An efficient algorithm finds noticeable trends and examples concerning the Černy conjecture

A.N. Trahtman***

Bar-Ilan University, Dep. of Math., 52900, Ramat Gan, Israel

Lectures Notes in Computer Science, 4162(2006), 789-800

Abstract. A word w is called synchronizing (recurrent, reset, directed) word of a deterministic finite automaton (DFA) if w sends all states of the automaton on a unique state. Jan Černy had found in 1964 a sequence of n-state complete DFA with shortest synchronizing word of length $(n-1)^2$. He had conjectured that it is an upper bound for the length of the shortest synchronizing word for any n-state complete DFA. The examples of DFA with shortest synchronizing word of length $(n-1)^2$ are relatively rare. To the Černy sequence were added in all examples of Černy, Piricka and Rosenauerova (1971), of Kari (2001) and of Roman (2004).

By help of a program based on some effective algorithms, a wide class of automata of size less than 11 was checked. The order of the algorithm finding synchronizing word is quadratic for overwhelming majority of known to date automata. Some new examples of n-state DFA with minimal synchronizing word of length $(n-1)^2$ were discovered. The program recognized some remarkable trends concerning the length of the minimal synchronizing word.

http://www.cs.biu.ac.il/∼trakht/Testas.html.

Keywords: deterministic finite automaton, synchronizing word, algorithm, complexity, Černy conjecture.

Introduction

We consider a DFA with complete state transition graph Γ and transition semigroup S over alphabet Σ. Let n be the size of DFA and q be the size of Σ.

The problem of synchronization of DFA is natural and various aspects of this problem were touched upon the literature. Synchronization makes the behavior of an automaton resistant against input errors since, after detection of an error, a synchronizing word can reset the automaton back to its original state, as if no error had occurred. Therefore different problems of synchronization draw the attention.

* Email: trakht@macs.biu.ac.il

** this version differs of LNCS version. Here an untrue lemma is omitted

A problem with a long story is the estimation of the minimal length of synchronizing word. Most known as a Černy conjecture, it was aroused independently by distinct authors. Jan Černy had found in 1964 [2] n-state complete DFA with shortest synchronizing word of length $(n-1)^2$ for $q = 2$. He had conjectured that it is an upper bound for the length of the shortest synchronizing word for any n-state complete DFA. The problem can be reduced to automata with strongly connected graph [2]. The best known upper bound is now equal to $(n^3 - n)/6$ [5], [8], [9], [12]. The conjecture holds true for a lot of automata, but in general the problem remains open. This simply looking conjecture is now one of the most longstanding open problems in the theory of finite automata. Moreover, the examples of automata with shortest synchronizing word of length $(n-1)^2$ are infrequent. After the sequence found by Černy and example of Černy, Piricka and Rosenauerova [3] of 1971 for $q = 2$, the next such example was found by Kari [6] only in 2001 for $n = 6$ and $q = 2$. Roman [14] had found an analogical example for $n = 5$ and $q = 3$ in 2004. There are no examples of automata for the time being such that the length of the shortest synchronizing word is greater than $(n-1)^2$.

The testing of synchronizing automata is an indispensable part of investigation in this area [1], [4], [10], [11], [13], [19]. The best known to date algorithm of Eppstein [4], [10] improves an algorithm of Natarjan [11] and finds a synchronizing word for n-state DFA in $O(n^3 + n^2q)$ time.

We present a new efficient algorithm for finding a synchronizing word. The actual running time of the algorithm on a lot of examples proved to be essentially less than in case of $O(n^3q)$ time complexity. For clear majority of automata, the time complexity is $O(n^2q)$. It gives a chance to extend noticeably the class of considered DFA. This algorithm plays a central role in the program for search of automata with minimal reset word.

The program studied all automata with strongly connected transition graph of size $n \leq 10$ for $q = 2$ and of size $n \leq 7$ for $q \leq 4$. All known and some new examples of DFA with shortest synchronizing word of length $(n-1)^2$ from this class of automata were checked. So all examples of DFA with shortest synchronizing word of length $(n-1)^2$ in this area are known for today. The size of the alphabet of the examples is two or three. The situation in the neighborhood of the bound $(n-1)^2$ of Černy (minimal reset words of relatively great length) was also studied.

There are no contradictory examples for the Černy conjecture in this class of automata. Moreover, the program does not find new examples of DFA with reset word of length $(n-1)^2$ for automata with $n > 4$ as well as for $q > 3$. No such examples exist for alphabet of size four if $n \leq 7$.

