For submission to the Journal of Automata, Languages and Combinatorics
Created on February 4, 2019

# HARDLY REACHABLE SUBSETS AND COMPLETELY REACHABLE AUTOMATA WITH 1-DEFICIENT WORDS

François Gonze[(A)] Raphaël M. Jungers[(A,B)]

[(A)] *ICTEAM institute,*
*Université catholique de Louvain,*
*Avenue Georges Lemaitre 4, 1348, Louvain la Neuve*
francois.gonze@uclouvain.be raphael.jungers@uclouvain.be

ABSTRACT

This article focuses on subset reachability in synchronizing automata.

First, we study the length of shortest words reaching subsets of states in synchronizing automata. We provide an automata family with subsets that cannot be reached by words shorter than $2^n/n$, thus disproving a recent conjecture of Don. We then analyze relaxed versions of this conjecture.

Second, we analyze the $\Gamma_1$-graph construction. The $\Gamma_1$-graph is derived from 1-deficient words, and is a key tool for studying completely reachable automata. We introduce the concept of roots of 1-deficient words, which allows to state explicit concatenation rules for these words. Based on these results, we provide a polynomial-time algorithm for constructing the $\Gamma_1$-graph. Then, we disprove a conjecture by Bondar and Volkov linking the strong connectivity of this graph and the concatenation of 1-deficient words of completely reachable automata. Finally, we prove an alternative version of this conjecture.

*Keywords:* Subset reachability, short reaching word, completely reachable automata, $\Gamma_1$-graph.

## Introduction

Automata[1] are useful tools in various fields of applied mathematics. In pattern recognition, they allow to parse texts and efficiently find letter sequences. In language theory, they are the basic tools defining formal languages and context free languages. In theoretical computer science, automata provide simple models for the behaviour of computing devices. More recently, automata have also been at the core of synthesis

A preliminary version of this paper has been presented at the conference DLT 2018.

[(B)] R. M. Jungers is a F.R.S.-FNRS Research Associate. This work was supported by the French Community of Belgium and by the IAP network DYSCO.

[1] Formal definitions are provided in the next subsection.

and verification of complex automated systems. A general presentation of automata and their applications is provided by Berthé and Rigo [4], and by Linz [22].

Synchronization is an important topic in automata theory. Indeed, if a machine can be modelled as a synchronizing automaton, then it is possible to fix its state by applying a sequence of commands corresponding to a synchronizing word. It has direct applications in robotics [23], matrix theory [5], consensus theory [11] and group theory [1]. The most famous problem in this field was proposed by Jan Černý in 1964 [10]. It states that if an automaton has a synchronizing word, then it also has a short synchronizing word:

**Conjecture 1** (Černý's conjecture, 1964 [10])**.** *Let $A = (Q; \Sigma; \delta)$ be a synchronizing automaton with $|Q| = n$. Then, it has a synchronizing word of length at most $(n-1)^2$.*

In [9], Černý proposes an infinite family of automata attaining this bound, for any number of states. Fig. 1 represents the automaton of this family with four states. Many improvements have been achieved recently for some particular classes of automata (see [2, 13, 15, 20, 26, 28]), by improving the best general bound [27], or by proposing new probabilistic approaches (see [8, 19, 24]). However, Conjecture 1 is not proven yet in its general formulation. A survey on the subject is provided by Mikhail Volkov [29].

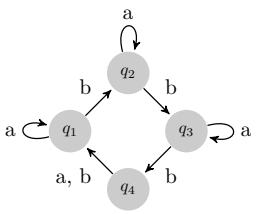


Figure 1: A synchronizing automaton attaining Černý's bound

Conjecture 1 can be adressed by the two natural and complementary approaches that are subset synchronization and subset reachability. In the subset synchronization approach, one considers the length of a shortest word which synchronizes a set $S$. It is studied in [16, 21, 30] among others. In the subset reachability approach, one considers the length of a shortest word reaching a set $S$. It is studied in [6, 12]. Subset reachability can be seen as a direct application of the extension method[2], which was used to prove Conjecture 1 for particular classes of automata (see [26] for a detailed analysis of this method). The algorithmic complexity of subset reachability problems is studied in [3]. Here, we focus on properties of subset reachability.

In Section 1, we study the length of shortest words reaching some subsets of states.

[2]In the extension method, one proves that any subset is the image of a larger set under the action of a short word. Then, starting with a set $S$ and extending it recursively, one obtains the set $Q$, and the concatenation of the extending words is a word reaching $S$. Therefore, starting with extending a single state with a single letter (such state always exist), this set can next be extended $n-2$ times to obtain $Q$. If each extending word in this process is not longer than $n$, it provides a synchronizing word of length smaller or equal to $n \times (n-2) + 1 = (n-1)^2$.

In [12], Don showed for a class of automata that for any reachable subset of $k$ states of any automaton of this class, there exists a word of length $n(n-k)$ reaching this subset. He also conjectured that it was true for any synchronizing automaton. First, we show that this conjecture does not hold by providing a family of counterexamples where the length of the shortest word reaching a set of $n-2$ states is quadratic. Second, we analyze alternative versions of the conjecture and the questions they raise. In particular, we build a family of automata such that a set of $\lfloor \frac{n}{2} \rfloor$ states cannot be reached by a word shorter than $2^n/n$.

In Section 2, we study the construction of the $\Gamma_1$-graph and its link with *completely reachable automata*. An automaton is said to be completely reachable if any of its proper subsets is reachable. The $\Gamma_1$-graph is built from *1-deficient words*, i.e. words under which $Q$ has an image of cardinality $n-1$. This graph is a key tool for studying completely reachable automata, as shown in [6]. In [7], this tool is extended to the $\Gamma$-graph and allows to fully characterize completely reachable automata. In [6], Bondar and Volkov conjectured that the $\Gamma_1$-graph is strongly connected if any subset is reachable by a concatenation of 1-deficient words. They also left the question about the algorithmic complexity of building the $\Gamma_1$-graph open. In this section, we first introduce the concept of *root states* of a 1-deficient word. Together with the already well known concepts of *excluded state* and *duplicate state*, it allows us to state general concatenation rules for 1-deficient words. Second, using these rules, we provide a polynomial time algorithm building the $\Gamma_1$-graph. Third, we provide a completely reachable automaton which does not fulfil Bondar and Volkov's conjecture. Finally, we prove a slightly modified version of the conjecture.

The preliminary version of this paper has been presented at the DLT2018 conference [18]. The proofs of Lemma 1, Lemma 2 and Proposition 3 , as well as the section 2.1, on the algorithmic construction of the $\Gamma_1$-graph, are new.

*Definitions*

A *deterministic complete finite automaton* is a triple $A = (Q; \Sigma; \delta)$ with $Q$ a set of states, $\Sigma$ an alphabet of letters and $\delta : Q \times \Sigma \to Q$ a transition function defined for all states and letters. In the following, we will use both "automaton" and "DFA" to name such automata. It is convenient to represent a DFA with a directed graph in which each state is represented by a node, and each transition $\delta(q_i, l_i) = q_j$ is represented by a directed edge from $q_i$ to $q_j$ labelled by letter $l_i$. For example, Fig.1 shows an automaton with $Q = \{q_1, q_2, q_3, q_4\}$, $\Sigma = \{a, b\}$, $\delta(q_1, a) = q_1$, $\delta(q_1, b) = q_2$, $\delta(q_2, a) = q_2$, $\delta(q_2, b) = q_3$, $\delta(q_3, a) = q_3$, $\delta(q_3, b) = q_4$, $\delta(q_4, a) = q_1$ and $\delta(q_4, b) = q_1$. In the following, we will often directly use a graph to describe an automaton since we can recover the formal description $(Q; \Sigma; \delta)$ of the automaton from the graph representation.

