SIAM J. DISCRETE MATH.
Vol. 26, No. 1, pp. 177–192

THE SYNCHRONIZING PROBABILITY FUNCTION OF AN AUTOMATON*

RAPHAËL M. JUNGERS†

Abstract. We study the synchronization phenomenon for deterministic finite state automata and the related longstanding Černý conjecture. We formulate this conjecture in the setting of a two-player probabilistic game. Our goal is twofold. On the one hand, the probabilistic interpretation is of interest in its own right and can be applied to real-world situations. On the other hand, our formulation makes use of standard convex optimization techniques, which appear powerful to shed light on Černý's conjecture. We analyze the synchronization phenomenon through this particular point of view. Among other properties, we prove that the synchronization process cannot stagnate too long in a certain sense. We propose a new conjecture and demonstrate that its validity would imply Černý's conjecture. We show numerical evidence for the pertinence of the approach.

Key words. synchronizing automata, Černý's conjecture, probabilistic method, linear programming, autonomous agents localization

AMS subject classifications. 68R05, 68R10, 90B15, 68Q45, 05D40

DOI. 10.1137/100816109

1. Černý's conjecture.

A (deterministic, finite state, complete) automaton is a set of m row-stochastic matrices $\Sigma \subset \{0,1\}^{n\times n}$ (where m, n are positive integers). That is, the matrices in Σ have binary entries, and they satisfy $A\mathbf{e} = \mathbf{e}$, where $\mathbf{e}$ is the all-ones (column) vector. We write Σ^t for the set of matrices which are products of length t of matrices taken in Σ. For convenience of product representation, to each matrix $A_c \in \Sigma$ is associated a letter c such that the product $A_{c_1} \dots A_{c_t} \in \Sigma^t$ can be written $A_{c_1 \dots c_t}$.

It is convenient to look at an automaton in terms of a discrete time dynamical system, where an agent moves on a graph. In this interpretation, at each time step, the m matrices are possible candidates for an adjacency matrix of a graph on n vertices, and this adjacency matrix can change from time to time. If one specifies a sequence of $T \in \mathbb{N}$ letters $c_1 \dots c_T$, and a starting node for the agent (say, $v_{i_0} : 1 \le i_0 \le n$), there is a single corresponding path $v_{i_0} - v_{i_1} - \dots - v_{i_T}$ such that the entry $(i_{t-1}, i_t)\,(1 \le t \le T)$ of the matrix A_{c_t} is equal to one (this is because the matrices are stochastic). Thus, if one knows the initial vertex v_{i_0} and the sequence of matrices $c_1 \dots c_T$, the last vertex of the path is given by the product $e_{i_0}^T A_{c_1 \dots c_T}$, where the kth standard basic vector e_k represents the fact that the agent is in vertex k.

Now, imagine that the position of the agent is not known, but one is allowed to choose the succession of letters $c_1, c_2 \dots$ so that he has some control on the trajectory of the agent. An automaton is said to be *synchronizing* if it is possible to drive the agent to a fixed position and localize it.

DEFINITION 1. *An automaton* $\Sigma \subset \{0,1\}^{n\times n}$ *is* synchronizing *if there is a finite product* $A = A_{c_1} \dots A_{c_T} : A_{c_i} \in \Sigma$ *which satisfies*

$$A = \mathbf{e} e_i^T,$$

*Received by the editors November 29, 2010; accepted for publication (in revised form) November 15, 2011; published electronically February 16, 2012. This research was supported by FRS-FNRS and BAEF.

http://www.siam.org/journals/sidma/26-1/81610.html

†ICTEAM Institute, Université catholique de Louvain, 1348 Louvain-la-Neuve, Belgium (raphael.jungers@uclouvain.be).

where $\mathbf{e}$ *is the all-ones vector and* e_i *is the* i*th standard basis vector. In this case, the sequence of letters* $c_1 \dots c_T$ *is said to be a* synchronizing word.

Thus, if an automaton is synchronizing, one can drive an agent to a fixed node, without a priori knowing its position, just by applying a synchronizing word. Synchronizing words, which are also sometimes called reset sequences, have appeared independently in many different communities and times, due to their very natural motivation. Synchronizing automata have applications in theoretical computer science, biocomputing, robotics, etc. (see [20] for a recent survey). They were defined in 1964 [9] and have led since then to a huge literature. They are recognizable in polynomial time but the shortest synchronizing word of a given synchronizing automaton is NP-hard to compute [4, 12]. They are related to the famous road-coloring conjecture of Adler and Weiss [1], which has been recently solved by Trahtman [18]. The main open problem on synchronizing automata is undoubtedly the following one.

CONJECTURE 1 (Černý's conjecture, 1964 [10]). *Let* $\Sigma \subset \{0,1\}^{n\times n}$ *be a synchronizing automaton. Then, there is a synchronizing word of length at most* $(n-1)^2$.

In [9], an infinite sequence of automata on n vertices (and containing two matrices): $C_n \subset \{0,1\}^{n\times n}$, $n = 1, \dots,$ is proposed that exactly require $(n-1)^2$ time steps to synchronize. These automata have a simple structure: A_a is the identity matrix, except that the last row is e_1, while $A_{b(i+1,j+1)} = 1$ iff $j = i+1 \mod n$ for $0 \le i, j \le n-1$ (see Figure 1(d)). Let us mention that except for these, very few synchronizing automata are known that necessitate so many steps to synchronize. We call the automata C_n the *Černý automata*.

Černý's conjecture has been proved in many particular cases (see, for example, [2, 3, 8, 9, 11, 12, 14, 17]) but is still open in its general formulation. It has been the subject of intense research for several decades, and we quote M. Volkov: "this simply looking conjecture is arguably the most longstanding open problem in the combinatorial theory of finite automata" [20].

Until now, the best upper bound on the length of a minimal synchronizing word for an automaton of size n is not quadratic but is equal to $(n^3-n)/6$ [15].[1] Recently, some attemps to introduce a probabilistic point of view to this problem have appeared in the literature. (See [16] for a recent presentation of the main ideas.) In the following we also introduce probabilistic ideas. However, this does not seem to connect directly to the above mentioned approaches, as these approaches put probabilities on the matrix to choose, while we introduce probabilities on the nodes of the graph.

In the remainder of this paper, we first (in section 2) introduce the mathematical object we want to study in this paper, which we call the *synchronizing probability function of an automaton*. Then in section 3 we analyze this function. Among other things, we show that this function must increase regularly in some sense. We hope that this may open new opportunities for a proof of Černý's conjecture. In section 4 we analyze numerical computations motivated by our analysis and present conjectures based on our observations. We prove that the main conjecture implies Černý's conjecture. In section 5, we conclude and state a few remarks on our approach. In Figure 1, the reader will find representations of a few automata that are studied in this paper.