And what is more, the examples with minimal length of reset word disappear even for values near the Černy bound $(n-1)^2$ with growth of the size of the automaton as well as of the size of the alphabet. The gap between $(n-1)^2$ and the nearest of the minimal lengths of reset word appears for $n = 6$. There are no 6-state automata with minimal length of synchronizing word of 24 for $q \leq 4$. The following table displays this interesting trend for the length of minimal reset

words less than $(n-1)^2$.

size	n=5 $q <= 4$	n=6 $q <= 4$	n=7 $q <= 4$	n=8 q=2	n=9 q=2	n=10 q=2
$(n-1)^2$	16	25	36	49	64	81
max length	15	23	32	44	58	74

The program uses also straightforward algorithm for finding synchronizing word of minimal length. A help algorithm of the program verifies whether or not a given DFA is synchronizing. It is a modification of an algorithm of $O(n^2q)$ time complexity supposed by Eppstein [4], [10]. Our version has $O(n^2q)$ time complexity only in the worst case and we use usually only its linear part.
The comparison of the experimental data suggests that the length of the synchronizing word found by central algorithm of the program is not far from the length of the minimal synchronizing word. This length was not greater than n^2 in all billions cases studied for today. The results of the algorithms altogether correspond to the Černy conjecture. All above algorithms are implemented in our package TESTAS [19].

Preliminaries

Let us consider a deterministic finite automaton with state transition graph Γ and transition semigroup S over alphabet Σ. The states of the automaton are considered below as vertices of the transition graph Γ.
The number of vertices of the graph Γ is denoted by $|\Gamma|$.
A maximal strongly connected component of a directed graph will be denoted for brevity as **SCC**.
If there exists a path $v \in \Sigma^+$ from vertex $\mathbf{p}$ to vertex $\mathbf{q}$ in the transition graph of DFA then let us denote the vertex $\mathbf{q}$ as $\mathbf{p}v$.
Let Γv denote the mapping of the graph [automaton] Γ by help of $v \in \Sigma^+$, let us call $|\Gamma v|$ *rank* and $|\Gamma| - |\Gamma v|$ *defect* of the mapping v.
A word $v \in \Sigma^+$ is called *synchronizing word* of an automaton A with transition graph Γ if $|\Gamma v| = 1$. An automaton (and its transition graph) possessing a synchronizing word is called *synchronizing*.
A word w is called *2-reset word* of the pair $\mathbf{p}, \mathbf{q}$ if $\mathbf{p}w = \mathbf{q}w$.
Suppose $\mathbf{p} \succeq \mathbf{q}$ if $\mathbf{p}w = \mathbf{q}$ for some word w.
A state [a vertex] $\mathbf{q}$ is called *sink* of an automaton [of a graph] if $\mathbf{p} \succeq \mathbf{q}$ for all $\mathbf{p}$.
An automaton [a graph Γ] is called *complete* if for every state [vertex] $\mathbf{p}$ and every $\sigma \in \Sigma$ the state [vertex] $\mathbf{p}\sigma$ exists.
The direct product Γ^2 of two copies of graph Γ over an alphabet Σ consists of vertices $(\mathbf{p}, \mathbf{q})$ and edges $(\mathbf{p}, \mathbf{q}) \to (\mathbf{p}\sigma, \mathbf{q}\sigma)$ labelled by σ. Here $\mathbf{p}, \mathbf{q} \in \Gamma$, $\sigma \in \Sigma$.

1 Some auxiliary properties

Two following two simple lemmas belong rather to the folklore.

Lemma 1 *[2] [18], [10] The directed labelled graph Γ is synchronizing if and only if Γ^2 has sink state.*

Lemma 2 *[18] The sets of synchronizing words of the graphs Γ and Γ^2 coincide.*

Lemma 3 *Suppose $\mathbf{p} \notin \Gamma s$ for a word s and a state $\mathbf{p}$ of transition graph Γ of DFA.*
Then there exist two minimal integer k and r such that $\mathbf{p}s^k = \mathbf{p}s^{k+r}$. The pair of states $\mathbf{p}, \mathbf{p}s^r$ has 2-reset word s^k and for every $i < k$ the pair of states $\mathbf{p}s^i, \mathbf{p}s^{r+i}$ has 2-reset word s^{k-i}. The word s^k is a 2-reset word for at least k different pairs of states.
In the case $r = 1$ every pair of states $\mathbf{p}s^i, \mathbf{p}s^k$ for every $i < k$ has 2-reset word s^{k-i}.