In this context, we define *words* of $\Sigma^*$ as sequences of letters of $\Sigma$. Transition functions are recursively extended to words as follows: for a word $w = l_i w'$ in $\Sigma^*$, with $l_i$ a letter and $w'$ a word, $\delta(q_i, w) = \delta(\delta(q_i, l_i), w')$. Similarly, transition functions are extended to sets of states as follows: for a set $S \in Q$ and a word $w \in \Sigma^*$, $\delta(S, w) = \{q_j | q_j = \delta(q_i, w), q_i \in S\}$. In order to simplify the notation, we write $q_i w$

for $\delta(q_i, w)$ and $Sw$ for $\delta(S, w)$. We call *rank* of a word $w$ the cardinality of its image $|Qw|$. Conversely, for the ease of notation, we call *deficiency* of a word $w$ the value $n - |Qw|$. For example *1-deficient* words are words of rank $n - 1$.

We define the *power automaton* of an automaton $A = (Q; \Sigma; \delta)$ as the automaton obtained by setting all the possible subsets of $Q$ as states, by maintaining the same letters as $A$ and with the transition function equal to the transition function of $A$ on its sets of states. The *square graph* of an automaton is equal to the restriction of the power automaton to pairs. A set $S \in Q$ is called *reachable* if there exists a word $w \in \Sigma^*$ such that $Qw = S$. This is equivalent to having a path from $Q$ to $S$ in the power automaton. In that case we say that the word $w$ reaches $S$. A set $S \in Q$ is called *extendable* by a word $w$ if $S = S'w$ for $S' \subseteq Q$ with $|S'| > |S|$. The *diameter* of a graph is the maximal length of the shortest path between two of its states.

An automaton is *sychronizing* if it has a reachable set of size 1. A word $w$ such that $|Qw| = 1$ is said to be a *synchronizing word*. For example, the automaton in Fig.1 is synchronizing as the word $w = abbbabbba$ is such that $Qw = q_1$, and therefore $w$ is a synchronizing word. Similarly, we say that a word $w$ synchronizes a set $S$ if $|Sw| = 1$.

For each 1-deficient word $w$ of the automaton, the state $Q \backslash Qw$ is the excluded state of $w$, and is written $excl(w)$. Similarly, the state $q$ such that there exists two states $q_1$ and $q_2$ with $q_1 w = q_2 w = q$ is the duplicate state of $w$, and is written $dupl(w)$. For an automaton $A = (Q; \Sigma; \delta)$, we define the $\Gamma_1$-graph as follows: $\Gamma_1(A) = (Q, V)$, with the same set of nodes $Q$ as the automaton and the set of edges $V = \{(q_i, q_j) | q_i = excl(w), q_j = dupl(w), \text{ for some } w \in \Sigma^* \text{ and } |Qw| = |Q| - 1\}$.

## 1. Subset Reachability

In this section, we provide an answer to Don's conjecture on subset reachability and analyze questions arising from it.

**Conjecture 2** (Conjecture 18 in [12])**.** *Let $A = (Q; \Sigma; \delta)$ be an $n$-state automaton. If $S \subset Q$ is a set of size $k$ and there exists a word $w$ such that $Qw = S$, then there exists a word with this property of length at most $n(n - k)$.*

This conjecture would imply that Conjecture 1 is also true [12, Proof of Theorem 12]. Indeed, any pair would be reachable with a word of length $n(n - 2)$. Since in any synchronizing automaton there is also a reachable pair which can be synchronized with a single letter, reaching this pair with a word of length $n(n - 2)$ and then synchronizing it with a single letter leads to a synchronizing word of length $n(n - 2) + 1 = (n - 1)^2$.

In order to analyze the conjecture, we need the following lemma:

**Lemma 1.** *Let $A = (Q; \Sigma; \delta)$ be an automaton. Let $S \subset Q$ be a reachable subset, and $w \in \Sigma^*$ be a shortest word such that $Qw = S$. Let us write $w = l_1 \dots l_k$, with $k$ the length of $w$ and $l_i \in \Sigma$ the letters of $w$. Let the subsets $S_i = Ql_1 \dots l_i$ with $0 < i < k$ be the subsets reached by prefixes of $w$.*

*Then, all the subsets $S_i$ are different, and they all have a cardinality larger than or equal to the cardinality of $S$.*

*Proof.* If two subsets $S_i = Ql_1 \dots l_i$ and $S_j = Ql_1 \dots l_j$, with $i < j$ are equal, then the word $l_1 \dots l_i l_{j+1} \dots l_k$, which is shorter than $w$, also reaches $S$, which is a contradiction.

In a complete deterministic automaton, each state has exactly one image. Therefore for any word $w$ and and set $S$, we have $|Sw| \leq |S|$. □

This lemma implies that any reachable set of size $n-1$ can be reached with a word of length $n$. Indeed, there are only $n+1$ different subsets of cardinality higher than or equal to $n-1$. Therefore, we will search for a counterexample to Conjecture 2 with a set of size $n-2$ which cannot be reached with a word of length lower than or equal to $2n$.

One way to build such automata is to define the letters in the following way. First choose a set of permutation letters such that the square graph of the automaton limited to these letters has a large diameter. Second, choose a pair $(q_i, q_j)$ such that the shortest path to an other pair $(q_k, q_l)$ in the square graph has a large length $D$. Then, add a 2-deficient letter $l$ such that $Q \backslash Ql = \{q_i, q_j\}$. With this construction, the set $Q \backslash \{q_k, q_l\}$ cannot be reached by words shorter than $D+1$.

In Fig.2, we present the automaton $\mathcal{P}_{1,n}$, with $n$ congruent to 3 modulo 4 which is built in that way.

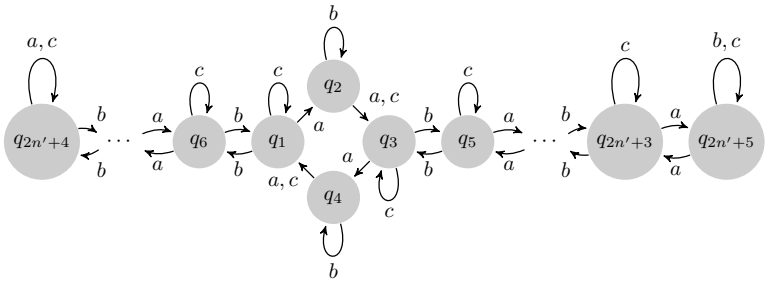


Figure 2: The automaton $\mathcal{P}_{1,n}$

**Proposition 1.** *The shortest word reaching the set* $Q \backslash \{q_{n'+2}, q_{n'+4}\}$ *with* $n' = (n-5)/2$ *in the automaton in* $\mathcal{P}_{1,n}$ *is of length* $n^2/4 + 5n/4 - 6$.

*Proof.* We first notice that letters $a$ and $b$ of the automaton $\mathcal{P}_{1,n}$ are a set of permutation letters studied in [15] such that the shortest path from the pair $q_2q_4$ to $q_{n'+2}q_{n'+4}$ is of length $n^2/4 + 5n/4 - 7$. As $a$ and $b$ are permutations, we have $Qa = Q$ and $Qb = Q$, so the first letter of any shortest word reaching a set $S$ should be $c$.

Second, we notice that letter $c$ maps $q_2$ to $q_3$ and $q_4$ to $q_1$, and the other states to themselves, so $Qc = Q \setminus \{q_2, q_4\}$ and is of cardinality $n-2$. Moreover, if letter $c$ is applied to any set containing $n-2$ states, it either sends it to $Q \setminus \{q_2, q_4\}$, or to a set of lower cardinality. Therefore, due to Lemma 1, in any shortest word reaching $Q \setminus \{q_{n'+2}, q_{n'+4}\}$, the letter $c$ can only be at the first position.