2. The synchronizing probability function. Our starting idea is to twist the notion of synchronizing automaton by looking at its interpretation in terms of an agent moving on a graph and whose position is not known exactly. It is natural to

[1] While the present paper was under review, this 30-year-old bound was reduced to $n(7n^2+6n-16)/48$ in the preprint [19].

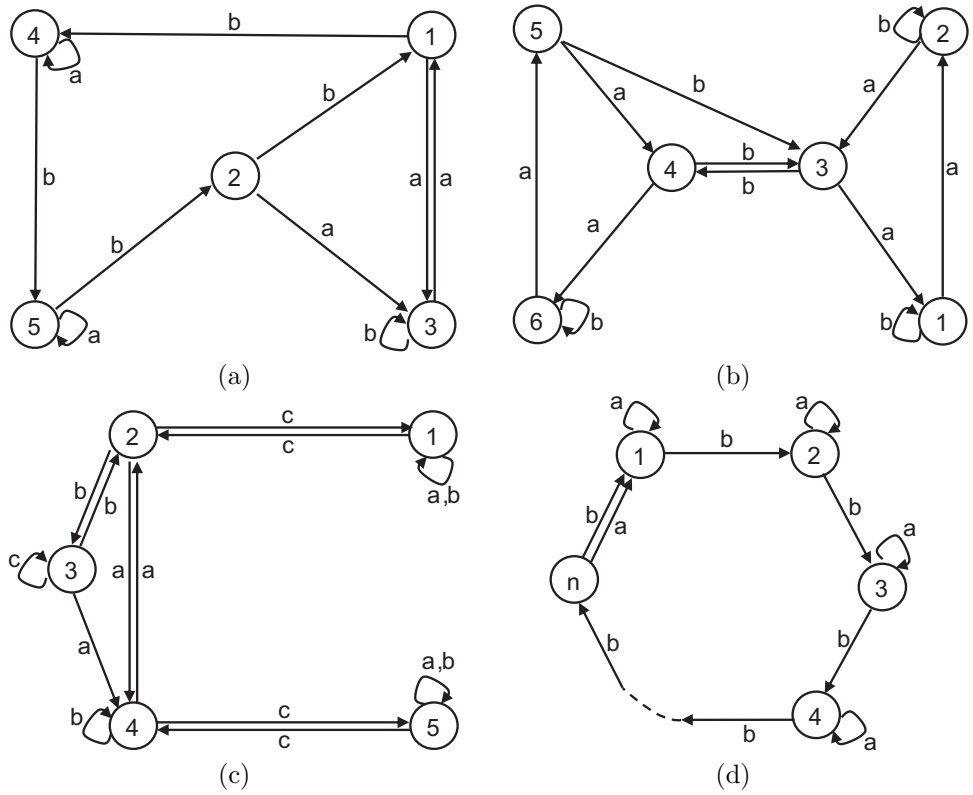

FIG. 1. *A few automata with slowly increasing synchronizing probability function, which are studied in this paper.* (a) *An automaton on* 5 *nodes.* (b) *Kari's automaton.* (c) *Roman's automaton.* (d) *Černý's family of automata.*

introduce a vector p of probability density on the set of nodes, which represents the possible positions of the agent. In the classical setting of synchronizing automata, postmultiplying a vector with matrices in Σ allows one to improve his knowledge on the agent's position. The probabilistic natural counterpart is that one modifies the probability distribution on the nodes.

However, this probability is not specified as an instance of the problem, and in order to define it, we think of the situation as a two-player game. In this game, the first player tries to catch the second one, which is hidden in the graph. The *policy of the second player* is defined as a probability distribution on the nodes, that is, any vector $p \in \mathbb{R}^{+n}$, $\mathbf{e}^T p = 1$. This agent starts in node i with probability p_i, and will then end up in the node corresponding to $e_i^T A$, where A is the matrix that the first player will choose. Since the first player wants to maximize the probability to catch player two, he will pick up the node where the probability for player two to be is maximal, that is,

$$\operatorname{argmax}_l (p^T A)_l.$$

So, the probability that player two will be caught is

$$\max_{l,A} ((p^T A)_l). \tag{1}$$

Obviously, player two wants to minimize that probability. Thus, we introduce the mathematical object we want to study: the *synchronizing probability function* of the automaton Σ. In the following, $\Sigma^{\leq t}$ is the set of products of length at most t of matrices taken in Σ. By convention, and for the ease of notation, it contains the product of length zero, which is the identity matrix.

DEFINITION 2 (synchronizing probability function). *Let $n \in \mathbb{N}$ and $\Sigma \subset \{0,1\}^{n\times n}$ be an automaton. The* synchronizing probability function *of Σ is the function*[2] $k_\Sigma : \mathbb{N} \to \mathbb{R}^+$:

$$k_\Sigma(t) = \min_{p\in\mathbb{R}^{+n},\ \mathbf{e}^T p=1} \left\{ \max_{A\in\Sigma^{\leq T}} \{\max_l (p^T A)_l\} \right\}. \tag{2}$$

We fix by convention $\Sigma^0 = \{I\}$, and this implies that for any automaton, $k(0) = 1/n$. In a general setting, the first player might well make use of a probabilistic policy: for a given automaton Σ and a fixed length t, we define *the probabilistic policy* π of player one as a set of s triples (where s is the number of different choices in the policy):

$$\pi = \{(w_i, v_i, q_i) : 1 \leq i \leq s\}, \tag{3}$$

where w_i are words of length t or less on the alphabet of the automaton, v_i is the index of a node, and q_i is the probability for this particular choice (w_i, v_i) to be chosen by player one (and thus $\sum q_i = 1$). In other words, the first player selects a couple (w_i, v_i) with probability q_i. He then applies the sequence of matrices given by w_i and picks up node v_i.

The following proposition is obvious.

PROPOSITION 1. *Conjecture* 1 *is equivalent to the following conjecture. Let $\Sigma \subset \{0,1\}^{n\times n}$ be a synchronizing automaton. Then,*

$$\forall t \geq (n-1)^2, \quad k_\Sigma(t) = 1.$$

Proof. In (2), the minimum is equal to one iff there is a matrix in $\Sigma^{\leq T}$ that has a column whose entries are all equal to one, which means precisely that the automaton is synchronized. ∎

To the best of our knowledge, this probability function has never been looked at in the literature. There has recently been some attempt to look at synchronizing automata with a probabilistic reasoning; see, for instance, [16]. However, in that reference, only the matrices are chosen following to a certain probability distribution, and thus it does not seem to directly connect with our approach.

