Fundamenta Informaticae XX (2003) 1–15
IOS Press

Computing forbidden words of regular languages

Marie-Pierre Béal [C], **Maxime Crochemore**[*]
Institut Gaspard-Monge,
Université de Marne-la-Vallée,
77454 Marne-la-Vallée Cedex 2, France.
{beal,mac}@univ-mlv.fr

Filippo Mignosi[†], **Antonio Restivo**[†], **Marinella Sciortino**[†]
Dipartimento di Matematica ed Applicazioni,
Università di Palermo,
Via Archirafi 34, 90123 Palermo, Italy.
{mignosi,restivo,mari}@math.unipa.it

Abstract. We give a quadratic-time algorithm to compute the set of minimal forbidden words of a factorial regular language. We give a linear-time algorithm to compute the minimal forbidden words of a finite set of words. This extends a previous result given for the case of a single word only.

We also give quadratic-time algorithms to check whether a regular language is factorial or anti-factorial.

Keywords: Anti-factorial language, factor automaton, factorial language, failure function, forbidden word, formal languages, sofic shift, symbolic dynamics.

1. Introduction

A *factorial language* is a language that contains all factors (or subwords) of its words. A word is called *minimal forbidden* for a given language if it does not belong to it and all its proper factors do. The

[C]Corresponding author
[*] Work by this author is partially supported by NATO Science Programme grant PST.CLG.977017.
[†]Partially support by MURST Project “Linguaggi Formali ed Automi: Teoria e Applicazioni”.

notion of minimal forbidden words is basic in combinatorics of words (see for instance [1] or [13]). For instance, some relevant information on the structure of a word can be detected by looking at its minimal forbidden factors.

Minimal forbidden words have recently found applications in several areas like lossless compression (data compression using antidictionaries [8]) and also reconstruction of DNA sequences from its fragments (the fragment assembly problem [4], [14]).

In this paper we focus on the problem of computing the set of minimal forbidden words of a factorial regular language. This problem has been solved in linear time in [7] when the factorial language is the set of factors of a single word w. Moreover, it is proved in [14] that the sizes of the minimal forbidden words are in general very small with respect to size of the word w.

We give a quadratic-time algorithm to compute the set of minimal forbidden words of a factorial regular language and we discuss the tightness of this bound. The output is a deterministic automaton that recognizes the set of minimal forbidden words. This automaton is not the minimal one in general. We complete the preceding result by giving a linear-time algorithm to compute the minimal forbidden words when the language is the set of factors of a finite set of words. This extends directly the result of Crochemore *et al.* obtained in [7] where the set is reduced to a single word. Our construction uses also the construction of a factor automaton of a finite set of words from [3]. A corollary of the algorithm is an upper bound on the number of minimal forbidden words of a finite set of words. We give a quadratic upper bound of the minimal length of a minimal forbidden word of the language recognized by a deterministically labelled cover automaton (i.e. with all states initial and final). We prove the tightness of this quadratic bound. We also give quadratic-time algorithms to check whether a regular language is factorial or anti-factorial.

In the last section of the paper, we mention an application to the study of sofic shifts in symbolic dynamics [12]. Recall that a sofic shift is a symbolic dynamical system that can be described as the set of bi-infinite sequences labels of paths in a finite automaton. A measure of complexity of a language introduced in [1] is based on a sequence of nonnegative integers that counts, for any n, the number of minimal forbidden words of length n of the language. The exponential growth rate of the sequence associated with the factors of a symbolic dynamical system is a topological invariant. A consequence of the polynomial time computation of the set of minimal forbidden words of a factorial regular language is a polynomial-time test for the equality of invariants of two sofic shifts given by deterministic representations of their factors.

2. Minimal forbidden words of a regular language

Let A be a finite alphabet and A^* be the set of the words drawn from the alphabet A, the empty word ϵ included. Let $L \subseteq A^*$ be a *factorial language*, i.e. a language satisfying:

$$\forall u, v \in A^* \; uv \in L \Longrightarrow u, v \in L.$$

The complement language $L^c = A^* - L$ is an ideal of A^*. By denoting by $MF(L)$ the base of this ideal we have that $L^c = A^* MF(L) A^*$. The set $MF(L)$ is called the set of *minimal forbidden words* for L. A word $v \in A^*$ is forbidden for the factorial language L if $v \notin L$, which is equivalent to say that v occurs in no word of L. In addition, v is minimal if it has no proper factor that is forbidden. Remark that

$$L = A^* - A^* MF(L) A^*. \qquad (1)$$

It is also straightforward that a word $v = a_1 a_2 \cdots a_n$ belongs to $MF(L)$ iff the two conditions hold:

- v is forbidden i.e. , $v \notin L$,
- both $a_1 a_2 \cdots a_{n-1} \in L$ and $a_2 a_3 \cdots a_n \in L$ (the prefix and the suffix of v of length $n-1$ belong to L).