Third, as $a$ and $b$ are permutations, the image of a set $S$ by a word $w \in \{a,b\}^*$ is equal to the complement of the image of the complement of $S$ by $w$, i.e. $Sw = Q \setminus ((Q \setminus S)w)$. In our case, applying letters $a$ and $b$ to a set containing $n-2$ states is equivalent to applying it to the complement pair and taking the complement of the image as result. Therefore, the shortest word $w \in \{a,b\}^*$ mapping the set $Q \setminus \{q_2, q_4\}$ to $Q \setminus \{q_{n'+2}, q_{n'+4}\}$ is also the shortest word mapping $\{q_2, q_4\}$ to $\{q_{n'+2}, q_{n'+4}\}$, which is of length $n^2/4 + 5n/4 - 7$. Therefore, $cw$ is the shortest word reaching $Q \setminus \{q_{n'+2}, q_{n'+4}\}$, and is of length $n^2/4 + 5n/4 - 6$. □

The automaton of $\mathcal{P}_{1,n}$ is a generic counterexample to Conjecture 2, which claims that the distance from $Q$ to any set of size $n-2$ should be $2n$.

We notice that the reachable sets of $\mathcal{P}_{1,n}$ can be reached with words of polynomial length. Therefore, one can wonder if it is a general property:

**Problem 1.** *Let $A = (Q; \Sigma; \delta)$ be an $n$-state automaton. If there exists a word $w$ such that $Qw = S$, does there always exist a word of polynomial length with respect to $|S|$ with this property?*

### *1.1. Answer to Problem 1: Exponential Length Shortest Words*

For any $n$, we can build an automaton such that it has a set $S \subset Q$ containing $\lfloor \frac{n}{2} \rfloor$ states which cannot be reached with a word shorter than $L = \binom{n-1}{\lfloor \frac{n}{2} \rfloor}$. This will be shown in Definition 1 and Proposition 2 below.

**Definition 1.** *We define the automaton $\mathcal{P}_{2,n}$ as follows. It has $n$ states $q_1 \dots q_n$, a letter $a$ and letters $l_i$. First, define letter $a$ as follows:*

$q_1 a = q_2$
$q_i a = q_i$ *for* $1 < i \leq \lfloor \frac{n}{2} \rfloor + 1$
$q_i a = q_2$ *for* $\lfloor \frac{n}{2} \rfloor + 1 < i \leq n$.

*Then, take all subsets $S \subset (Q \setminus q_1)$ with $|S| = \lfloor \frac{n}{2} \rfloor$ and enumerate them as $S_1, \dots, S_L$, with $L = \binom{n-1}{\lfloor \frac{n}{2} \rfloor}$. Define letters $l_i$ such that $S_i l_i = S_{i+1}$ and $(Q \setminus S_i) l_i = q_1$.*

In Fig.3, we show a representation of $\mathcal{P}_{2,n}$ (we do not represent the effect of $l_1, \dots, l_{L-1}$ on other states than $q_1$ for the sake of clarity). We will show that the set $S_L$ cannot be reached with a word shorter than the size of the alphabet. Indeed, in this automaton, the only way to exclude $q_1$ from a subset is to use letter $a$. Moreover, we have that $|Qa| = |S_1| = |S_L|$. Therefore, in order to reach $S_L$, after using an $a$, letters reducing the cardinality of the current set cannot be used any more. This will imply that there is only one possible letter following any prefix of a shortest word reaching $S_L$.

**Proposition 2.** *The shortest word reaching the set $S_L = \{q_{n-\lfloor \frac{n}{2} \rfloor + 1}, \dots, q_n\}$ in the automaton $\mathcal{P}_{2,n}$ is $\binom{n-1}{\lfloor \frac{n}{2} \rfloor}$ letters long, which is larger than $2^n/n$ (for $n > 6$).*

*Proof.* From the definition of letters $l_i$, it is clear that the word $a l_1 l_2 \dots l_{L-1}$ maps $Q$ on $S_L$.

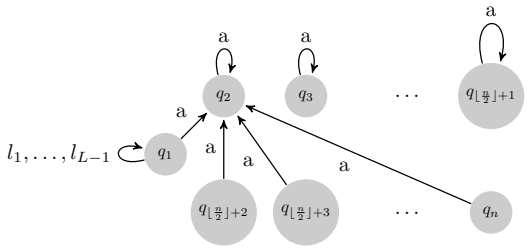


Figure 3: Part of the automaton $\mathcal{P}_{2,n}$

Moreover, it is the shortest of such words. Indeed, let $w$ be a word such that $Qw = S_L$. We notice that the only letter which maps $q_1$ on another state is $a$. Therefore, as $S_L$ does not include $q_1$ we have that: first, $w$ must contain at least one letter $a$; second, every letter after the last $a$ does not map any state of the set reached by the prefix of $w$ preceding this letter on $q_1$.

Now, if we consider a set $S_i$ of $\lfloor \frac{n}{2} \rfloor$ states from $q_2, \ldots, q_n$, the only letter $l$ which does not map any states of $S_i$ on $q_1$ is $l_i$. Therefore, if a subset $S_i$ was reached with a prefix of $w$ containing the last $a$ of $w$, the only possible letter $l$ such that $q_1 \notin S_i l$ is $l_i$. This implies that the letter following the prefix is $l_i$ and that the next subset is $S_i l_i = S_{i+1}$. Since $Qa = \{q_2, \ldots, q_{\lfloor \frac{n}{2} \rfloor+1}\}$ is already of cardinality $\lfloor \frac{n}{2} \rfloor$, the first subset after the last $a$ must be $\{q_2, \ldots, q_{\lfloor \frac{n}{2} \rfloor+1}\} = S_1$. Then, by induction, the letters following the last $a$ in $w$ are $l_1, \ldots, l_{L-1}$, in that order. Therefore, any word $w$ reaching $S_l$ has a suffix $a l_1 l_2 \ldots l_{L-1}$, which is $\binom{n-1}{\lfloor \frac{n}{2} \rfloor}$ letters long. Since this suffix is itself a word reaching $S_L$, it is also the shortest word reaching $S_L$. □

The number of letters of the automaton $\mathcal{P}_{2,n}$ is exponential with respect to the number of states. We notice that it is possible to build families of automata with a fixed number $K$ of letters, such that the shortest word reaching a particular subset has exponential length (as a function of $n$). Indeed, a construction of this kind with 32 letters for any $n$ can be obtained from the construction given in [14, Theorem 20].

In the families $\mathcal{P}_{1,n}$ and $\mathcal{P}_{2,n}$, shortest words reaching a defined subset were such that a first letter was used to lower the cardinality of the set, and the following letters were only permutations. Applying the first letter again would indeed have reduced the number of states below the cardinality of this set or reached a set already reached, which is forbidden by Lemma 1. Moreover, it turns out that these automata have a set $S$ which cannot be reached by a word of polynomial length, but such that some subsets of $S$ can be reached with such word. This leads to the following question:

**Problem 2.** *Let $A = (Q; \Sigma; \delta)$ be an $n$-state automaton. If $S \in Q$ is a set of size $k$ and there exists a word $w$ such that $Qw \subseteq S$, does there exist a word with this property of length at most $n(n-k)$?*

*1.2. Answer to Problem 2: Included subsets*

We notice that, although this question is weaker than Conjecture 2, a positive answer would still imply that Conjecture 1 is true.