We hope this function will act as a sort of Lyapunov function in order to prove Černý's conjecture. As shown in Figures 2, 3, and 4, the function seems to increase quite regularly. So, suppose (for instance) that one proves that for all t such that $k(t) < 1$, $k(t+n-1) - k(t) \geq 1/n$; then the conjecture would be proved, because $k(0) = 1/n$. We show below that this function has many appealing properties and seems to accurately represent the synchronizing phenomenon. The intuitive reason

[2]There are several possible choices for the set of matrices that player one can apply. We choose this one, which seems the most appropriate for proving results easily: all the matrices in $\Sigma^{\leq t}$. In terms of the game interpretation, this means that we do not impose player one to apply a product of length exactly t, but rather t is only an upper bound on the length of the product that player one can choose. We prove below that choosing $\Sigma^{\leq t}$ instead of Σ^t does not affect the value of the function.

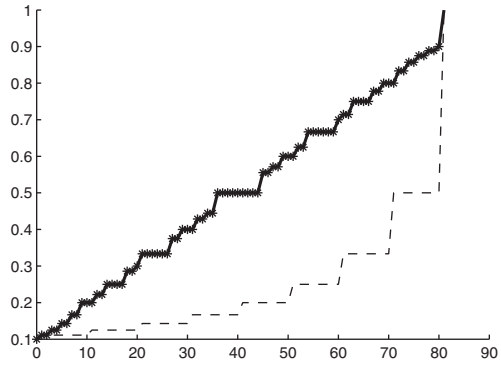

FIG. 2. *The function $k(t)$ for the automaton C_{10} (solid curve and stars). The dashed curve is the inverse of the minimal number of nonzero columns in a product of length t. For some automata, this latter curve does not grow regularly at all, which is perhaps part of the reason why a proof of Černý's conjecture is hard to find. Throughout the paper, we use stars in our figures to refer to the value taken by the Černý automaton, which we take as a reference value, being the slowest known synchronizing behavior.*

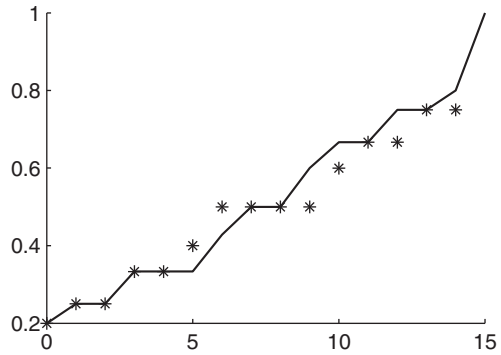

FIG. 3. *The function $k(t)$ represented for the automaton* (a) *of Figure* 1, *which is an automaton on* 5 *nodes. In the case of slow growths, as is the case for this particular automaton (the synchronizing time is* 15, *while the conjectured maximum is* 16*), the function grows very much like for Černý's automaton (represented by the stars), but up to small variations.*

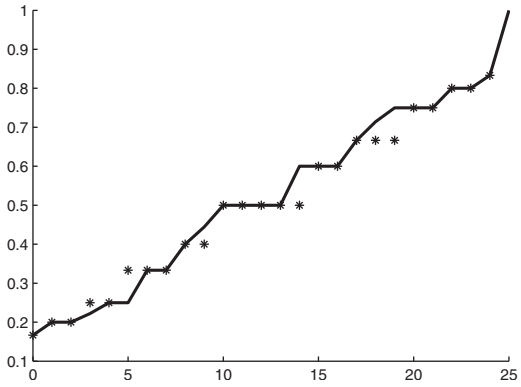

FIG. 4. *The function $k(t)$ for the Kari automaton (defined in Figure* 1(b)*).*

for the good behavior of this function is that it takes precisely into account the evolution of the matrix semigroup when the length of the products increases, as we now explain. Suppose indeed that player two chooses his probability function p more naively. Then of course the score of player one can be higher. For instance, it might seem that a good strategy for player two is to hide in each state with equal probability (i.e., $p = \mathbf{e}/n$). However, this might be an inadequate choice. This is the case, for instance, for the automaton in Figure 1(d) for $t = 1$. Indeed, applying matrix A_a, player one could realize a score equal to $2/n$, since $\max p^T A_a = 2/n$. This is highly suboptimal for player two: if, on the contrary, he hides in every node but the first one with probability $1/(n-1)$, then the probability of being caught drops to $1/(n-1)$ at most, whatever policy player one adopts. Finally, note that the optimal probability distribution can change with t, and, for instance, at time $t = n$, the policy $p = \mathbf{e}/n$ actually becomes optimal (but only at that precise time).

In fact, this particular choice of $p = \mathbf{e}/n$ is important in practice, since with this choice, the best strategy for player one is to apply the column of a matrix in $\Sigma^{\leq t}$ with the largest possible weight (i.e., the largest number of ones). This can in turn be put in relation with a popular method in the literature for designing synchronizing sequences, known as the "extension method." In matrix terms, the idea of this method is to find products with columns of increasingly larger weights, starting with a column of weight two. The method first chooses an arbitrary index $1 \leq i \leq n$ and then works iteratively: if one has a product (say, A_w) such that $\mathbf{e}^T A_w e_i = k$, then he tries to look for a product A_u such that $\mathbf{e}^T A_u A_w e_i = k + 1$. For several particular families of synchronizing automata, one is able to show that such a word u always exists, whose length is smaller than n. It is obvious that in this case Černý's conjecture then holds, because after $(n-2)$ steps the product constructed must contain a column which is the all-ones vector. This product has then a length at most $(n-1)^2$.

However, the extension method is known to be suboptimal in several cases: if one builds a product in this way, it may well have a larger length in the end than the shortest synchronizing product. The reason is that trying to increase the weight of a column in a greedy manner, one does not look to the long-term optimum, and then, after a few steps, the only available products that still can increase the weight of a column may be too long. This is the case, for instance, for a family of automata (the "Berlinkov Automata" [5]), which we analyze below.

On the other hand, the synchronizing probability function tries to find a synchronizing word in a much more careful way, since no information is assumed on the initial probability distribution. This forces player one to be more careful and to keep more than one product. We believe that the requirement of being able to gather a certain probability *whatever the initial distribution was,* rather than just assuming that the initial distribution was homogeneous, is critical. In this paper we show theoretical as well as numerical arguments in this direction.

We end this section by formulating the optimization problems of player one and player two as linear programming problems. The theory of linear programming allows us to prove that both these problems have the same value, which is coherent with the intuition that there must be a unique probability that player one localizes player two if both of them play optimally. The theory of linear programming (see [7] for a survey) enables us to prove many appealing properties for the synchronizing probability function but, except for the theorem below, we will prove all the results from scratch for the sake of clarity and in order to ease the intuition on this function. From now on, we note $\mathbf{e}$ for the all-ones column vector without explicitly stating its dimension if it is clear from the context.