Proof. The sequence $\mathbf{p}s, \mathbf{p}s^2, ..., \mathbf{p}s^t, ...$ is finite and belongs to Γs. Therefore such k and r exist. Two states $\mathbf{p}s^i$ and $\mathbf{p}s^{r+i}$ are mapped by the power s^{k-i} on $\mathbf{p}s^k = \mathbf{p}s^{k+r}$ as well as the states $\mathbf{p}$ and $\mathbf{p}s^r$ are mapped by the power s^k on $\mathbf{p}s^k$. All states $\mathbf{p}s^i$ are distinct for $i \leq k$, whence the word s^k unites at least k distinct pairs of states.
In the case $r = 1$, two states $\mathbf{p}s^i$ and $\mathbf{p}s^k$ are mapped by the word s^{k-i} on $\mathbf{p}s^k = \mathbf{p}s^{k+1}$ as well as the pair of states $\mathbf{p}$, $\mathbf{p}s^k$ is mapped by the power s^k on $\mathbf{p}s^k$. All states $\mathbf{p}s^i$ are distinct for $i \leq k$, whence the word s^k unites also in this case at least k distinct pairs of states.

Lemma 4 *Suppose $\mathbf{r}\alpha = \mathbf{t}\alpha$ for a letter α and two distinct states $\mathbf{r}$, $\mathbf{t}$ of transition graph Γ of DFA and let the states $\mathbf{r}$ and $\mathbf{r}\alpha$ be consecutive states of a cycle C of Γ.*
Then there exists a word s of length of the cycle C such that $\mathbf{r}s = \mathbf{r}$ and $|\Gamma s| < |\Gamma|$. For some state $\mathbf{p} \in \Gamma \setminus \Gamma s$ there exists a minimal integer k such that $\mathbf{p}s^k = \mathbf{p}s^{k+1}$. The pair of states $\mathbf{p}, \mathbf{p}s^k$ has 2-reset word s^k and for every $i < k$ the pair of states $\mathbf{p}s^i, \mathbf{p}s^k$ has 2-reset word s^{k-i}. The word s^k unites at least $k + 1$ distinct states.

Proof. A word s with first letter α can be obtained from consecutive letters on the edges of the cycle C. Therefore $|s|$ is equal to the length of the cycle and $\mathbf{r}s = \mathbf{r}$. $|\Gamma s| < |\Gamma|$ follows from $\mathbf{r}\alpha = \mathbf{t}\alpha$.
From $\mathbf{r}s = \mathbf{r} \neq \mathbf{t}$ and $\mathbf{r}\alpha = \mathbf{t}\alpha$ follows that $\mathbf{t}s = \mathbf{r} \neq \mathbf{t}$ and $\mathbf{t}s^i \neq \mathbf{t}$ for any integer i. In the case $\mathbf{t} \in \Gamma \setminus \Gamma s$ suppose $\mathbf{p} = \mathbf{t}$, and so the state $\mathbf{p}$ is defined. In opposite case for some state $\mathbf{t}_1$ holds $\mathbf{t}_1 s = \mathbf{t}$. If $\mathbf{t}_1 \in \Gamma \setminus \Gamma s$ suppose $\mathbf{p} = \mathbf{t}_1$, else for some state $\mathbf{t}_2$ holds $\mathbf{t}_2 s^2 = \mathbf{t}$. Let us continue this procedure until $\mathbf{t}_{k-1} \in \Gamma \setminus \Gamma s$ for some k such that $\mathbf{t}_{k-1} s^{k-1} = \mathbf{t}$. Such minimal k exists and all states $\mathbf{t}$, $\mathbf{t}_1$, ..., $\mathbf{t}_j$ for $j \leq k$ are distinct because $\mathbf{t}s^i \neq \mathbf{t}$ for any integer i. The state $\mathbf{t}$ therefore has a preimage $\mathbf{p} = \mathbf{t}_{k-1}$ in $\Gamma \setminus \Gamma s$ by mapping s^{k-1}, whence $\mathbf{p}s^k = \mathbf{p}s^{k+1} = \mathbf{r}$. So the pair of states $\mathbf{p}, \mathbf{p}s^k$ has 2-reset word s^k and for every $i < k$ the pair of states $\mathbf{p}s^i, \mathbf{p}s^k$ has 2-reset word s^{k-i}. The states $\mathbf{p}s^i$ for $i \leq k$ and $\mathbf{p}$ are distinct because of the choice of k. The word s^k maps all these states on the state $\mathbf{r}$.