The set $MF(L)$ is an *anti-factorial language* or a *factor code*, which means that it satisfies: $\forall u, v \in MF(L)$ $u \neq v \Longrightarrow u$ is not a factor of v, property that comes from the minimality of words of $MF(L)$. Equivalently

$$MF(L) = AL \cap LA \cap (A^* - L). \tag{2}$$

As a consequence of both equalities (1) and (2) we recall the following proposition from [7] stating that L and $MF(L)$ are both regular or both not regular.

Proposition 2.1. Given a factorial language L, L is regular if and only if $MF(L)$ is regular.

In this section we address the problem of computing in polynomial time a deterministic finite state automaton recognizing $MF(L)$ given a deterministic finite state automaton recognizing L. If L is the set of factors of a single word w, then the set of its minimal forbidden factors can be computed in time $O(|w| \times |A|)$, where A is the alphabet (see [7, p. 117] for details).

A finite state deterministic automaton is a tuple (Q, A, i, T, δ) where A is the alphabet, Q is the finite set of states, i the unique initial state, T the set of final states and δ the transition function. An automaton is *trim* if each state is accessible from the initial state and coaccessible from a final state.

Assume that a trim automaton $\mathcal{A}$ recognizing the factorial language L is given. The automaton $\mathcal{A}$ is in the form (Q, A, i, Q, δ) because all states are necessarily terminal states. The algorithm MF-AUTOMATON below computes from $\mathcal{A}$ a deterministic automaton $\mathcal{A}' = (Q', A, i', T', \delta')$ recognizing $MF(L)$ where

- the set Q' of states is the set $Q \times Q \cup \{(-, i)\} \cup \{(p, \$) \mid p \in Q\}$, ("$-$","\$" are symbols that do not belong to A),
- A is the current alphabet,
- the initial state i' is the state $(-, i)$,
- the set T' of terminal states is the set of states $(p, \$)$ for $p \in Q$.

Algorithm MF-AUTOMATON (automaton $\mathcal{A} = (Q, A, i, Q, \delta)$)
$i' \leftarrow (-, i)$, $T' \leftarrow \{(p, \$) \mid p \in Q\}$;
$Q' \leftarrow Q \times Q \cup \{i'\} \cup T'$;
for each $a \in A$
  if $\delta(i, a)$ defined
    $\delta'((-, i), a) \leftarrow (i, \delta(i, a))$;
  else
    $\delta'((-, i), a) \leftarrow (i, \$)$;
for each pair of states (p, q) accessible by δ' from i' with $p, q \in Q$, $p \neq q$ **and** each $a \in A$
  if $\delta(q, a)$ defined
    $\delta'((p, q), a) \leftarrow (\delta(p, a), \delta(q, a))$;
  else
    if $\delta(p, a)$ defined
      $\delta'((p, q), a) \leftarrow (\delta(p, a), \$)$;
return automaton $\mathcal{A}' = (Q', A, i', T', \delta')$;

The pairs (p, q) considered in the algorithm are such that $p = \delta(i, w)$, $q = \delta(i, aw)$ for some letter a and some word w. Note that $\delta(p, a)$ is well defined at line 10 when $\delta(q, a)$ is defined since it is assumed that the language L is factorial and $\mathcal{A}$ is trim.

Proposition 2.2. Let $\mathcal{A}$ be a deterministic trim automaton which recognizes a factorial language L. Algorithm MF-AUTOMATON computes from $\mathcal{A}$ a deterministic automaton that recognizes $MF(L)$.

Proof:
Let us first assume that the hypothesis $p \neq q$ at line 8 is removed. There exists in $\mathcal{A}'$ a path labelled $a_1a_2 \dots a_n$ from i' to (p, q) with $p, q \in Q$, if and only if there is a path in $\mathcal{A}$ labelled $a_2 \dots a_n$ from i to p, and a path in $\mathcal{A}$ labelled $a_1a_2 \dots a_n$ from i to q. There exists in $\mathcal{A}'$ a path labelled $a_1a_2 \dots a_n$ from i' to $(p, \$)$ with $p \in Q$, if and only if there is a path in $\mathcal{A}$ labelled $a_2 \dots a_n$ from i to p, and there is no path in $\mathcal{A}$ labelled $a_1a_2 \dots a_n$ from i. The word $a_2 \dots a_n$ is a prefix of a word in L and since L is factorial and $\mathcal{A}$ is trim, it also belongs to L. This implies that $a_1a_2 \dots a_{n-1} \in L$, $a_2 \dots a_n \in L$, and $a_1a_2 \dots a_n \notin L$. Therefore $a_1a_2 \dots a_n$ is a minimal forbidden word.