THEOREM 1. *The* synchronizing probability function $k_\Sigma(t)$ *of* Σ *is given by*

$$\begin{aligned} \min_{p} \quad & k \\ s.t. \quad & p^T B \le k\mathbf{e}^T \quad \forall B \in \Sigma^{\le t} \\ & \mathbf{e}^T p = 1 \\ & p \ge 0. \end{aligned} \tag{4}$$

It is also given by the solution of

$$\begin{aligned} \max_{q} \quad & k \\ s.t. \quad & Aq \ge k\mathbf{e} \\ & \mathbf{e}^T q = 1 \\ & q \ge 0, \end{aligned} \tag{5}$$

where $A = A(t)$ *is the* $n \times M(t)$ *block-row matrix with all the matrices in* $\Sigma^{\le t}$, *and* $M(t) = nm^t + nm^{t-1} + \cdots + n$.

Proof. It is straightforward to show that the programs (4) and (5) are the dual of each other. Since they both admit a feasible solution, their optima must be equal by the well-known duality theorem of linear programming [6, section 4.3]. □

The dual formulation (5) represents the point of view of player one. It shows that, in general, he has to randomize in order to ensure the optimality of his policy: if q_j corresponds to the ith column of the product $A_w \in \Sigma^{\le t}$, it represents the probability with which player one will choose this product together with node v_i. Thus, it corresponds precisely to the triple (w, v_i, q_j) in the description of its policy as in equation (3).

3. Study of the function $k(t)$. We now analyze the above described game. Some of the following results can be derived from classical optimization theory results, but we tried to present self-contained arguments. All these results are promising in view of a proof of Černý's conjecture. For instance, the first result shows that for $t = 0, 1, 2, 3, 4$, the discrete derivative of $k(t)$ is at worst more or less equal to $1/(n-1)^2$. If the function keeps increasing at this rate until $k(t) = 1$, then Černý's conjecture is true. Also, item 5 shows that at the last step of the synchronization process, the discrete derivative, is large.

PROPOSITION 2. *For any synchronizing automaton,*

1. $k(0) = 1/n$,
2. $k(1) \ge 1/(n-1)$,
3. $k(3) \ge 1/(n-1.5)$,
4. $k(4) \ge 1/(n-2)$, *and*
5. $k(t) < 1 \quad \Rightarrow \quad k(t) \le (n-1)/n$.

Proof.

1. Since $\Sigma^0 = \{I\}$, the solution $p = \mathbf{e}/n$, $k = 1/n$ is a feasible solution for (4), which shows that $k(0) \le 1/n$. On the other hand, $q = \mathbf{e}/n$, $k = 1/n$ is a feasible solution for (5), which shows that $k(0) \ge 1/n$.
2. Let us denote $A \in \Sigma$ any matrix in Σ which has a zero column. Then, taking $q_i = 1/(n-1)$ for the variables in (5) corresponding to the other columns of A, we obtain a feasible solution with $k = 1/(n-1)$.
3. At $t = 3$, the block-row matrix $A(3)$ (i.e., the set of columns of all the products of length three or less) has at least three different columns of weight two, or

one column with weight at least three. Indeed, at every step t, there must be at least one new column in $A(t)$. We now provide a feasible solution in each of these two cases for the linear program (5), yielding a lower bound on $k(t)$. In the first case (i.e., $A(3)$ has no column of weight three or more), if there are two columns with in total four different entries equal to one, we give a coefficient $1/(n-2)$ to these columns and also to all the unit vectors e_i such that v_i does not correspond to any of those four entries. If, on the other hand, the three columns share only three different nonzero entries in a symmetric way, we give them a coefficient $1/(2n-3)$, and we give $2/(2n-3)$ to the other unit vectors.
The only remaining possibility for the case where $A(3)$ has no column of weight three or more is that all columns of weight two have a common nonzero entry (say, the first one). We show by contradiction that this is impossible in a synchronizing automaton. Indeed, at $t=2$, there are at least two different such columns of weight two, which implies that $p^* = (0, 1/(n-1), \dots, 1/(n-1))$ is the only solution to (4) and $k(2) = 1/(n-1)$. Also, if all columns of weight two in $A(3)$ have the first entry equal to one, we have that

$$p^{*T} A_c A(2) \leq 1/(n-1)$$

for any $A_c \in \Sigma$ (because the columns in $A_c A(2)$ are columns in $A(3)$). Thus, $p^{*T} A_c$ is equal to the only solution to (4), and we have that

$$\forall A_c \in \Sigma, p^{*T} A_c = p^{*T},$$

which implies that Σ is not synchronizing.
In the second case (i.e., $A(3)$ contains a column of weight at least three) we give a coefficient $1/(n-2)$ to the column of weight larger than three and to the other unit vectors.
Now, in all these situations, the corresponding vector Aq is (entrywise) larger than $\mathbf{e}/(n-1.5)$.

4. It is well known [15, Theorem 3.8] that for any synchronizing automaton, there is a product of four matrices with two zero columns. Giving the coefficient $q_i = 1/(n-2)$ to all the other columns in the product, one gets $k(4) \geq 1/(n-2)$.
5. Note that $k(t) < 1$ implies that every column in $A(t)$ has at least one zero. Thus,

$$(\mathbf{e}/n)^T k \mathbf{e} \leq (\mathbf{e}/n)^T Aq = ((\mathbf{e}/n)^T A)q \leq ((n-1)/n)\mathbf{e}^T q = (n-1)/n. \qquad \square$$

The next proposition states that for any automaton Σ and integer t, the second player can make his policy public (provided it is optimal) without loosing optimality. That is, even if the first player knows the policy chosen by the second player, he cannot improve the probability to catch him. The same holds for the second player with the policy of the first.

PROPOSITION 3. *Denote $k_p(t)$ the greatest probability that player one can ensure if he knows that player two has chosen the policy p. If p corresponds to an optimal solution of* (4)*, then $k_p(t) = k(t)$.*

Denote $k_q(t)$ the smallest probability that player two can ensure if he knows that player one has chosen the policy q. If q corresponds to an optimal solution of (5)*, then $k_q(t) = k(t)$.*

Proof. Since p is an optimal solution of (4), for any policy q of player one,

$$k_p(t) = p^T A(t) q \leq k(t) \mathbf{e}^T q = k(t).$$

The proof of the other statement is similar. □

PROPOSITION 4. *For any automaton Σ and integer t, there is always an optimal policy (as defined in* (3)*) for the first player with a number of columns smaller than or equal to n, where n is the number of nodes in the automaton.*

Proof. Let us suppose that all the entries of the optimal solution q^* are positive. In the other situation, we can just remove the zero entries and the corresponding columns in $A(t)$ without changing the optimum in (5).