Obvious is the following

Lemma 5 *Suppose* $\mathbf{q}s = \mathbf{q}$ *for* m *states* $\mathbf{q}$ *from* Γ *and for some word* s *such that* $s^k = s^{k+1}$. *Then* $|\Gamma s^k| = m$.

2 Synchronizing Algorithms

The following help construction was supposed by Eppstein [4]. Let us keep for any pair of states $\mathbf{r}, \mathbf{q}$ the first letter α of the minimal 2-reset word w of the pair of states together with the length of the word w. The corresponding letter of the pair of states $\mathbf{r}\alpha, \mathbf{q}\alpha$ is the second letter of w. The 2-reset word w of minimal length can be restored on this way. The time and space complexity of this preprocessing is $O(|\Gamma^2|)$ [4] and it will be used in majority of considered algorithms.
A help algorithm with $O(|\Gamma|^2 q)$ time complexity in the worst case based on Lemmas 1 and 2 verifies whether or not a given DFA is synchronizing [4], [19]. The main part of the algorithm follows [4] (see also [10]). Our modification of the algorithm finds first all SCC of the graph (a linear algorithm) and then checks the minimal SCC of the graph (if exists). The program for search of automata with relatively great minimal reset word uses this algorithm on the preliminary (and quite often linear) stage.
An efficient semigroup algorithm, essential improvement of the algorithm from [4], based on the properties of syntactic semigroup and inspired by Lemmas 3 - 5 is used on the next stage and plays a central role in the program.

2.1 A semigroup algorithm for synchronizing word

We consider the square Γ^2 and the reverse graph I of Γ. The graph I is not deterministic for synchronizing graph Γ.
Suppose that the graph Γ is synchronizing, all sink states are found on the stage of checking of the synchronizability, the graph Γ^2 and the reverse graph I were build.
Let us find by help of the reverse graph I for any pair of states $\mathbf{r}, \mathbf{q}$ from Γ^2 the first letter of the minimal 2-reset word w of the pair and the length of w [4]. So for any pair of states $(\mathbf{r}, \mathbf{q})$ can be restored a 2-reset word w of minimal length. The set of states $(\mathbf{r}, \mathbf{q})$ can be ordered according to the length of the word w. The ordering can be made linear in the size of the set. One can find first the number of all pairs $(\mathbf{r}, \mathbf{q})$ with given length of minimal 2-reset word for any length, then adjust an interval for to place the pairs and then allocate the pairs of states in the interval according to the value of the length.
We use also an another idea for to reorder the pairs of states. The number of preimages of the state $\mathbf{r}w = \mathbf{q}w$ by mapping w^k for any integer k can be used for the ordering together with the length $|w|$. Let us call this order *the second*. The number of preimages can be found in linear time for given pair of states

$(\mathbf{r}, \mathbf{q})$ using the reverse graph I. The corresponding words may form a set of generators of a subsemigroup of the semigroup A of all reset words and we will use only linear number of pairs studied for this aim.
The important part of the preprocessing supposed by Eppstein was the computing of the mapping Γw of the graph Γ induced by the minimal 2-reset word w of the pair of states $\mathbf{r}, \mathbf{q}$. This stage begins from the shortest words w and therefore is linear for any considered pair of states $\mathbf{r}, \mathbf{q}$. Nevertheless, the time complexity of the stage is $O(\Gamma^3)$. For to avoid the extremes of this step, our algorithm stops on linear number of pairs. The obtained set G of 2-reset words is considered as a set of generators of some subsemigroup from A and will be marked together with corresponding pairs of states. The time complexity of this step is therefore $O(\Gamma^2)$. Let us reorder G in the *second* order and use the mapping of the graph induced by powers of generators.
Let Γ_i be consecutive images of the graph $\Gamma = \Gamma_0$ such that for $w_i \in A$ holds $\Gamma_i w_{i+1} = \Gamma_{i+1}$ and $|\Gamma_i| > |\Gamma_{i+1}|$. Let A_i be a semigroup generated by the set w_1, ... w_i. Let us check pairs of states corresponding to the words from G. If the pair belongs to Γ_i then the corresponding minimal reset word w_{i+1} may be used for to find the image Γ_{i+1}.
In the case no minimal 2-reset word of a pair from Γ_i was marked, let us consider the products of marked words. If some product unites a pairs of states of Γ_i, then let us use the mapping, mark the product of words and the pair of states. Let us notice that on this step are considered not all marked pairs. The number of considered products must be linear in the size of Γ. The product of two mappings can be found in linear time. Therefore the time complexity of this stage is $O(|\Gamma|k)$ for the defect k of the mapping of Γ_i.
If two considered stages still do not find a reset word, then the new generator must be added to considered subsemigroup A_i. Let us take a pair of states $\mathbf{r}, \mathbf{q}$ from Γ_i with reset word w_i. Suppose $w_i = u_i v_i$ such that the word v_i was marked. Then the mapping w_i can be found in $|\Gamma||u_i|$ time. Let us notice that only on this step the time complexity may by greater than quadratic.