However, even for this weaker version, the answer is negative. Indeed, the automaton $\mathcal{P}_3$ in Fig. 4, which was introduced in [17], is such that the set $S = \{1, 2, 3\}$ or any of its subset cannot be reached with a word of length lower than 6. This can be verified by a breadth first search algorithm on the power automaton, presented in Fig.5.

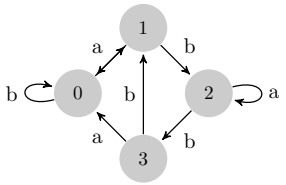


Figure 4: The automaton $\mathcal{P}_3$

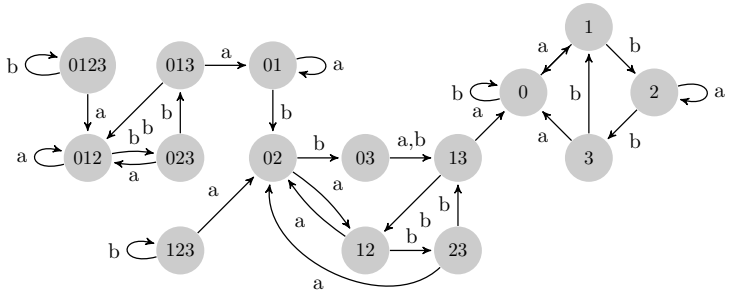


Figure 5: Power automaton of $\mathcal{P}_3$

*1.3. Complexity implication of the conjecture*

Conjecture 2 would not only imply Conjecture 1 but also the collapse of polynomial hierarchy[3]. Indeed, it is shown in [6] that the following problem is PSPACE-complete:

**Problem 3.** *Given a DFA $A = (Q, \Sigma, \delta)$ and a non-empty subset $S \subseteq Q$, is it true that $S$ is reachable in $A$?*

Would Conjecture 2 hold true, then, whenever $S$ is reachable, one could guess a word $w$ of length at most $n(n-k)$ where $n = |Q|$ and $k = |S|$ and then check in

[3]This was kindly communicated to us by an anonymous reviewer of the conference version of this work.

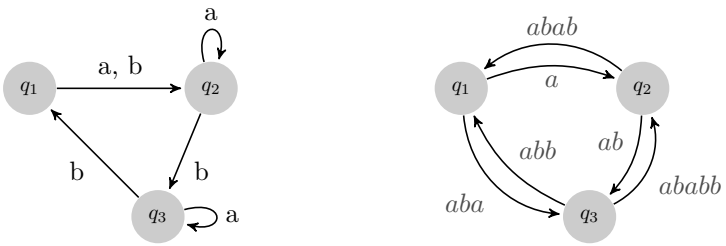


Figure 6: An automaton and the corresponding $\Gamma_1$-graph

polynomial time that $Qw = S$. Thus, the above problem would be in NP whence NP = PSPACE.

## 2. $\Gamma_1$-graph of completely reachable automata

As formally defined in the definition section, the *excluded* state of a 1-deficient word $w$ is the state which is not in its image. Conversely, the *duplicate* state of $w$ is the state which is the image of two states by $w$. From these two concepts, the $\Gamma_1$-*graph* is defined as the unlabelled graph with the same nodes as the automaton and edges corresponding to all directed pairs $(excl(w), dupl(w))$ of 1-deficient words $w$. An example of such a graph is given in Figure 6 (the edges have been labelled in gray for illustration purpose). For example, the word $a$ induces the edge $(q_1, q_2)$ in $\Gamma_1$ because $q_1$ is not in the image of $a$, so $excl(a) = q_1$ and $q_2$ is the image of both $q_1$ and $q_2$ by $a$, so $dupl(a) = q_2$.

It is shown by Don in [12] that if the $\Gamma_1$-graph is cyclic, then the automaton is completely reachable. In the work of Bondar and Volkov [6], which study completely reachable automata, it is noticed that complete reachability holds under the weaker assumption of strong connectivity of the $\Gamma_1$-graph. However, they also notice that the converse is not true: there are completely reachable automata for which the $\Gamma_1$-graph is not strongly connected. In order to show this, they provide an example using 2-deficient letters. These letters do not contribute to the $\Gamma_1$-graph but allow to reach subsets which are not reachable with 1-deficient words. The second problem that we consider is the restriction of this statement to 1-deficient words:

**Conjecture 3** (Conjecture 3 in [6])**.** *If for every proper non-empty subset $S$ of the state set of an $n$-state DFA $A = (Q; \Sigma; \delta)$ there is a product $w$ of 1-deficient words such that $S = Qw$, the graph $\Gamma_1(A)$ is strongly connected.*

The paper [6] also leaves open the algorithmic aspects of building the $\Gamma_1$-graph. The authors point out that building the $\Gamma_1$-graph is non trivial, since the number of transformations of rank $n-1$ can reach $n!C_2^n$. Moreover, a fast algorithm building the $\Gamma_1$-graph would provide a fast positive criterion for complete reachability of an automaton and could be used to experimentally investigate properties of the $\Gamma_1$-graph.

In order to formalize the link between strong connectivity of the graph $\Gamma_1$ and subset reachability, we need the following definition of a strongly connected graph:

**Definition 2.** *A directed graph $\Gamma = (N, E)$ with nodes $N$ and edges $E$ is strongly connected iff for any set $S \subset N$, there exists an edge $e = (n_1, n_2) \in E$ with $n_1 \notin S$ and $n_2 \in S$.*

We say that the edge $e = (n_1, n_2) \in E$ with $n_1 \notin S$ and $n_2 \in S$ is *intersecting* the set $S$.

If $\Gamma_1$ is strongly connected, then the automaton is completely reachable. Indeed, if any set $S \subset Q$ is intersected by an edge of $\Gamma_1$, there exists a 1-deficient word $w$ with $dupl(w) \in S$ and $excl(w) \notin S$. This first implies that each state in $S$ has a preimage by $w$. Second, as $dupl(w) \in S$, it implies that there is a state in $S$ which has two preimages. Therefore $|Sw^{-1}| > S$, and $w$ is extending $S$. As this holds for every subset $S$, the automaton is completely reachable.

### *2.1. Algorithmic construction of* $\Gamma_1$

In order to analyze the effect of word concatenation, we cannot restrict ourselves to the concepts of excluded state and duplicate state of a word. Indeed, for two 1-deficient letters $a$ and $b$, the knowledge of these states is not enough to determine whether $ab$ or $ba$ are of deficiency one or two. To analyze this question, we need an additional concept: the *roots* of a 1-deficient word. The roots are the states which are sent on the duplicate state. More formally, for a 1-deficient word $w$, $root(w) = \{q_i | q_i w = dupl(w)\}$. We notice that applying a 1-deficient word $w$ to a set $S \in Q$ does not decrease the cardinality of the set if at least one of the roots of $w$ is not in $S$. This leads to the following concatenation rule:

**Lemma 2.** *Let $A = (Q; \Sigma; \delta)$ be an $n$-state DFA. Let $w_1, w_2 \in \Sigma^*$ be two 1-deficient words. If $excl(w_1) \notin root(w_2)$, then $w_1 w_2$ is 2-deficient. If $excl(w_1) \in root(w_2)$ then $w_1 w_2$ is 1-deficient. In that case, $excl(w_1 w_2) = excl(w_2)$, $dupl(w_1 w_2) = dupl(w_1) w_2$ and $root(w_1 w_2) = root(w_1)$.*

*Proof.* If $excl(w_1) \notin root(w_2)$, then $root(w_2) \subset Qw_1$, and the two states in $root(w_2)$ are synchronized by $w_1$ while the other states are permuted, so we have $|Qw_1 w_2| = |Qw_1| - 1 = |Q| - 2$.