Now, if there are more than n columns in A, the system

$$Aq' = 0 \tag{6}$$

has a nonzero solution.

Since this equation is homogeneous, we can scale the solution, and $\lambda q'$ is still a solution. Suppose without loss of generality that

$$\mathbf{e}^T q' \leq 0. \tag{7}$$

Then, taking

$$\lambda = \min_{(q'_i < 0)} \{q^*_i / (-q'_i)\}, \tag{8}$$

we obtain that $q^* + \lambda q'$ is a feasible solution. Indeed, from (7) we infer that $\mathbf{e}^T(q^* + \lambda q') \leq 1$. From (8) we infer that for all i, $(q^* + \lambda q')_i \geq 0$. Finally, $(q^* + \lambda q')$ is still an optimal solution, since (6) implies that $A(q^* + \lambda q') \geq k$. (If $\mathbf{e}^T(q^* + \lambda q') < 1$ one can of course increase a non-zero-entry of $(q^* + \lambda q')$ until the sum is equal to one, without losing optimality.)

Now $q^* + \lambda q'$ has a zero-entry, and we can remove the corresponding column without changing the optimum. The result follows by inductively removing columns until there are no more than n of them. □

PROPOSITION 5. *For any automaton Σ and integer t, the synchronizing probability function $k_\Sigma(t)$ remains the same if the set of matrices $\Sigma^{\leq t}$ is replaced by Σ^t in its definition* (4).

Proof. We prove it for the optimization problem (5). This proves the proposition since the optimal value is the same for (4) and (5). Since the feasible domain is smaller when $\Sigma^{\leq t}$ is replaced by Σ^t, it is clear that the optimal value decreases. We prove now that actually the same value remains feasible.

Suppose that the columns a_j, $j = 1, \ldots, J$, which are columns of products of length $t_j < t$ have a nonzero coefficient q_j in (5). We will find a product of length exactly t with a column which is greater than or equal to a_j (componentwise).

For each j, let i_j be the index of the column a_j in its corresponding matrix (i.e., $a_j = A_{:,i_j}$ for a matrix $A \in \Sigma^{t_j}$). Take any matrix $M \in \Sigma^{t-t_j}$ and any i such that $M_{i_j,i} = 1$. (Recall that the matrices are stochastic, so M and i exist.) Then, the product AM is of length t and its ith column is greater than or equal to a_j and we can then replace it in the optimal solution of (5). □

We state a last property that will be useful for the analysis of the synchronizing probability function. (These are the *complementary slackness* conditions of linear programming; see [6, 7].)

THEOREM 2. *For any set of matrices (represented by the block-row matrix A) and any couple of optimal solutions $(p^*(t), q^*(t))$ of the problems (4) and (5) we have*

- $q_j^* \cdot (k - (p^{*T}A)_j) = 0$ *for all* $1 \leq j \leq M(t)$ *and*
- $p_i^* \cdot ((Aq^*)_i - k) = 0$ *for all* $1 \leq i \leq n$.

Proof. Since $Aq^* - k\mathbf{e} \geq 0$ and $p^{*T}A - k\mathbf{e}^T \leq 0$, and also $p^*, q^* \geq 0$, we have

$$\begin{aligned} 0 &\leq p^{*T}(Aq^* - k\mathbf{e}) \\ &= p^{*T}Aq^* - k \\ &= (p^{*T}A - k\mathbf{e}^T)q^* \\ &\leq 0. \end{aligned}$$

Hence, since all the terms in $p^{*T}(Aq^* - k\mathbf{e})$ are nonnegative and all the terms in $(p^{*T}A - k\mathbf{e}^T)q^*$ are nonpositive, they must all be zero. ∎

In the remark below, we make a few observations on the algorithmic problem of computing the synchronizing probability function.

Remark 1.

- The synchronizing probability function depends only on the columns of the matrices in Σ^T and not on the matrices themselves.
- This shows that the different opportunities for player one at time t are simply represented by a set of columns, while the original formulation tended to indicate that the set of *matrices* $\Sigma^{\leq t}$ was relevant. The problem is in some sense decoupled.
- This allows us to design a fast method to compute the function $k(t)$: for each time $t = 1, \ldots$ compute the set $A(t)$ of columns that can be generated as the set $\{Ma : a \in A(t-1), M \in \Sigma\}$, starting with $A(0)$ the set of columns of the identity matrix. Then, trim the set $A(t)$ by just keeping the columns that are not majorated, and solve the (hopefully much smaller) linear problem.
- So, one can view this problem as involving a sequence H_t of hypergraphs (represented by a rectangular matrix whose columns are the hyperedges), and we can define an operator on hypergraphs, defined by the set Σ (here, if $\Sigma = \{A_0, A_1\}$)

$$H_t \to H_{t+1} : \quad H_{t+1} = A_0 H_t \cup A_1 H_t.$$

  Our goal is to show that if one hypergraph H_t contains the hyperedge $\mathbf{e}$, then it is the case for $t = (n-1)^2$. The results below go in that direction.

We define two polytopes that give some insight on the combinatorial structure of the problem.

DEFINITION 3. *Let Σ represent an automaton and t be a positive integer. The polytopes P_t and Q_t are the sets of optimal solutions of, respectively,* (4) *and* (5).

The following lemma and what follows formalize the rough idea that if k remains constant, something must evolve in order to increase k in the end. Below, for a set of vectors P and a set of matrices Σ, $P^T\Sigma$ represents the set $\{v^T A : v \in P \text{ and } A \in \Sigma\}$.

LEMMA 1. *If $k(t) = k(t+1)$,*

1. $P_{t+1} \subset P_t$,
2. $\bigcup_{A \in \Sigma} A^T P_{t+1} \subset P_t$, *and*
3. *for all* $p \in P_t, \max(p) \leq k(t)$.

Proof. Only the second item is not trivial. Under the hypotheses, if $p \in P_{t+1}$, then for all $B \in \Sigma^{t+1}$, $p^T B \leq k\mathbf{e}^T$. Take now any $A \in \Sigma$. It follows that for any $C \in \Sigma^t$, $p^T AC \leq k\mathbf{e}^T$. Hence, $A^T p \in P_t$. ∎

In the following, we call a *critical column* any column a_j of the matrix A in the linear program (5) such that there is a solution q with $q_j > 0$.