Conversely if $a_1a_2 \dots a_n \in MF(L)$, $a_1a_2 \dots a_{n-1} \in L$, $a_2 \dots a_n \in L$ and $a_1a_2 \dots a_n \notin L$. There is path in $\mathcal{A}'$ labelled by $a_1a_2 \dots a_n$ from i' to some state $(p, \$)$ with $p \in Q$. Thus $a_1a_2 \dots a_n$ is recognized by $\mathcal{A}'$.

Finally states (p, p) where $p \in Q$ can be removed since they are not coaccessible from T'. This proves that $\mathcal{A}'$ recognizes $MF(L)$. □

Proposition 2.3. If the transition function is implemented by a transition matrix, then the algorithm MF-AUTOMATON runs in time $O(|Q|^2 \times |A|)$ on input automaton $\mathcal{A} = (Q, A, i, Q, \delta)$.

Proof:
Computation of the time complexity is straightforward. □

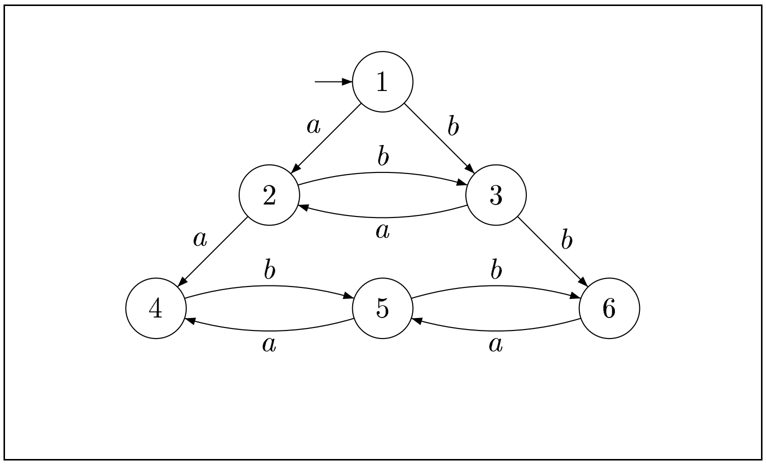

Figure 1. Automaton $\mathcal{B}$

Example 2.1. Let us consider the factorial language L recognized by the automaton $\mathcal{B}$ pictured in Figure 1. The automaton constructed from $\mathcal{B}$ with algorithm MF-AUTOMATON is the automaton $\mathcal{B}'$ described in Figure 2, where the final states are represented with rectangles.

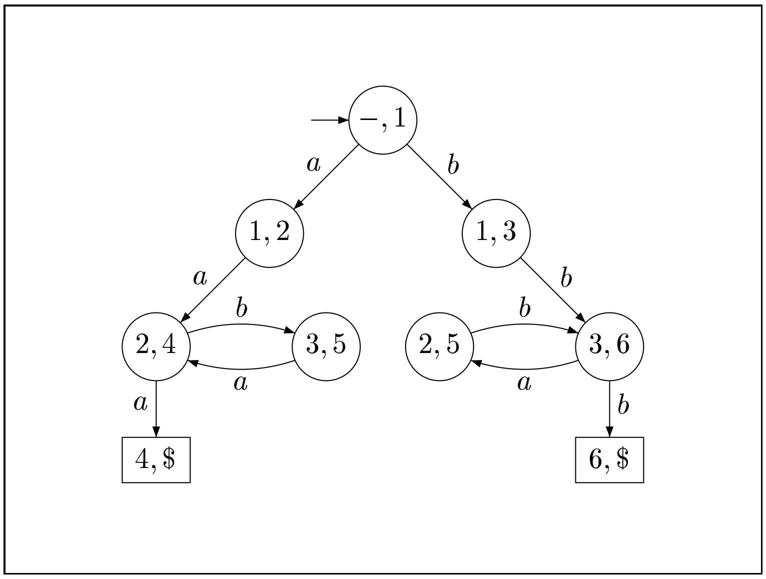

Figure 2. Automaton $\mathcal{B}'$

Proposition 2.4. Let L be a factorial language recognized by an n-state deterministic automaton, and strictly included in A^*. Then L has a minimal forbidden word of length at most n.

Proof:
Let $\mathcal{A}$ be the automaton. If the trim part of $\mathcal{A}$ is complete (δ defined for every state and every letter), $L = A^*$, a contradiction. Therefore there is a state q and a letter a for which $\delta(q, a)$ is undefined. Let w be the label of a shortest path from the initial state to q. The length of w is at most $n - 1$ and wa is a forbidden factor, then the conclusion follows. □

3. About the optimality of the quadratic bound

In this section we discuss the tightness of the quadratic bound in Proposition 2.3 in relation with the initial description of the regular language. On the one hand, we show an example of an exponential determinization from the natural description of a factorial language (also known as a *cover*). A cover, or *cover automaton*, is an automaton where all states are both initial and final. On the other hand, on such a representation, we show that the size of its associated automaton of minimal forbidden words can be at least quadratic. But we don't know whether the quadratic bound of Proposition 2.3 is tight.