Moreover, if $excl(w_1) \in root(w_2)$, then all the states in $Qw_1$ have a different image by $w_2$ and $|Qw_1 w_2| = n - 1$. In that case, we have the following properties.

The state $dupl(w_1 w_2)$ is the image of $dupl(w_1)$ by word $w_2$, i.e. $dupl(w_1) w_2$.

The states $root(w_1 w_2)$ are equal to $root(w_1)$, since for these states we have that $q_r w_1 \in dupl(w_1)$ and therefore $q_r w_1 w_2 \in dupl(w_1) w_2$.

The state $excl(w_1 w_2)$ is $excl(w_2)$, since $excl(w_2) \notin Qw_2$ and $Qw_1 w_2 \subseteq Qw_2$. □

Of course, the concatenation of a 1-deficient word with a permutation also provides a 1-deficient word. In that case, the resulting word has the following properties:

**Lemma 3.** *Let $A = (Q; \Sigma; \delta)$ be an $n$-state DFA. Let $w \in \Sigma^*$ be a 1-deficient word and $p \in \Sigma^*$ be a permutation. Then both $pw$ and $wp$ are 1-deficient words. Moreover, $excl(wp) = excl(w)p$, $excl(pw) = excl(w)$, $dupl(wp) = dupl(w)p$, $dupl(pw) = dupl(w)$, $root(wp) = root(w)$ and $root(pw) = root(w)p^{-1}$.*

These two lemmas allow to build the $\Gamma_1$-graph algorithmically. Indeed, we can identify which words can be concatenated into other 1-deficient words, which induce edges in $\Gamma_1$. If two words combine into a 2-deficient word, then the concatenation does not contribute to $\Gamma_1$. The Algorithm 1 provides a pseudo code implementing the effect of Lemmas 2 and 3. The algorithm computes all 4-tuples composed of the two root states, the excluded state and the duplicate state of 1-deficient letters of the automaton, then computes the pairs of excluded and duplicate states for all 1-deficient words of the automaton. Line 1 initiates the list for letters of deficiency 1. Line 2 initiates the lists in which the results are stored. In lines 3 to 11 the 4-tuples are computed for single letters. Lines 12 to 33 describe a loop corresponding to increasing the length of words tested by one letter if new results were obtained at the previous step. Inside of this, lines 12 to 30 test each pair generated at the previous step. For each of them, lines 14 to 21 test the concatenation with a permutation letter and lines 22 to 29 test the concatenation with another 1-deficient letter.

In order to prove the properties of the algorithm, we need the following properties about 4-tuples obtained from two different words:

**Lemma 4.** *Let $w_1$ and $w_2$ be two 1-deficient words of an $n$-state DFA $A = (Q; \Sigma; \delta)$ such that $root(w_1) = root(w_2)$, $excl(w_1) = excl(w_2)$, $dupl(w_1) = dupl(w_2)$.*

(I) *If $l$ is a 1-deficient letter such that $w_1 l$ is 1-deficient, then $w_2 l$ is also 1-deficient and $root(w_1 l) = root(w_2 l)$, $excl(w_1 l) = excl(w_2 l)$ and $dupl(w_1 l) = dupl(w_2 l)$.*

(II) *If $p$ is a permutation letter, then $pw_1$, $w_1 p$, $pw_2$ and $w_2 p$ are 1-deficient words and $root(pw_1) = root(pw_2)$, $excl(pw_1) = excl(pw_2)$, $dupl(pw_1) = dupl(pw_2)$, $root(w_1 p) = root(w_2 p)$, $excl(w_1 p) = excl(w_2 p)$ and $dupl(w_1 p) = dupl(w_2 p)$.*

*Proof.* This is a direct application of Lemma 2 and Lemma 3.

(I) If $l$ is a 1-deficient letter such that $w_1 l$ is 1-deficient, then from Lemma 2 we have that $excl(w_1) \in root(l)$. As $excl(w_1) = excl(w_2)$, we have $excl(w_2) \in root(l)$ and by Lemma 2 $w_2 l$ is 1-deficient. Then, again from Lemma 2, $root(w_1 l) = root(w_1) = root(w_2) = root(w_2 l)$, $excl(w_1 l) = excl(l) = excl(w_2 l)$ and $dupl(w_1 l) = dupl(w_1) l = dupl(w_2) l = dupl(w_2 l)$.

(II) If $p$ is a permutation letter, then Lemma 3 implies $root(pw_1) = root(w_1)p^{-1} = root(w_2)p^{-1} = root(pw_2)$, $excl(pw_1) = excl(w_1) = excl(w_2) = excl(pw_2)$, $dupl(pw_1) = dupl(w_1) = dupl(w_2) = dupl(pw_2)$, $root(w_1 p) = root(w_1) = root(w_2) = root(w_2 p)$, $excl(w_1 p) = excl(w_1)p = excl(w_2)p = excl(w_2 p)$ and $dupl(w_1 p) = dupl(w_1)p = dupl(w_2)p = dupl(w_2 p)$.

□

**Corollary 1.** *Let $w_1$ and $w_2$ be two 1-deficient words of an $n$-state DFA $A = (Q; \Sigma; \delta)$ such that $root(w_1) = root(w_2)$, $excl(w_1) = excl(w_2)$, $dupl(w_1) = dupl(w_2)$. Then for any 1-deficient word $w_3$ or permutation word $p$, the following pairs of words (i) $w_1 w_3$ and $w_2 w_3$, (ii) $pw_1$ and $pw_2$, (iii) $w_1 p$ and $w_2 p$ are such that the first one is 1-deficient if and only if the second one is. In that case both words have the same corresponding 4-tuples composed of the root states, excluded state and duplicate state.*

**Algorithm 1** Construction algorithm for the $\Gamma_1$-graph

**INPUT**: *An automaton* $(Q, \Sigma, \delta)$. *We consider* $|Q| = n$. *The alphabet is separated between permutation letters and 1-deficient letters as follows:* $\Sigma = \Sigma_0 \cup \Sigma_1$, *with* $m_0$ *permutation letters* $p_1, \ldots, p_{m_0}$ *in* $\Sigma_0$ *and* $m_1$ *1-deficient letters* $l_1, \ldots, l_{m_1}$ *in* $\Sigma_1$.

**OUTPUT**: *The exhaustive list of pairs composed of* $excl(w)$ *and* $dupl(w)$ *for any 1-deficient word* $w$ *of the automaton, and the list of 4-tuples composed of* $excl(l)$, $dupl(l)$, $root1(l)$ *and* $root2(l)$ *for any 1-deficient letter* $l$ *of the automaton.*