LEMMA 2. *At any time t such that $k(t) < 1$, the dimension of P_t satisfies*

$$dim(P_t) \leq n - 2.$$

Proof. Take $k(t) < 1$, and consider the inequality $Aq \geq k\mathbf{e}$. At least two linearly independent columns in the matrix A are necessary to fulfill this inequality (this is because the column $\mathbf{e}$ is not available in A). By Theorem 2, these two constraints (say, a_1, a_2) satisfy

$$p^T a_1 = k, \quad p^T a_2 = k \qquad \forall p \in P_t$$

and $\dim(P_t) \leq n - 2$. ∎

We can now prove our main result.

THEOREM 3. *Let $\Sigma \subset \mathbb{R}^{n \times n}$ represent a synchronizing automaton and t be a positive integer. Then, if $k(t) < 1$,*

$$k(t + n - 1) > k(t).$$

Proof. Let us suppose that $k(t + 1) = k(t)$.

We define $A_c(t)$ to be the set of critical columns. Then there exists a vector $q > 0$ such that $A_c(t)q \geq k\mathbf{e}$. Thus, for all $M \in \Sigma$, $MA_c(t)q \geq k\mathbf{e}$ (because the transition matrices of an automaton are row-stochastic). As a consequence, for any column a of $A_c(t)$, and any $M \in \Sigma$, Ma is a critical column at time $t + 1$.

Define R_t to be the set of optimal solutions of (4) with the matrix $A_c(t)$ (obviously, $P_t \subset R_t$). Recall (Lemma 1) that

$$P_{t+1} \subset P_t.$$

We define

$$A'(t + 1) = A_c(t) \cup \{MA_c(t) : M \in \Sigma\}$$

and P'_{t+1} as the set of points $p \geq 0$, $\mathbf{e}^T p = 1$, such that $p^T A'(t + 1) = k\mathbf{e}$. Note that $P_{t+1} \subset P'_{t+1}$, since the columns in $A'(t + 1)$ are a subset of the critical columns in $A(t + 1)$. Also, $P'_{t+1} \subset R_t$ since the columns in $A_c(t)$ are a subset of columns in $A'(t + 1)$.

We first show that $P'_{t+1} \neq R_t$. Indeed, since $A'(t + 1)$ contains all the columns of $A_c(t)$ multiplied by a matrix in Σ, it is clear that

$$\forall A \in \Sigma, \forall p \in P'_{t+1}, A^T p \in R_t. \tag{9}$$

Supposing $P'_{t+1} = R_t$, the above equation implies that $B^T R_t \subset R_t$ for all $B \in \Sigma^s$, $s \geq 1$, which implies that Σ is not synchronizing. Indeed, this implies that for all $p \in R_t$, for all $B \in \Sigma^s$, $p^T B \leq k\mathbf{e}^T$.

So, there must be a matrix $M \in \Sigma$ and a column a_j of $A_c(t)$ such that $Ma_j \notin A_c(t)$. Again, since $MA_c(t)q \geq k\mathbf{e}$, Ma_j is a new critical column.

Now, by Theorem 2, the new critical column Ma_j is such that $p^T Ma_j = k$ for all $p \in P'_{t+1}$. Let H be the hyperplane represented by this constraint. Since $R_t \cap H \neq R_t$, and $R_{t+1} \subset P'_{t+1} \subset R_t \cap H$, it follows that $\dim(R_{t+1}) < \dim(R_t)$. Since the dimension of R_t is less than or equal to $n - 2$ (indeed, one can replace P_t with R_t without

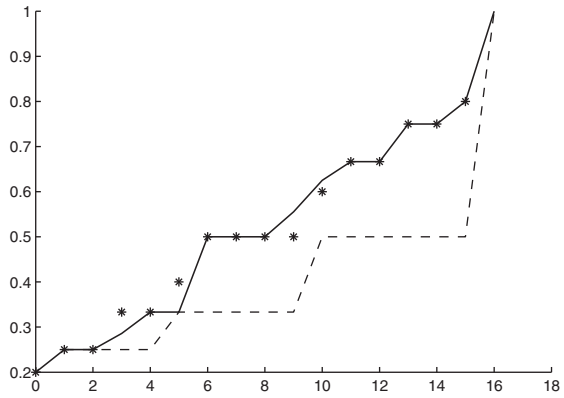

FIG. 5. *The function $k(t)$ for the Roman automaton (an automaton on five nodes which reaches the conjectured maximal number of steps). The dashed curve is the inverse of the minimal number of nonzero columns in a product of length t.*

changing the proof in Lemma 2), the dimension of R_{t+n-1} should be negative if k remained constant between t and $t+n-1$, and we have reached a contradiction. ∎

The theorem above seems to be promising: in classical upper bounds on the length of a synchronizing word [15], such a word of length smaller than n^3 is found because it is shown how to decrease the minimal number of nonzero columns in a product of length t by concatenating it with a product whose length is provably $O(n^2)$. Since one needs to decrease this number $O(n)$ times (from n to 1), one gets the $O(n^3)$ bound (visually it corresponds to the dashed curve in Figure 2). As seen in Figure 5, it is sometimes necessary to have a product of length $\Omega(n^2)$ (or at least more than n) to increase the curve, like in the last step in this example. In Theorem 3, we have a function that we can increment by concatenating products of length only $n-1$ at most, which lets us hope to get an overall bound of $(n-1)^2$ in the end.

4. A new conjecture on synchronizing automata. Due to numerical computations, we make the following conjecture.

CONJECTURE 2. *For any synchronizing automaton Σ and for any $j \geq 1$, $j \leq n-1$,*

$$k(1+(j-1)(n+1)) \geq j/(n-1). \tag{10}$$

In Figure 6 we represent the synchronizing probability function for another important family of automata introduced by Berlinkov [5].[3] The particularity of this family is that for some values of m, k, the extension method does not provide a satisfactory algorithm to find a shortest synchronizing word. Indeed, at some steps, one must wait as much as $2n-3$ steps to increase the maximal weight [5]. As shown in Figure 6, these automata respect Conjecture 2. Even for the value $(m,k) = (11,1)$, the function $k(t)$ increases slowly but always remains greater than the lower bound from Conjecture 2.

[3]For any couple $(m,k) \in \mathbb{N}^2$, the Berlinkov automaton $B(m,k)$ is an automaton on $m+k+1$ nodes $\{v_0, \ldots, v_{m+k}\}$ defined as follows: for all $i : 0 \leq i \leq m-1$, there is an edge with label a from v_i to v_{i+1}, and for all $i : m+1 \leq i \leq m+k-1$, there is an edge with label b from v_i to v_{i+1}. For all $i : m+1 \leq i \leq m+k$, there is an edge with label a from v_i to v_2. There is also an edge with label b from v_{m+k} to v_0, an edge with label b from v_0 to v_{m+1}, and an edge with label a from v_m to v_0. Finally, all the edges missing in the automaton described above are defined to be self-loops.