An automaton is said to have a *deterministic labelling*, if it has at most one edge going out from a given state and with a given label. It is *deterministic*, if it has moreover a unique initial state. Regular factorial languages are very often represented by a finite state cover automaton with a deterministic labelling. In order to apply the algorithm MF-AUTOMATON given in Section 2, this automaton has to be transformed into a deterministic automaton which has therefore a unique initial state. This transformation is called a *determinization*.

In the example below we show that determinization by the usual subset construction (cf. [11]) applied on an n-state automaton can lead to a 2^n-state automaton.

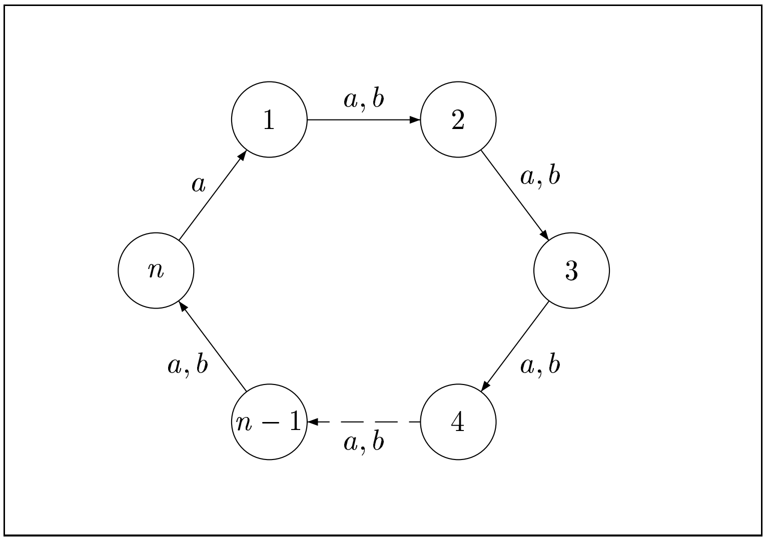

Figure 3. Automaton $\mathcal{A}_n$

Let us consider the automaton $\mathcal{A}_n = (Q, A, Q, Q, \delta)$, where $Q = \{1, 2. \ldots, n\}$, displayed in Figure 3. It is a cover since all states are both initial and final. It has a deterministic labelling described by the transition function δ.

Proposition 3.1. The deterministic finite state automaton obtained from the cover $\mathcal{A}_n$ with the classical subset construction has exactly 2^n states.

The proof below was suggested by an anonymous reviewer.

Proof:
Let L be the language recognized by the cover automaton. It is enough to prove that two distinct words u, v of length n are not equivalent for the Nerode right-congruence, i.e, if $u \neq v$, there is a word w such that $uw \in L$ and $vw \notin L$, or conversely. Let us assume that there is an index i with $1 \leq i \leq n$ such that $u_i = a$ and $v_i = b$. Remark that, if b^{n-1} labels a path of $\mathcal{A}_n$, this path begins at state number 1. Then $w = a^i b^{n-1}$ satisfies $uw \in L$ and $vw \notin L$. $\square$

In the following proposition, we prove that the number of states of a minimal deterministic automaton that recognizes the minimal forbidden words of a regular factorial language given by an n-state deterministically labelled automaton, is at least n^2 in the worst case.

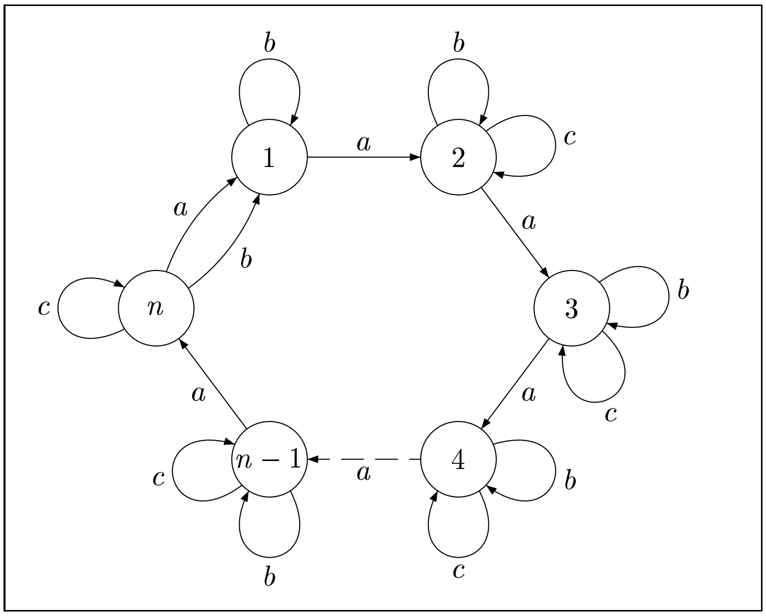

Figure 4. Automaton $\mathcal{B}_n$

Proposition 3.2. Let L be a factorial language recognized by a finite n-state cover with a deterministic labelling. The number of states of a minimal deterministic automaton $\mathcal{M}_n$ that recognizes $MF(L)$ is $\Omega(n^2)$.