Initiate empty list $L_0$
▷ List of 4-tuples of single letters
Initiate empty lists $L_1$, $L_2$, $L_3$
▷ List of pairs. $L_1$ for the final output, $L_2$ for the previous step of the algorithm, $L_3$ for the current step of the algorithm
▷ Single letters computation
**for** $i = 1, \ldots, m_1$ **do**
    Compute $excl(l_i)$, $dupl(l_i)$, $root1(l_i)$, $root2(l_i)$
    **if** $[excl(l_i), dupl(l_i), root1(l_i), root2(l_i)]$ is not in $L_0$ **then**
        append $[excl(l_i), dupl(l_i), root1(l_i), root2(l_i)]$ to $L_0$
    **end if**
    **if** $[excl(l_i), dupl(l_i)]$ is not in $L_1$ **then**
        append $[excl(l_i), dupl(l_i)]$ to $L_1$, $L_2$ and $L_3$
    **end if**
**end for**
▷ Words computation
**while** $L_3$ is not empty **do** ▷ If the algorithm did not stagnate at previous step
    Empty the list $L_3$ ▷ Reset current step
    **for** $i = 1, \ldots, length(L_2)$ **do**
        $[V_1, V_2] = L_2(i)$ ▷ Test each pair added in the previous step
        **for** $j = 1, \ldots, m_0$ **do** ▷ With each possible permutation
            $TestPair = [V_1 l_j, V_2 l_j]$
            **if** $TestPair$ is not in $L_1$ **then** ▷ If the result is new, store it
                append $TestPair$ to $L_1$ and $L_3$
            **end if**
        **end for**
        **for** $j = 1, \ldots, m_1$ **do** ▷ With each possible 1-deficient letter
            ▷ Test if the letter can be concatenated at right
            **if** $V_1$ equal $root1(l_j)$ or $root2(l_j)$ **then**
                $TestPair = [excl(l_j), V_2 l_j]$
                **if** $TestPair$ is not in $L_1$ **then** ▷ If the result is new, store it
                    append $TestPair$ to $L_1$ and $L_3$
                **end if**
            **end if**
        **end for**
    **end for**
    Empty the list $L_2$ ▷ Reset previous step
    $L_2 = L_3$ ▷ Current step becomes previous step
**end while**

This implies that if, in a greedy exhaustive search for pairs composed of an excluded and a duplicate state, two words $w_1$ and $w_2$ have the same pair, then $w_2$ can be ignored as all the results obtained from concatenating a letter to $w_1$ will replicate the pair obtained by concatenating the same letter to $w_2$.

Now, we notice that if no new pair is found with words of length $k+1$ compared to pairs obtained with words of length $k$ or lower, then all pairs have been found. More generally, this result is true for 4-tuples corresponding to words:

**Lemma 5.** *Let $A = (Q; \Sigma; \delta)$ be an $n$-state DFA. If for any 1-deficient word $w_1 \in \Sigma^{k+1}$ there exists a 1-deficient word $w_2 \in \Sigma^{\leq k}$ such that $root(w_1) = root(w_2)$, $excl(w_1) = excl(w_2)$, $dupl(w_1) = dupl(w_2)$, then for any 1-deficient word $w_3 \in \Sigma^{\geq k+1}$ there exists a 1-deficient word $w_4 \in \Sigma^{\leq k}$ such that $root(w_3) = root(w_4)$, $excl(w_3) = excl(w_4)$, $dupl(w_3) = dupl(w_4)$.*

*Similarly, if for any 1-deficient word $w_1 \in \Sigma^{k+1}$ there exists a 1-deficient word $w_2 \in \Sigma^{\leq k}$ such that $root(w_1) = root(w_2)$, $excl(w_1) = excl(w_2)$, then for any 1-deficient word $w_3 \in \Sigma^{\geq k+1}$ there exists a 1-deficient word $w_4 \in \Sigma^{\leq k}$ such that $root(w_3) = root(w_4)$, $excl(w_3) = excl(w_4)$.*

*Proof.* We proceed by induction.

First, by definition, any 1-deficient word $w_3 \in \Sigma^{k+1}$ is such that there exists a 1-deficient word $w_4 \in \Sigma^{\leq k}$ such that $root(w_3) = root(w_4)$, $excl(w_3) = excl(w_4)$, $dupl(w_3) = dupl(w_4)$.

Second, we assume that for any 1-deficient word $w_1 \in \Sigma^{k+1 \leq, \leq k+i}$, $i > 1$, there exists a 1-deficient word $w_2 \in \Sigma^{\leq k}$ such that $root(w_1) = root(w_2)$, $excl(w_1) = excl(w_2)$, $dupl(w_1) = dupl(w_2)$.

Then we have that any 1-deficient word $w_3 \in \Sigma^{\leq k+i+1}$ can be written as:

(I) $w'_3 p$, if the last letter $p$ is a permutation and the prefix $w'_3$ is 1-deficient;

(II) $w'_3 l$, if the last letter $l$ is 1-deficient and the prefix $w'_3$ is 1-deficient;

(III) or $pw'_3$, if the last letter is 1-deficient and it turns out that the prefix preceding it is a permutation.

We notice that in the three cases above, $w'_3$ is a 1-deficient word of length $k+i$. Therefore, from the assumption above, there exists a 1-deficient word $w'_4 \in \Sigma^{\leq k}$ such that $root(w'_3) = root(w'_4)$, $excl(w'_3) = excl(w'_4)$, $dupl(w'_3) = dupl(w'_4)$. Then, the 4-tuple corresponding to $pw'_3$, $w'_3 p$ or $w'_3 l$ is the same as the one corresponding to $pw'_4$, $w'_4 p$ or $w'_4 l$, which is of length at most $k+1$ and 1-deficient. Again from the assumption, any 4-tuple corresponding to $pw'_4$, $w'_4 p$ or $w'_4 l$ also corresponds to a word $w_4 \in \Sigma^{\leq k}$. Therefore we have that for any 1-deficient word $w_3 \in \Sigma^{\leq k+i+1}$, there exists a 1-deficient word $w_4 \in \Sigma^{\leq n}$ such that $root(w_3) = root(w_4)$, $excl(w_3) = excl(w_4)$, $dupl(w_3) = dupl(w_4)$.

This implies that this property is true for any $i \geq 1$, and therefore the lemma is true.

The proof of the restriction to the excluded state and the duplicate state is the same, except that case (III) is directly solved as the roots are not taken into account, and therefore in that case the word $w'_3$ already satisfies the condition. □

**Theorem 1.** *Algorithm 1 provides the exhaustive list of pairs which can be obtained from 1-deficient words and has a computational complexity in $O(mn^2)$, with $m$ the number of letters and $n$ the number of states of the automaton.*

*Proof.* First, we notice that any pair obtained through the algorithm corresponds to the application of Lemma 2 and 3 to a concatenation of letters, and therefore is a valid pair corresponding to a word.

Second, any pair corresponding to a 1-deficient word will be found by the algorithm. We will again proceed by induction:

(I) first, any pair corresponding to a 1-deficient word $w$ of length 1 (single letter) is in the list $L_1$ due to lines 3 to 11.

(II) Second, assume that all pairs corresponding to any 1-deficient word $w$ of length $k$ are in the list. Then, any word $w_{k+1}$ can be written as

- (A) $w'_k p$, if the last letter $p$ is a permutation and the prefix $w'_k$ is 1-deficient
- (B) $w'_k l$, if the last letter $l$ is 1-deficient and the prefix $w'_k$ is 1-deficient
- (C) or $pw'_k$, if the last letter is 1-deficient and it turns out that the prefix preceding it is a permutation.

In all three cases, $w'_k$ is of length $k$, and therefore the corresponding pair is in the list $L_1$ (the case $pw'_k$ is therefore directly solved). Therefore, it was also added in this list at some step, at which it was in the list $L_2$. At that step, from line 12 of Algorithm 1 the pair corresponding to the words $w'_k p$, $w'_k l$ were tested for any 1-deficient letter or permutations. Therefore the pair corresponding to $w_{k+1}$ was either already in the list, or was added in the list at that step.