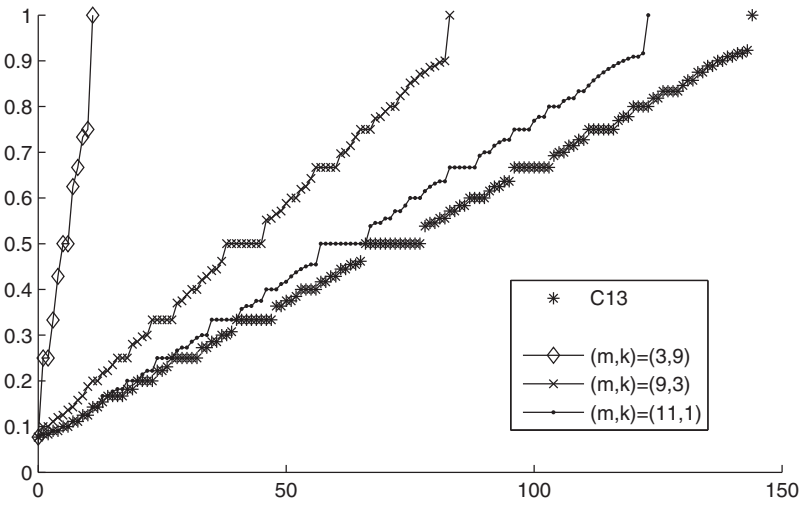

FIG. 6. *The function $k(t)$ represented for the Berlinkov automata $B(m,k)$ for a few values of m and k. For all these values, the automata have 13 nodes and are thus compared with the extremal automaton C_{13}.*

Note that Conjecture 2 is true for $j = 1$ (see Proposition 2). It appears that the general statement implies a positive answer to Černý's conjecture.

THEOREM 4. *Conjecture 2 is stronger than Conjecture 1.*

We split the proof into a few lemmas for clarity.

LEMMA 3. *Let A be a matrix in $\{0,1\}^{n\times s}$, $n \geq 3$, with at least one zero-entry in each column. If there is a nonnegative vector q, $q^T\mathbf{e} = 1$ such that*

$$Aq \geq \frac{n-2}{n-1}\mathbf{e}, \tag{11}$$

then there must exist such a q and a column $a_i : \mathbf{e}^T a_i = n-1$ with $q_i \geq 1/(n-1)$.

Proof. We fix l the number of different columns a_i of weight $n-1$ in A (i.e., $\mathbf{e}^T a_i = n-1$). Suppose first that $l \geq n-1$. Then, taking $n-1$ of these columns with a coefficient $q_i = 1/(n-1)$ does the job.

Suppose now by contradiction that $l < n-1$, and all these columns have a coefficient $q_i < 1/(n-1)$. Then,

$$\begin{aligned}
\mathbf{e}^T Aq &\leq \sum_{\mathbf{e}^T a_i = n-1} q_i(n-1) + \sum_{\mathbf{e}^T a_i \leq n-2} q_i(n-2) \\
&< \frac{l}{n-1}(n-1) + \frac{n-1-l}{n-1}(n-2) \\
&\leq \frac{l(n-1) - (l+1)(n-2)}{n-1} + \frac{n}{n-1}(n-2) \\
&\leq \frac{n-2}{n-1}n + \frac{l-(n-2)}{n-1},
\end{aligned}$$

a contradiction with (11) and the fact that $l \leq (n-2)$. ∎

LEMMA 4. *Let A be a matrix in $\{0,1\}^{n\times s}$, $n \geq 3$, with at least one zero-entry in each column. If there is a nonnegative vector q, $q^T\mathbf{e} = 1$ such that*

$$Aq \geq \frac{n-2}{n-1}\mathbf{e}, \tag{12}$$

then there exists such a q, together with at least $n-2$ different columns a_i in A such that $\mathbf{e}^T a_i = n-1$. For these columns, we have $q_i = 1/(n-1)$.

Proof. We prove the lemma by induction. It is obvious for $n = 3$.

We suppose in this proof that $q > 0$, as columns corresponding to $q_i = 0$ are irrelevant in the lemma. From Lemma 3, we can suppose without loss of generality that $q_1 \geq 1/(n-1)$, $a_1 = \mathbf{e} - e_1$ (i.e., the only zero entry of a_1 is the first one). Now, q_1 is actually exactly equal to $1/(n-1)$. Indeed,

$$(Aq)_1 \geq \frac{n-2}{n-1},$$

and this implies that

$$\sum_2^s q_i \geq \frac{n-2}{n-1}.$$

Moreover, this latter fact implies that $A_{1,i} = 1$ for all $i > 1$.

Then, denoting A', q' the matrix and vector obtained by removing the first row of A and q and the first column of A, we obtain a system in dimension $n-1$ such that

$$A'q' \geq \frac{n-3}{n-1}.$$

Multiplying this equation by $(n-1)/(n-2)$ and denoting by q'' the vector $((n-1)/(n-2))q'$, we get

$$A'q'' \geq \frac{n-3}{n-2}, \quad \mathbf{e}^T q'' = 1,$$

and we can apply the result by induction on (q'', A'). ∎

LEMMA 5. *Let Σ be a synchronizing automaton and t such that*

$$k(t) \geq \frac{n-2}{n-1}.$$

Then, $k(t+3) = 1$.

Proof. By Lemma 4 we can suppose without loss of generality that

$$a_i = \mathbf{e} - e_i, \quad 1 \leq i \leq n-2, \tag{13}$$

$$a_{n-1} \geq \mathbf{e} - e_{n-1} - e_n \tag{14}$$

are the only columns in $A(t)$ (where the last inequality is entrywise). By the proof of Theorem 3, $A(t+1)$ must contain a new column which is not majorated by any column in $A(t')$ for any $t' < t+1$. There are only two such columns at time t, which are not equal to $\mathbf{e}$. Thus, after three steps, the supplementary column must be $\mathbf{e}$, which implies that $k(t+3) = 1$. ∎

Proof of Theorem 4. Taking $j = n-2$ in (10), we obtain that

$$k((n-1)^2 - 3) \geq (n-2)/(n-1),$$

and Lemma 5 implies that $k((n-1)^2) = 1$. ∎

Taking $j = 2$ in Conjecture 2, we deduce two seemingly simpler conjectures, which are open to the best of our knowledge.

CONJECTURE 3. *For any synchronizing graph, $k(n+2) \geq 2/(n-1)$.*

In turn, there is another conjecture that is implied by the above. To see this we state an easy proposition.