Proof:
Let us consider the language $L \subset A = \{a, b, c\}$ recognized by the automaton $\mathcal{B}_n = (Q, A, Q, Q, \delta)$ pictured in Figure 4. It is based on Černý's automaton [5]. Recall that a synchronizing word is a word u such that $\delta(Q, u)$ has cardinality one. It is also known that the shortest synchronizing word of $\mathcal{B}_n$ on the 2-letter alphabet $\{a, b\}$ is the word w of length $(n-1)^2$ with

$$w = (ba^{n-1})^{n-2}b.$$

Let us now consider the word wc. It does not belong to L and its prefix and suffix of length $(n-1)^2$ do. Thus wc belongs to $MF(L)$ and there is a path in $\mathcal{M}_n$ from the initial state to a final state labelled by wc. Let $\pi = ((p_i, a_{i+1}, p_{i+1}))_{0 \le i \le (n-1)^2}$ be this path. Let us assume that there are two indices $0 \le i < j \le (n-1)^2+1$ with $p_i = p_j$ and $(i, j) \neq (0, (n-1)^2+1)$. Then $i < (n-1)^2$ since $w \notin MF(L)$. Since $wc \in MF(L)$, $a_1a_2 \ldots a_ia_{j+1} \ldots a_{(n-1)^2}c \in MF(L)$. Thus $a_1a_2 \ldots a_ia_{j+1} \ldots a_{(n-1)^2}$ would by a synchronizing word of $\mathcal{B}_n$ in the 2-letter alphabet $\{a, b\}$, focusing on state 1. This is impossible since its length is less than $(n-1)^2$. As a consequence, the states along the path π are all distinct and $\mathcal{M}_n$ has at least $(n-1)^2$ states.

Note that the word wc is not the shortest word of $MF(L)$. The word $u = (ca)^{n-1}c$ is a minimal forbidden word of L. □

A language L is *transitive* if and only if whenever u, v are two words belonging to L, there is a word w such that uwv belongs to L.

In the rest of the section, we study the size of a shortest minimal forbidden word of an n-state cover. In the following proposition, we prove that a regular factorial transitive language recognized by an n-state cover has a minimal forbidden word of length at most quadratic.

Proposition 3.3. Let $L \neq A^*$ be a factorial transitive language recognized by a finite n-state cover automaton with a deterministic labelling. Then L has a minimal forbidden word of length at most $n(n+1)/2$.

Proof:
Without loss of generality, one can choose a minimal automaton $\mathcal{A}$ with a deterministic labelling, and all states final and initial, recognizing L. This minimal cover is also called the Shannon cover of L (cf. [12]), and has a strongly connected graph. Let Q be the set of states of $\mathcal{A}$ of cardinality n and δ the transition function. Since $L \neq A^*$, the automaton $\mathcal{A}$ is not complete. Let q be a state and a a letter such that $\delta(q, a)$ is not defined.

For any subset E of Q, we define $m(E)$ as

$$m(E) = \min_{\{w \in A^* \mid \delta(E,w) = \emptyset\}} |w|,$$

and for any integer m with $0 \leq m \leq n$,

$$T(m) = \max_{\{E \subset Q \mid |E| = m\}} m(E).$$

Thus there exists a minimal forbidden word of L of length at most $T(n)$.

If $s, t \in Q$, we denote by $d(s, t)$ the minimal length of a path from state s to state t. Let m be an integer with $1 \leq m \leq n$. Let $E \subset Q$ with $|E| = m$. Then there is at least one state $p \in E$ such that $d(p, q) \leq n - m$. It follows that

$$m(E) \leq n - m + 1 + \max_{1 \leq i \leq m} T(m - i).$$

Since $\max_{1 \leq i \leq m} T(m-i) \leq T(m-1)$, we get for $1 \leq m \leq n$,

$$\begin{aligned} T(0) &= 0, \\ T(m) &\leq n - m + 1 + T(m-1). \end{aligned}$$

It follows that $T(n) \leq \frac{n(n+1)}{2}$. □

The tightness of the above quadratic bound is shown with the following example. Let us consider the n-state deterministically labelled automaton $\mathcal{C}_n = (Q, A, Q, Q, \delta)$, with $n = 2m + 1$, $A = \{a, b\}$, $Q = \{0, 1, \ldots, 2m\}$ and δ defined as follows,

$$\begin{aligned} \delta(p, a) &= p - 1 \bmod n, \text{ for any state } p, \\ \delta(p, b) &= p + m, \text{ for any state } p \text{ with } 1 \leq p \leq m, \\ \delta(p, b) &= p, \text{ for any state } p \text{ with } m + 1 \leq p \leq 2m. \end{aligned}$$

This automaton is represented in Figure 5.