Lemma 6 *Let Γ_i be consecutive images of the graph $\Gamma = \Gamma_0$ such that for v_i from semigroup A $\Gamma_i v_{i+1} = \Gamma_{i+1}$, $|\Gamma_i| > |\Gamma_{i+1}|$ and $|\Gamma_s| = 1$ for some integer s. Let A_i be a semigroup generated by the set w_1, ... w_i such that $w_i = u_i v_i$ is a reset word for some pair of states from Γ_{i-1} and v_i is a marked element of the subsemigroup A_{i-1}.*
Then the considered algorithm has $max(O(|\Gamma|^2 q), O(|\Gamma||u_1 ... u_s|)$ time complexity.

Proof. The time complexity of the step of the building of Γ^2 is $O(|\Gamma|^2 q)$. So $O(|\Gamma|^2 q)$ is a lower bound for the complexity of the considered algorithm.
Let the set w_1, ... w_i generate A_i. The creation of the mapping w_i needs $|\Gamma||u_i|+1$ steps because for the marked element v_i the mapping is known.
The element will be marked and used only if it is either a generator from A_i or a product of two marked elements. With a marked semigroup element will be associated the mapping of Γ defined by the element. The finding of the mapping

of the product of two elements with known images is linear in the size of the graph.

We repeat the process with the obtained image Γ_i. The defect of the mapping is growing on every step. After not over than $|\Gamma| - 1$ steps Γ will be synchronized. The process of recording of the synchronizing word is linear in the length of the word. The length of the synchronizing word found by the algorithm in billions of practical experiments was less than $|\Gamma|^2$ in all considered cases. The stage of adding of new generators was used only in a small number of cases, only some percents of considered synchronizing automata. The minimal number of generators of the semigroup A is usually small. For instance, for all Černy graphs there are only two generators. Therefore the time complexity of the algorithm is $O(|\Gamma|^2 q)$ in overwhelming majority of cases and the algorithm can be considered as almost quadratic.

2.2 Modification of Eppstein algorithm

Some version of the program uses also a modification of Eppstein algorithm [4], [10] for finding synchronizing word of $O(|\Gamma|^3 + (|\Gamma|^2 q)$ time complexity. The favorable idea of Eppstein was to keep with any pair of states $\mathbf{r}, \mathbf{q}$ the first letter of the minimal reset word w, its length and the image of the set of states by help of the mapping induced by the word w. The building of the images has $O(|\Gamma|^3)$ time complexity and is a most wasteful part of the algorithm.
Our modification of the Eppstein algorithm (called below a cycle algorithm) instead of a word w considers a power of this word until stabilization of the rank of the image. It proved to be fruitful in many cases including such extraordinary case as graphs of Černy [2]. The length of the reset word obtained by the algorithm in this case reaches its minimum. We omit sometimes this stage of the program despite the growing number of the graphs studied on the next stage. Nevertheless, the observation period of the whole of the program is essentially smaller in spite of the fact that the next stage is non-polynomial.

Theorem 7 *[5], [8] Let C be set of size k and let us consider a sequence of its subsets C_i of size m such that any C_i includes a two-element subset of C not included in every C_j for $j < i$. Then the length of the sequence is less than $(k - m + 2) * (k - m + 1)/2$.*

Corollary 8 *Let Γ be transition graph of an automaton with $|\Gamma|$ states and let us consider a sequence of subsets C_i of states of the automaton of size m or less such that any C_i includes a two-element subset of states of Γ not included in every C_j for $j < i$. Suppose the length of the sequence is $(|\Gamma| - m + 2) * (|\Gamma| - m + 1)/2$. Then at least one C_i contains less than m states. Any sequence of length $(|\Gamma|^3 - |\Gamma|)/6$ of considered kind for distinct m contains a set of size one.*