Now, we notice that the final list $L_1$ contains at most $n^2$ pairs. Each of them is added exactly once to the table and tested exactly once, when it is in the list $L_2$, and if no pair is added in list $L_2$ after line 32, then due to Lemma 5 the algorithm is stopped and all pairs have been found. Once a pair is added, the concatenation with all the letters is tested, so each new pair induces at most $m$ comparisons, which corresponds to line 18 and 25. We notice that, if the list $L_1$ is defined as a binary vector of length $n^2$ indicating if each pair is in the list or not, then verifying that a pair is in the list takes $O(1)$ operation, while the naive implementation comparing each new pair with all pairs already in the list would take $O(n^2)$. As there are at most $n^2$ pair, this loop can be executed at most $n^2$ times, leading to $n^2 m$ operations at most, which provides the algorithmic complexity of $O(mn^2)$. □

### *2.2. Completely Reachable Automaton with Weakly Connected $\Gamma_1$-graph*

We now define in Fig. 7 the automaton $\mathcal{P}_4$, which has six states and six letters of rank 5. Using Lemma 2, we will prove that it is a counterexample to Conjecture 3.

**Proposition 3.** *The automaton $\mathcal{P}_4$ is completely reachable, and the graph $\Gamma_1(\mathcal{P}_4)$ is not strongly connected.*

*Proof.* Figure 9 lists all the words of rank 5 of this automaton, with their duplicate states, excluded states and root states. In the first six lines are listed the single letters.

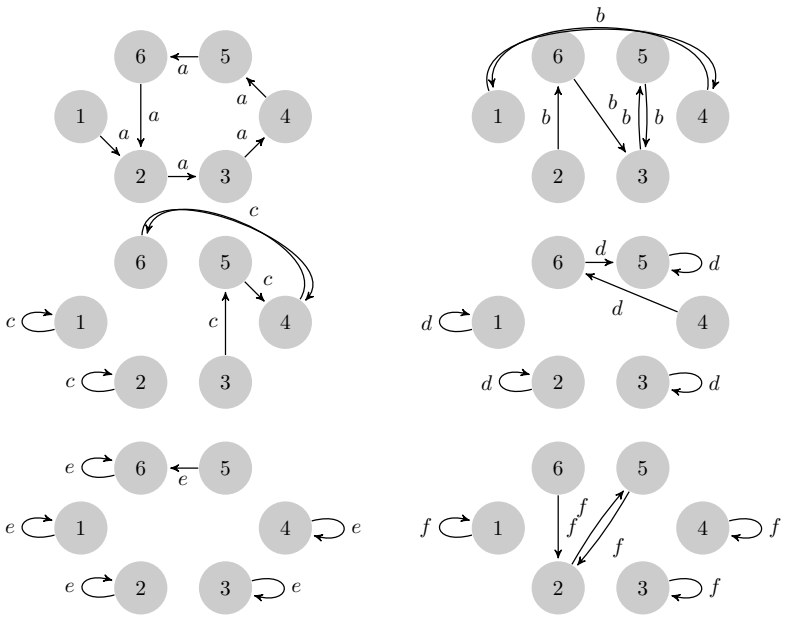


Figure 7: The 6 letters of the automaton $\mathcal{P}_4$

We notice that the roots of any letter are states 1, 5 and 6, and the only letters with 1, 5 or 6 as excluded states are $a$, $e$ and $f$. Therefore, due to Lemma 2 in words of rank 5, letter $a$ can only be preceded by letters $a$ or $f$, and all the other letters, including $e$ and $f$, can only be preceded by letters $e$ or $f$. This leads to the last six lines of the table, listing all words of rank 5. The duplicate states are found by applying the combination rule of Lemma 2.

The edges of the graph $\Gamma_1$ are defined by the excluded state and duplicate state of words in Fig.9, i.e. the second and third columns. Since state 1 does not appear in the duplicate state column, there are no edges of $\Gamma_1$ intersecting the set $\{1\}$, and therefore the graph $\Gamma_1$ is not strongly connected.

We notice however that any set of $\mathcal{P}_4$ is reachable. The edges of $\Gamma_1$ obtained with single letters form a cycle with a tail containing only state 1, as shown in Fig. 8. In this setting, any subset of $Q$ except for $\{1\}$ has an intersecting edge. Therefore, all sets of states except for $\{1\}$ are extendable by 1-deficient words, which implies by induction on the number of states that they are also reachable. The set $\{1\}$ itself can also be reached from set $\{4\}$ by applying letter $b$. Therefore, since $\mathcal{P}_4$ is completely reachable and $\Gamma_1(\mathcal{P}_4)$ is not strongly connected, it is a counterexample to Conjecture 3. □

Thus, Conjecture 3 turns out to be false. However, we notice that, in order to reach the set $\{1\}$ in the automaton $\mathcal{P}_4$, the last letter used was 1-deficient, but was acting as a permutation on the remaining states. This leads to the following observation: if a set $S$ is such that $Qw = S$, and such that $w = w_1w_2$, with $w_2$ a 1-deficient word,

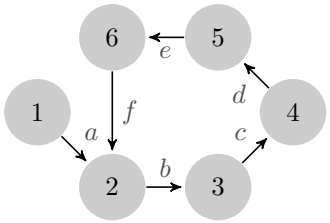


Figure 8: The graph $\Gamma_1(\mathcal{P}_4)$ restricted to single letters

| word | excluded state | duplicate state | root states |
|---|---|---|---|
| $a$ | 1 | 2 | 1, 6 |
| $b$ | 2 | 3 | 5, 6 |
| $c$ | 3 | 4 | 5, 6 |
| $d$ | 4 | 5 | 5, 6 |
| $e$ | 5 | 6 | 5, 6 |
| $f$ | 6 | 2 | 5, 6 |
| $a^*$ | 1 | 2, 3, 4, 5 or 6 | 1, 6 |
| $\{e,f\}^*$ | 5 or 6 | 2, 5 or 6 | 5, 6 |
| $\{e,f\}^*fa^*$ | 1 | 2, 3, 4, 5 or 6 | 5, 6 |
| $\{e,f\}^*b$ | 2 | 3 or 6 | 5, 6 |
| $\{e,f\}^*c$ | 3 | 2 or 4 | 5, 6 |
| $\{e,f\}^*d$ | 4 | 2 or 5 | 5, 6 |

Figure 9: Exhaustive list of words of rank 5

two possible cases arise. Either $|Qw_1| = |Qw|$, or $|Qw_1| = |Qw| + 1$. In the first case, the word $w_2$ acts as a permutation on the set $Qw_1$. In the latter case, $w_2$ synchronizes two states of $Qw_1$, and we observe the following:

**Lemma 6.** *Let $A = \{Q, \Sigma, \delta\}$ be an $n$-state DFA. Let $S \subset Q$, $w_1, w_2 \in \Sigma^*$, and $w_2$ be a 1-deficient word. If $Qw_1w_2 = S$ and $|Qw_1| = |Qw_1w_2| + 1$, then there is an edge intersecting $S$ in $\Gamma_1(A)$.*

*Proof.* The edge $(excl(w_2), dupl(w_2))$ is an edge of $\Gamma_1$ which intersects $S$. Indeed, we first have $excl(w_2) \notin S$ because $S = Qw_1w_2 \subseteq Qw_2$, and $excl(w_2) \notin Qw_2$. Second, we have $root(w_2) \subset Qw_1$, because otherwise $|Qw_1| = |Qw_1w_2|$, which contradicts the assumption that $|Qw_1| = |Qw_1w_2| + 1$. As $dupl(w_2)$ is the image of $root(w_2)$ by $w_2$, it implies that $dupl(w_2)$ is in $Qw_1w_2 = S$. □

Based on this observation, we prove the following modified version of Conjecture 3:

**Theorem 2.** *Let $A = \{Q, \Sigma, \delta\}$ be an $n$-state DFA. If for every proper non-empty subset $S \subset Q$ there is a word $w = w_1 w_2 \in \Sigma^*$ with $S = Qw$, such that $w_2$ is 1-deficient and $|Qw_1| = |Qw| + 1$, then the graph $\Gamma_1(A)$ is strongly connected.*

*Proof.* It is a direct application of Lemma 6 on any set of states $S \subset Q$ of the automaton. Indeed, it implies that any set has an intersecting edge in $\Gamma_1$, which is the definition of a strongly connected graph. □

In particular, this theorem applies to automata with a simple idempotent, as studied in [25], and for automata with letters generating the full transformation group $T_n$, as studied in [15].