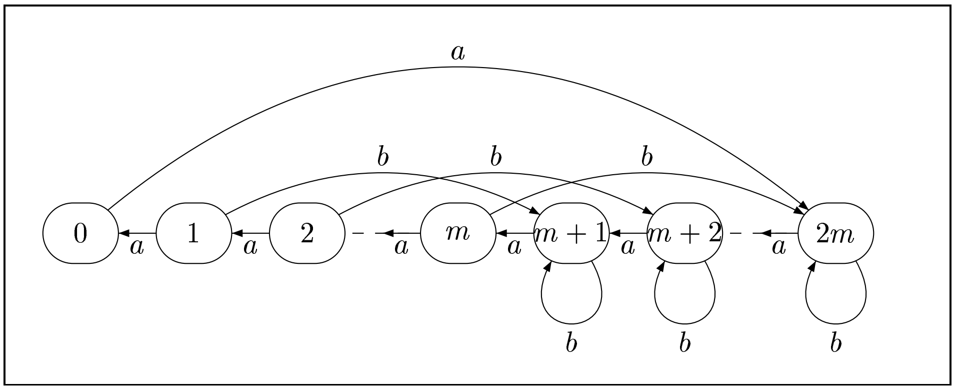

Figure 5. Automaton $\mathcal{C}_n$

Proposition 3.4. The length of a shortest minimal forbidden word of the language recognized by the n-state deterministically labelled automaton $\mathcal{C}_n$ is $(\frac{n-1}{2})^2$.

Proof:
One first remark that $\delta(Q, a) = Q$ and $\delta(Q, b) = P$, where $P = \{m+1, m+2, \ldots, 2m\}$. For any state p, $\delta(p, a^{2m+1}) = p$. We represent each subset S of P by a characteristic block (or vector) of 0 and 1's of length m. The number of 1's of a block v is denoted $|v|_1$. If v is a block and i an integer, we denote by $\delta_i(v)$ the block $\delta(v, a^i b)$. Note that $\delta(v, a^i b) \subset P$.

Let v be a block. For $0 \leq i \leq m$, $|\delta_i(v)|_1 = |v|_1$. The block $\delta_{m+1}(v)$ is obtained by a left shift of the digits of v, the last digit becoming 0. For $m+2 \leq i \leq 2m$, the block $\delta_i(v)$ is obtained by a left shift of the digits of v and by applying after $i - (m+1)$ times a left circular permutation on the digits. We call a step the application of some δ_i to a block. Its cost is $i+1$. It is then clear that the minimal number of steps going from the m-block 1^m to the m-block 0^m is m. It is obtained by applying m times δ_{m+1}. It is also the minimal cost a sequence of steps between these two blocks since the cost of δ_i is greater than the cost of δ_{m+1} for $m+2 \leq i \leq 2m$. This cost is $(m+2)m$. The length of a shortest minimal forbidden word for the language recognized by $\mathcal{C}_n$ is then $1 + m(m+2) = (\frac{n-1}{2})^2$. The word $b(a^{m+1}b)^m$ is such a word. □

This result can be connected to the tightness of the quadratic bound of the synchronization delay of an n-state local automaton obtained in [2].

4. Computing the set of minimal forbidden words of the factors of a finite set of words

In [7], Crochemore *et al.* have given a linear-time algorithm that computes the set of minimal forbidden words of the factors of a single word. In this section we extend their result to the case of factors of a finite set of words. The construction provides the trie of minimal forbidden words from a factor automaton of the set, which is not its minimal factor automaton in general. Recall that a *trie* is a tree for storing a set of words in which there is one node for every common prefix and in which the words are stored in the leaves . We obtain also a bound on the number of minimal forbidden words of the factors of a finite set of words.