The value $(|\Gamma|^3 - |\Gamma|)/6$ is well known and was mentioned time and again [5], [8], [9], [12]. The combinatorial theorem 7 can be used for estimation of the length

of the reset word obtained by Eppstein, cycle and semigroup algorithms. The theorem considers distinct mappings of the graph of the automaton induced by the letters of the alphabet of the labels such that any new mapping has at least one pair of states that does not belong to any previous mapping of the same rank. For given rank k of mapping in considered algorithms there are at most $(|\Gamma| + k)(|\Gamma| + k - 1)/2$ or less than $|\Gamma|$ distinct mappings. The pair of states with a most short reset word creates a sequence of such mappings and therefore the theorem 7 can be used here. Corollary 8 implies

Proposition 9 *The length of the reset word obtained by Eppstein, cycle and semigroup algorithms is less than* $(|\Gamma|^3 - |\Gamma|)/6$.

So the time complexity of the algorithm in the most worst case is $O(|\Gamma|^3 q)$. Really this most worst case is very rare, for all automata studied for today by these algorithms, it was less than $|\Gamma|^2$.

2.3 An algorithm for finding synchronizing word of minimal length

On the last stage, the program uses a straightforward algorithm for finding synchronizing word of minimal length. The last one is not polynomial in the most worst case (the finding of the synchronizing word of minimal length is NP-hard [4], [10], [15]). The program for search of minimal reset word uses this algorithm relatively rare.
The algorithm is a revision of an algorithm for finding the syntactic semigroup S of size s with q generators on the base of transition graph [17]. We find mappings of the graph of the automaton induced by the letters of the alphabet of the labels. Mappings with the same set of states are identified. It essentially simplified the process in comparison with the algorithm from [17]. Distinct mappings are saved. For this aim, any two mappings must to be compared, so we have $O(s(s-1)/2)$ steps. Let us notice that the size of the syntactic semigroup is in general not polynomial in the size of the transition graph.
The mappings correspond to semigroup elements. With any mapping let us connect a previous mapping and the letter that creates the mapping. On this way, the path on the graph of the automaton can be constructed.

Proposition 10 *The algorithm finds a list of all words (elements of syntactic semigroup) of length k where k is growing. The first synchronizing word of the list has minimal length.*

The time complexity of the considered procedure is $O(|\Gamma| q s^2)$ with $O(|\Gamma| s)$ space complexity.

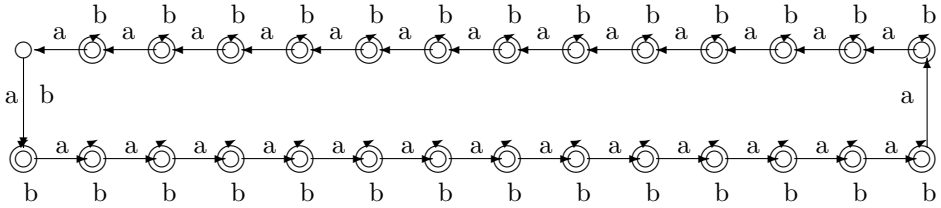
2.4 Checking synchronizability

The algorithm is based on the Lemma 1 and presents a modification of an algorithm from [4].
First let us check SCC using the first-depth search and find the SCC Γ_s of sink

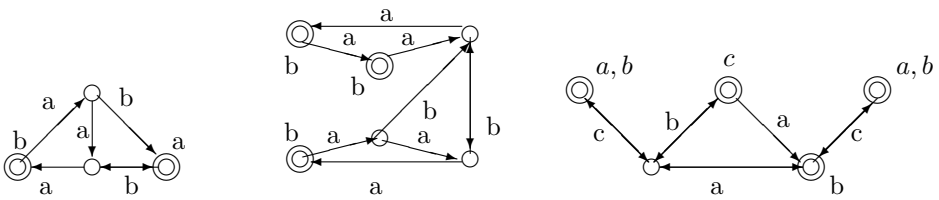
states from Γ. If there are no sink state then the graph has no synchronizing word and the algorithm stops. Exactly one sink state implies synchronizability and the algorithm also stops. The time and space complexity of these step are linear. Now we can consider the graph Γ_s with at least two sink states.
The next step is the consideration of Γ_s^2. We unite any pair of states $(\mathbf{p}, \mathbf{q})$ and $(\mathbf{q}, \mathbf{p})$, all states $(\mathbf{r}, \mathbf{r})$ are united in one state $(0, 0)$. Then let us mark sink state $(0, 0)$ and all ancestors of $(0, 0)$ using the first-depth search on the reverse of the obtained graph G. The graph Γ is synchronizing if any node of G will be marked. The time and space complexity of the algorithm in the most worst case is $O(|\Gamma|^2 q)$.