## 3. Conclusion

In this article, we focused on subset reachability in synchronizing automata. The first problem considered was the length of the shortest word reaching some subsets, motivated by [12, Conjecture 18]. We started by presenting a family of automata built from a set of permutations with a large square graph diameter. This family has subsets of $n - 2$ states which cannot be reached by words shorter than a quadratic value, and therefore is a counterexample to Conjecture 2. Then, we analyzed modified versions of Conjecture 2 : first if the length of the shortest reaching word could be bounded by a polynomial value, second if it was relaxed to *subsets included* in the target subset instead of the target subset itself. Both cases led to negative answers. In particular, we built a family of strongly connected synchronizing automata with subsets that cannot be reached with words shorter than $\frac{2^n}{n}$.

The second problem that we considered was the construction of the $\Gamma_1$-graph of an automaton, motivated by the work of Bondar and Volkov [6]. We added the concept of *root states* of 1-deficient letter to the already well studied excluded state and duplicate state. With these concepts in hand, we provided an analysis of the way 1-deficient words combine with each other and with permutations, and their influence on the $\Gamma_1$-graph. This analysis first allowed to build a polynomial time algorithm for constructing the $\Gamma_1$-graph, then to build a counterexample to Conjecture 3, and finally to prove a modified version of this conjecture, namely Theorem 2.

To conclude, we propose the following problem, which is the restriction of Conjecture 2 to completely reachable automata.

**Problem 4.** *Let $A = (Q; \Sigma; \delta)$ be an $n$-state completely reachable automaton. For any $0 < k < n$ and any set $S \in Q$ of size $k$, is it true that there exists a word $w$ such that $Qw = S$ and $|w| \leq n(n - k)$?*

A positive answer to this problem would prove Černý's conjecture for completely reachable automata.

**Acknowledgements**

The authors would like to thank Balázs Gerencsér and Vladimir Gusev for fruitful discussions and advice, Élodie Boucquey, Myriam Gonze and Xavier Gonze for careful reading of the paper, the anonymous reviewers of the conference version for their very helpful comments and advices and the anonymous reviewers of the final version for their very helpful comments and for suggesting a major improvement of the algorithm building the $\Gamma_1$-graph.

## References

[1] J. Araújo, P. Cameron, B. Steinberg, Between primitive and 2-transitive: Synchronization and its friends. *EMS Surveys in Mathematical Sciences* **4** (2017) 2, 101–184.

[2] M.-P. Béal, M. Berlinkov, D. Perrin, A quadratic upper bound on the size of a synchronizing word in one-cluster automata. *International Journal of Foundations of Computer Science* **22** (2011) 2, 277–288.

[3] M. Berlinkov, R. Ferens, M. Szykuła, Complexity of preimage problems for deterministic finite automata. *arXiv preprint arXiv:1704.08233* (2017).

[4] V. Berthé, M. Rigo, *Combinatorics, automata and number theory*. 135, Cambridge University Press, 2010.

[5] V. Blondel, R. Jungers, A. Olshevsky, On primitivity of sets of matrices. *Automatica* **61** (2015), 80–88.

[6] E. Bondar, M. Volkov, Completely reachable automata. In: *International Workshop on Descriptional Complexity of Formal Systems*. Lecture Notes in Computer Science 9777, Springer-Verlag, 2016, 1–17.

[7] E. Bondar, M. Volkov, A characterization of completely reachable automata. In: *International Conference on Developments in Language Theory*. Lecture Notes in Computer Science 11088, Springer-Verlag, 2018, 145–155.

[8] C. Catalano, R. M. Jungers, On random primitive sets, directable NDFAs and the generation of slowly synchronizing DFAs (2018).

[9] J. Černý, Poznámka k homogénnym eksperimentom s konečnými automatmi. *Matematicko-fysikalny Casopis SAV* **14** (1964), 208–216.

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

[11] P.-Y. Chevalier, J. Hendrickx, R. Jungers, Reachability of consensus and synchronizing automata. In: *Decision and Control (CDC), 2015 IEEE 54th Annual Conference on*. IEEE, 2015, 4139–4144.

[12] H. Don, The Černý conjecture and 1-contracting automata. *The Electronic Journal of Combinatorics* **23** (2016) 3, 3–12.

[13] L. Dubuc, Sur les automates circulaires et la conjecture de Černý. *RAIRO Informatique Theorique et Appliquée* **32** (1998), 21–34.

[14] L. Fleischer, M. Kufleitner, Green's relations in finite transformation semigroups. In: *International Computer Science Symposium in Russia*. Springer, 2017, 112–125.

[15] F. Gonze, V. Gusev, B. Gerencsér, R. Jungers, M. Volkov, On the interplay between Babai and Černý's conjectures. In: *International Conference on Developments in Language Theory*. Lecture Notes in Computer Science 10396, Springer-Verlag, 2017, 185–197.

[16] F. Gonze, R. Jungers, On the synchronizing probability function and the triple rendezvous time for synchronizing automata. *SIAM Journal on Discrete Mathematics* **30** (2016) 2, 995–1014.

[17] F. Gonze, R. Jungers, A. Trahtman, A note on a recent attempt to improve the Pin-Frankl bound. *Discrete Mathematics and Theoretical Computer Science* **17** (2015), 307–308.

[18] F. Gonze, R. M. Jungers, On completely reachable automata and subset reachability. In: *International Conference on Developments in Language Theory*. Lecture Notes in Computer Science 11088, Springer-Verlag, 2018, 330–341.

[19] R. Jungers, The synchronizing probability function of an automaton. *SIAM Journal on Discrete Mathematics* **26** (2012) 1, 177–192.

[20] J. Kari, Synchronizing finite automata on eulerian digraphs. *Theoretical Computer Science* **295** (2003), 223–232.

[21] A. Kirnasov, On glueing states of an automaton. *Discrete Mathematics and Applications dma* **13** (2003) 4, 371–389.

[22] P. Linz, *An introduction to formal languages and automata*. Jones & Bartlett Publishers, 2011.

[23] E. Moore, Gedanken-experiments on sequential machines. *Annals of mathematics studies* **34** (1956), 129–153.

[24] C. Nicaud, Fast synchronization of random automata. In: *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, (APPROX/RANDOM 2016)*. 2016, 43:1–43:12.

[25] I. Rystsov, Estimation of the length of reset words for automata with simple idempotents. *Cybernetics and Systems Analysis* **36** (2000) 3, 339–344.

[26] B. Steinberg, The averaging trick and the Černý conjecture. *International Journal of Foundations of Computer Science* **22** (2011), 1697–1706.

[27] M. Szykula, Improving the Upper Bound on the Length of the Shortest Reset Word. In: *35th Symposium on Theoretical Aspects of Computer Science (STACS 2018)*. LIPIcs 96, Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2018, 56:1–56:13.

[28] A. Trahtman, The Černý conjecture for aperiodic automata. *Discrete mathematics and Theoretical Computer Science* **9** (2007) 2, 3–10.

[29] M. Volkov, Synchronizing automata and the Černý conjecture. In: *LATA 2008*. Lecture Notes in Computer Science 5196, Springer-Verlag, 2008, 11–27.

[30] V. Vorel, Subset synchronization and careful synchronization of binary finite automata. *International Journal of Foundations of Computer Science* **27** (2016) 05, 557–577.