Let $P = \{p^1, p^2, \dots, p^r\}$ be a finite set of words of cardinal r. We denote by $|P|$ the sum of the lengths of the words of P and by $\mathit{Fact}(P)$ the set of factors of P. A linear-time construction of a deterministic finite state automaton recognizing $\mathit{Fact}(P)$ has been obtained by Blumer *et al.* in [3], see also [15, p. 17-19]. Their construction is an extension of the well-known incremental construction of the suffix automaton of a single word (see for instance [6]). The words are added one by one to the automaton. In the sequel we call this algorithm the DAWG algorithm since it outputs a deterministic automaton called a *directed acyclic word graph*. Let us denote by $\mathit{DAWG}(P) = (Q, A, i, Q, \delta)$ this automaton. We denote by $\mathit{Suff}(v)$ the set of suffixes of a word v and by $\mathit{Suff}(P)$ the union of all $\mathit{Suff}(v)$ for $v \in P$. The states of $\mathit{DAWG}(P)$ are the equivalence classes of the right invariant equivalence $\equiv_{\mathit{Suff}(P)}$ defined as follows. If $u, v \in \mathit{Fact}(P)$,

$$u \equiv_{\mathit{Suff}(P)} v \text{ iff } \forall i, 1 \leq i \leq r,\ \ u^{-1}\mathit{Suff}(p^i) = v^{-1}\mathit{Suff}(p^i).$$

and there is a transition labelled by a from the class of a word u to the class of ua. The automaton $\mathit{DAWG}(P)$ has a unique initial state, which is the class of the empty word, and all its states are final. Note that the syntactic congruence $\sim$ defining the minimal automaton of the language is

$$u \sim v \text{ iff } \bigcup_{i=1}^{r} u^{-1}\mathit{Suff}(p^i) = \bigcup_{i=1}^{r} v^{-1}\mathit{Suff}(p^i),$$

which is not the same as the above equivalence. In other words, the $\mathit{DAWG}(P)$ is not always a minimal automaton.

The construction of $\mathit{DAWG}(P)$ is performed in time $O(|P| \times \log |A|)$. A time complexity of $O(|P|)$ can be obtained with an implementation of automata with sparse matrices (see [6]).

Example 4.1. For instance the deterministic factor automaton obtained with the DAWG algorithm for $P = \{abbab, abaab\}$ is represented in Figure 6. The thick edges are the solid edges, the other ones being fragile. Note that this deterministic automaton is not minimal since states 3 and 7, 5 and 9, and 6 and 10 can be merged pairwise.

We denote by s the suffix function associated with $\mathit{DAWG}(P)$. We first define the function s' that associates with each non-empty word of $\mathit{Fact}(P)$ a word of $\mathit{Fact}(P)$. For $v \in Fact(P), v \neq \varepsilon$, $s'(v)$ is the longest word $u \in \mathit{Fact}(P)$ that is a suffix of v and such that $u \not\equiv_{\mathit{Suff}(P)} v$. Then, $s(p)$ is the state containing $s'(v)$ where v is any word in p.

Since the graph of $\mathit{DAWG}(P)$ is an acyclic graph, algorithm MF-TRIE from [7, Section 4] can be applied to the input $\mathit{DAWG}(P)$. We recall this algorithm for the sake of completeness.

MF-TRIE (automaton $\mathit{DAWG}(P) = (Q, A, i, Q, \delta)$ and its suffix function s)
for each state $p \in Q$ in width-first search from i **and** each $a \in A$
    if $\delta(p, a)$ undefined **and** ($p = i$ **or** $\delta(s(p), a)$ defined)
        $\delta'(p, a) \leftarrow$ new sink;
    else
        if $\delta(p, a) = q$ **and** q not already reached
            $\delta'(p, a) \leftarrow q$;
return $(Q, A, i, \{sinks\}, \delta')$;

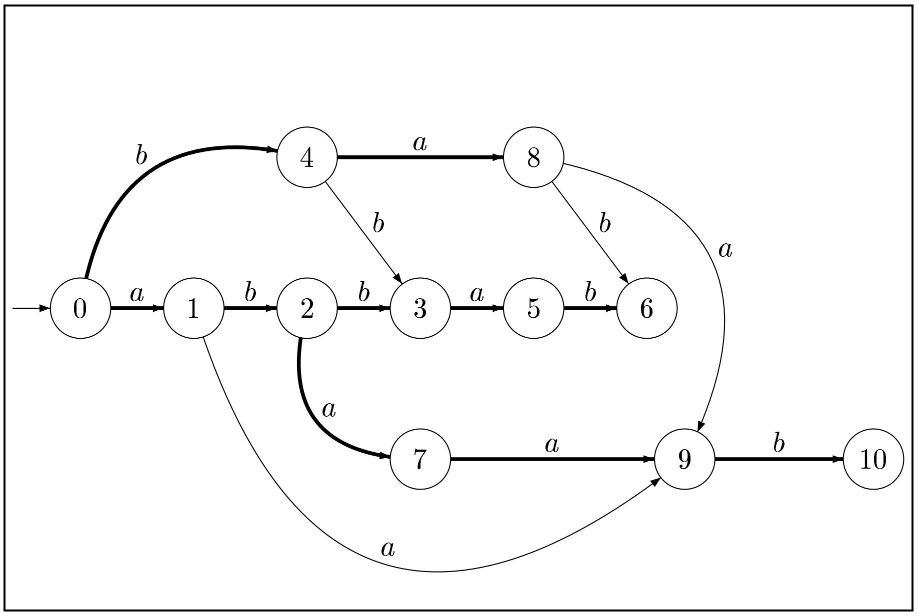

Figure 6. Automaton *DAWG*(P) for $P = \{abbab, abaab\}$

Example 4.2. The output of algorithm MF-TRIE on *DAWG*(P) with the input $P = \{abbab,\ abaab\}$ is shown in Figure 7 where the squares are final or sink states of the trie (see [15]). The set of minimal forbidden words of *Fact*(P) is