3 Experimental data

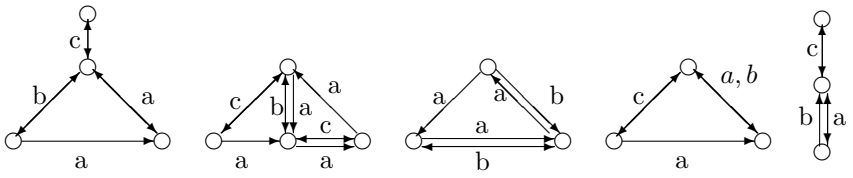
The considered synchronization algorithms were used in a program for search of automata with minimal reset word of relatively great length. The program has investigated all complete DFA for $n \leq 10$, $q = 2$ and for $n \leq 7$, $q \leq 4$.
An automaton with k states outside sink SCC A of the transition graph can be mapped on A by word of length not greater than $k(k-1)/2$. Therefore only automata with strongly connected transition graphs need investigation. The graphs with synchronizing proper subgraph obtained by moving off letters from the alphabet are omitted too. The program reduced also the number of studied isomorphic copies of automata. The case of $n = 2$ is not considered because any synchronizing automaton with two states has reset word of length $(n-1)^2 = 1$.
The known n-state automata with minimal reset word of length $(n-1)^2$ are presented by sequence of Černy [2] (here n=28):



by automata supposed by Černy, Piricka and Rosenauerova [3] (CPR), by Kari [6] and Roman [14].



Our program has found five new following examples on the border $(n-1)^2$. The loops of the complete graphs are omitted here for simplicity.



The corresponding reset words of minimal length are: *abcacabca*, *acbaaacba*, *baab*, *acba*, *bacb*. All considered algorithms have found the same reset word for every example. The size of the syntactic semigroup found by the package TESTAS is 148, 180, 24, 27 and 27 correspondingly.

No doubts that some automata from this list, especially for $n = 3$, were sometimes studied by specialists, but we have not found any mention of.

There are no contradictory examples for the Černy conjecture in considered class of automata. Moreover, the program does not find new examples of automata with reset word of length $(n-1)^2$ for $n > 4$ and $q > 3$.

And what is more, the examples with minimal length of reset word disappear even for values near the Černy bound $(n-1)^2$ with growth of the size of the automaton. The gap appears for $n = 6$. There are no 6-state automata with minimal length of synchronizing word equal to 24 for $q \leq 4$.

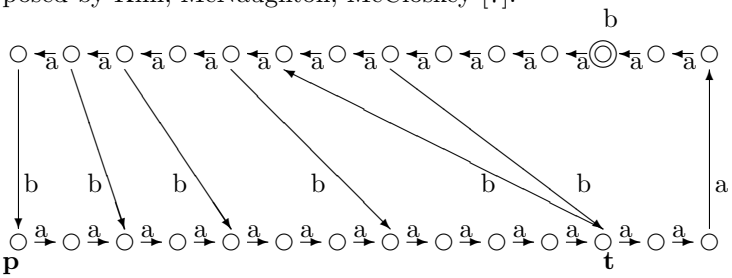
The following table displays this noteworthy trend for the maximum of lengths of minimal reset words. The mentioned above examples on the Černy border are not taken in account in the third line of the table.

size	n=5 $q <= 4$	n=6 $q <= 4$	n=7 $q <= 4$	n=8 q=2	n=9 q=2	n=10 q=2
$(n-1)^2$	16	25	36	49	64	81
max length	15	23	32	44	58	74

The gap between $(n-1)^2$ and the length of the minimal reset word grows with n. This growing gap supports the following funny

Conjecture 1 *The set of n-state DFA ($n > 2$) with minimal reset word of length $(n-1)^2$ contains only the sequence of Černy and the eight automata mentioned above, three of size 3, three of size 4, one of size 5 and one of size 6.*

Let us consider the synchronization algorithms from the package TESTAS on some above-mentioned objects and on a modification [16] of a graph KMM supposed by Kim, McNaughton, McCloskey [7].