PROPOSITION 6. *If all columns in $A(t)$ are of weight at most j, then $k(t) \leq j/n$.*

Proof. Let q be a solution of (5); we have

$$kn = k\mathbf{e}^T\mathbf{e} \leq \mathbf{e}^T A q \leq (j\mathbf{e}^T)q \leq j. \qquad \square$$

Now it is easy to see that Conjecture 3 implies the next one.

CONJECTURE 4. *For any synchronizing graph, there is a product of length $n+2$ that has one column with three ones.*

We do not have a proof for this simple statement, and to the best of our knowledge this problem is open. If it is the case, it may be worth looking at this seemingly much simpler problem.

5. Conclusion. In this paper, we have twisted the notion of synchronizing automaton, viewing it in the setting of a two-player game on an automaton. Beyond the possible real-life applications of this natural setting, our aim was to bring some understanding to the synchronization process, which is not well understood. The results presented in this paper go in that direction. More precisely, the synchronization process seems smoother when looking at the synchronizing probability function $k(t)$: we prove that this function cannot remain constant during more than $n-1$ steps.

Our experimental work based on the concepts introduced in this paper suggests ideas. Since the function $k(t)$ grows in a rather monotonous (and fast) way, it might lead to new methods for deriving upper bounds on the minimal length of a synchronizing word. Since the synchronization process looks very homogeneous and regular, the synchronizing probability function might be a useful tool to generate slowly synchronizing automata: by looking to $k(t)$ for the first few values of t, one could directly infer that the automaton synchronizes slowly or not.

Also, our approach allowed us to reformulate Černý's conjecture as a consequence of another conjecture (Conjecture 2) and to propose new simpler ones, which might help us better understand synchronizing automata.

If this synchronizing probability function does not appear powerful enough, one might think of modifications of this concept that could be of interest. For instance, the first player might want to minimize the entropy of the probability distribution of the second player on the nodes, rather than maximize the probability of catching him. We have preferred the latter approach in this paper for two reasons. First, even though the entropy approach is also representable as a convex program, with all the appealing properties that it implies, the problem is not representable as a linear program. So, the numerical simulations, as well as the theoretical results that can be derived, are less powerful. Second, from a few preliminary numerical tests, it seems that the corresponding "synchronizing entropy function" behaves much less regularly than the synchronizing probability function.

We have introduced a new way to look at the synchronization problem, which has many appealing features and properties. These features might allow for new ideas for tackling Černý's conjecture, and lead to many open questions.

Acknowledgments. This research was conducted while the author was at LIDS, MIT and has greatly benefitted from interactions within the institute. In particular, the author wishes to thank Michel Goemans and Steve Boyd for useful discussions.

All the conjectures and observations reported in this paper have been corroborated by a benchmark of synchronizing automata available from http://www.ii.uj.edu.

pl/~roman/cerny_experiments/results.txt. The author thanks Adam Roman for making this benchmark available. It has been most helpful for the research presented here.

The numerical experiments have been implemented with the help of `CVX`, a package for specifying and solving convex programs [13].

Finally, the author thanks the referees for providing constructive comments and help in improving the present paper. In particular, item 4 of Proposition 2 was kindly suggested by a referee.

REFERENCES

[1] R. L. Adler and B. Weiss, *Similarity of automorphisms of the torus*, Mem. Amer. Math. Soc., 98 (1970).

[2] D. S. Ananichev and M. V. Volkov, *Synchronizing generalized monotonic automata*, Theoret. Comput. Sci., 330 (2005), pp. 3–13.

[3] M.-P. Béal, M. V. Berlinkov, and D. Perrin, *A quadratic upper bound on the size of a synchronizing word in one-cluster automata*, Internat. J. Found. Comput. Sci., 22 (2011), pp. 277–288.

[4] M. V. Berlinkov, *Approximating the Minimum Length of Synchronizing Words is Hard*, Lecture Notes in Comput. Sci. 6072, 2010, pp. 37–47.

[5] M. V. Berlinkov, *On a conjecture by Carpi and D'Alessandro*, in DLT'10: Proceedings of the 14th International Conference on Developments in Language Theory, Springer-Verlag, Berlin, 2010, pp. 66–75.

[6] D. Bertsimas and J. N. Tsitsiklis, *Introduction to Linear Optimization*, Athena Scientific, Belmont, MA, 1997.

[7] S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge University Press, New York, 2004.

[8] A. Carpi and F. D'Alessandro, *The synchronization problem for locally strongly transitive automata*, in MFCS'09: Proceedings of the 34th International Symposium on Mathematical Foundations of Computer Science, Springer-Verlag, Berlin, 2009, pp. 211–222.

[9] J. Černý, *Poznámka k homogénnym eksperimentom s konečnými automatami*, Mat. Casopis SAV, 14 (1964), pp. 208–216.

[10] J. Černý, A. Pirická, and B. Rosenauerova, *On directable automata*, Kybernetica, 7 (1971), pp. 289–298.

[11] L. Dubuc, *Sur les automates circulaires et la conjecture de Černý*, RAIRO Inform. Theor. Appl., 32 (1998), pp. 21–34.

[12] D. Eppstein, *Reset sequences for monotonic automata*, SIAM J. Comput., 19 (1990), pp. 500–510.

[13] M. Grant and S. Boyd, *CVX: Matlab software for disciplined convex programming, version* 1.21, http://cvxr.com/cvx (April 2011).

[14] J. Kari, *Synchronizing finite automata on Eulerian digraphs*, Theoret. Comput. Sci., 295 (2003), pp. 223–232.

[15] J.-E. Pin, *On two combinatorial problems arising from automata theory*, Ann. Discrete Math., 17 (1983), pp. 535–548.

[16] B. Steinberg, *The averaging trick and the Černý conjecture*, in DLT'10: Proceedings of the 14th International Conference on Developments in Language Theory, Springer-Verlag, Berlin, 2010, pp. 423–431.

[17] A. N. Trahtman, *The Černý conjecture for aperiodic automata*, Discrete Math. Theoret. Comput. Sci., 9 (2007), pp. 3–10.

[18] A. N. Trahtman, *The road coloring problem*, Israel J. Math., 172 (2009), pp. 51–60.

[19] A. N. Trahtman, *Modifying the upper bound on the length of minimal synchronizing word*, Lecture Notes in Comput. Sci. 6914, Springer, 2011, pp. 173–180.

[20] M. V. Volkov, *Synchronizing automata and the Černý conjecture*, in LATA'08: Proceedings of the 2nd International Conference on Language and Automata Theory and Applications, Springer-Verlag, Berlin, 2008, pp. 11–27.