on Cernys conjecture and rational series, Preprint IGM 2003-05, Unpublished, 2003.
[3] M. Berlinkov, Approximating the Minimum Length of Synchronizing Words is Hard, Proc. of CSR2010, Kazan, Russia, in: LNCS, vol. 6072, pp. 37–47.
[4] J. Černý, Poznámka k homogénnym eksperimentom s konecnými avtomatami, Mat.-Fyz. Čas. Slovensk. Akad. Vied. 14 (1964) 208–216 (in Slovak).
[5] L. Dubuc, Sur les automates circulaires et la conjecture de Černý, RAIRO Inform. Theor. Appl. 32 (1998) 21–34 (in French).
[6] D. Eppstein, Reset sequences for monotonic automata, SIAM J. Comput. 19 (1990) 500–510.
[7] J. Kari, Synchronizing finite automata on Eulerian digraphs, Theoret. Comput. Sci. 295 (2003) 223–232.
[8] P. Martyugin, Complexity of problems concerning reset words for some partial cases of automata, Acta Cybernet. 19 (2009) 517–536.
[9] J. Olschewski, M. Ummels, The Complexity of Finding Reset Words in Finite Automata, Proc. of MFCS 2010, in: LNCS, vol. 6281, 2010, pp. 568–579.
[10] J.-E. Pin, On two combinatorial problems arising from automata theory, Ann. Discrete Math. 17 (1983) 535–548.
[11] A. Salomaa, Composition sequences for functions over a finite domain, Theoret. Comput. Sci. 292 (2003) 263–281.
[12] M.V. Volkov, Synchronizing automata preserving a chain of partial orders, Theoret. Comput. Sci. 410 (2009) 3513–3519.