Complete closure KMML of this graph is obtained from KMM by adding loops in all necessary cases. The n-state automata supposed by Černy will be denoted

by C$< n >$.
The following table presents the name of the automaton, the number of its states, the size of the syntactic semigroup, the length of synchronizing word found by the Eppstein algorithm [4], by the cycle and the semigroup algorithm, by the minimal synchronizing word algorithm with the corresponding number of mappings of the set of states.

name	CPR	Roman	Kari	C6	C9	C17	KMM	KMML	C28	C151
graph size	4	5	6	6	9	17	28	28	28	151
semigroup size	145	1397	17265	2742	218718	huge	22126	$> 10^6$	huge	huge
Eppstein alg	9	17	26	27	78	375	4	51	1202	57190
cycle algorithm	9	18	27	25	64	256	4	57	729	22500
semigroup alg	9	17	27	25	64	256	4	27	729	22500
minimal length	9	16	25	25	64	256	4	27	729	22500
mappings	9	22	46	56	501	131053	12	41035	vast	vast

One can compare the results of the algorithms. Equality of the length of minimal synchronizing word and of synchronizing word found by the semigroup algorithm and by Eppstein or cycle algorithm holds in some cases. In particular, it's true even for such extreme objects as Černy automata. Moreover, we obtain not infrequently the same synchronizing words. The transition semigroup of the Černy automaton has a nilpotent element of order $n-1$, and the minimal synchronizing word of the automaton is a subword of a power of this element.
As for the size of the syntactic semigroup from the table, the most discouraging example gives us the Kari automaton. The size of the syntactic semigroup of the Černy automaton is very great too, it is about $O(2^{2n})$. Maximal size n^n of the syntactic semigroup is reached for the examples of $n = 3$, $q = 3$. It is the semigroup of all transformations of 3-element set.

References

1. D. S. Ananichev, A. Cherubini, M.V. Volkov, An inverse auromata algorithm for recognizing 2-collapsing words. Springer, Lect. Notes in Comp. Sci., 2450(2003),270-282
2. J. Černy, Poznamka k homogenym eksperimentom s konechnymi automatami, Math.-Fyz. Čas., 14(1964) 208-215.
3. J. Černy, A. Piricka, B. Rosenauerova, On directable automata, Kybernetika 7(1971), 289-298.
4. D. Eppstein, Reset sequences for monotonic automata. SIAM J. Comput., **19**(1990) 500-510.
5. P. Frankl, An extremal problem for two families of sets, Eur. J. Comb., 3(1982) 125-127.
6. J. Kari, A counter example to a conjecture concerning synchronizing word in finite automata, EATCS Bulletin, 73(2001) 146-147.
7. Kim S., McNaughton R., McCloskey R. A polynomial time algorithm for the local testability problem of deterministic finite automata, IEEE Trans. Comput., N10, 40(1991) 1087-1093.

8. A.A. Kljachko, I.K. Rystsov, M.A. Spivak, An extremely combinatorial problem connected with the bound on the length of a recurrent word in an automata. Kybernetika. 2(1987) 16-25.
9. Z. Kohavi, J. Winograd, Establishing certain bounds concerning finite automata, J. Comp. System Sci., 7(1973), 288-299.
10. D. Lee, M.Yannakakis, Principle and methods of testing finite state mashines - A survey, Proc. of IEEE, 8, 84(1996) 1090-1123.
11. B.K. Natarajan, An algorithmic approach to the automated design of parts orienters. Proc. of 27th Annual Symp. Foundations of CS, IEEE, 1986, 132-142. Springer, Lect. Notes Comp. Sci., 62(1978) 345-352.
12. J.-E. Pin, On two combinatorial problems arising from automata theory, Annals of Discrete Math., 17(1983) 535-548.
13. J.-K. Rho, F. Somenzi, C. Pixley, Minimum Length Synchronizing Sequences of Finite State Machine, Proc. of 30th ACM/IEEE DA Conf., 1993, 463-466.
14. A. Roman, A note on Cerny Conjecture for automata with 3-letter alphabet (submitted).
15. A. Salomaa, Generation of constants and synchronization of finite automata, J. of Univers. Comput. Sci., 8(2) (2002), 332-347.
16. A.N.Trahtman, Optimal estimation on the order of local testability of finite automata. Theoret. Comput. Sci., 231(2000) 59-74.
17. A.N. Trahtman, Verification of algorithms for checking some kinds of testability. In Algebraic Methods in Language Processing, TWLT 21, eds. F.Spoto, G. Scollo, A. Nijholt. 2003, 253-263.
18. A.N. Trahtman, Černy conjecture for DFA accepting star-free languages. ICALP, Workshop on synchronizing automata, Turku, Finland, 2004.
19. A.N. Trahtman, Some results of implemented algorithms of synchronization. 10-th Journees Montoises d'Inform. Theor., LIege, Belgia, 2004.