$$MF(Fact(P)) = \{c, aaa, bbb, aaba, aabb, abab, baba, babb, bbaa\}.$$

Proposition 4.1. Let *DAWG*(P) be the output of algorithm DAWG on the finite set of words P, and let s be its suffix function. Algorithm MF-TRIE builds the trie recognizing the set of minimal forbidden words of *Fact*(P), that is *MF*$($*Fact*$(P))$.

Proof:
The proof is essentially the same as those given for a set of single word in [7]. □

A main point in algorithm MF-TRIE is that it uses function s defined on states of the input DAWG. It is not possible to proceed similarly when considering the minimal factor automaton of *Fact*(P) because there is no analogue function s. However, it is possible to reduce the automaton *DAWG*(P) by merging states having the same future (right context) and the same image by s. For example, on the DAWG of Figure 7, states 6 and 10 can be merged because $s(6) = s(10) = 2$. States 3 and 7, nor states 5 and 9 cannot be merged with the same argument.

Proposition 4.2. Algorithms DAWG and MF-TRIE together run in time $O(|P| \times |A|)$ with input a finite set P, if the transition functions are implemented by transition matrices.

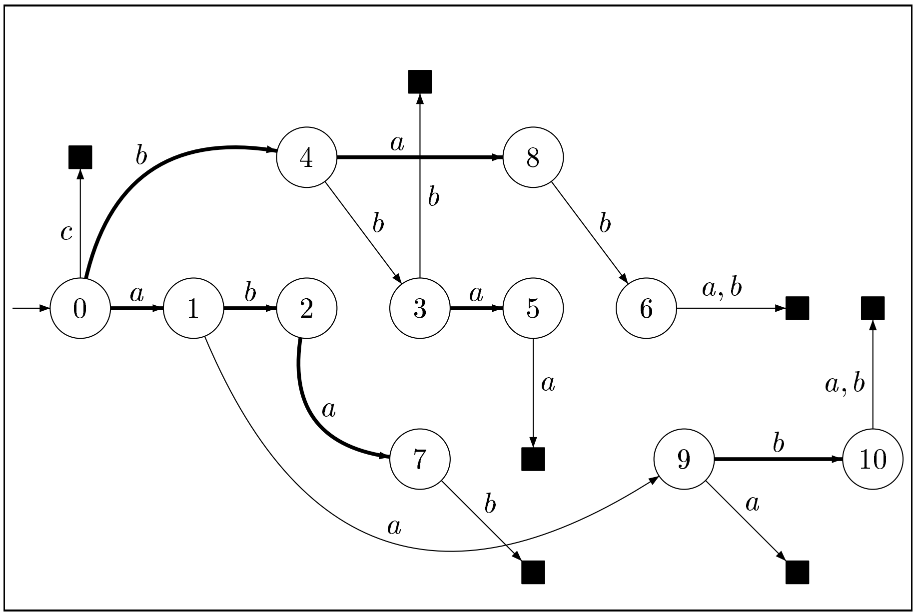

Figure 7. Automaton of minimal forbidden words of *Fact*$(abbab, abaab)$

Corollary 4.1. The number of minimal forbidden words of the set of factors of a finite set of words P is no more than $(2|P| - 2)(|A_P| - 1) + |A| - |A_P| + r$, if $|P| > 1$, where $|A_P|$ is the set of letters occurring in P and r the number of words in P. The bound becomes $|A|$ when $|P| \leq 1$.

Proof:
The number of words in *MF*(*Fact*(P)) is the number of sink states created during the execution of Algorithm MF-TRIE on *DAWG*(P). We denote by A_P is the set of letters occurring in P. The number of words in *MF*(*Fact*(v)) is the number of sink states created during the execution of algorithm MF-TRIE. These states have exactly one incoming edge originated at a state of the automaton *DAWG*(P). So, we have to count these edges.

From the initial state of *DAWG*(P) there is exactly $|A| - |A_P|$ such edges. From the r states of *DAWG*(P) without outgoing edge, there are at most $|A_P|$ such edges. From other states there are at most $|A_P| - 1$ such edges.

For $|P| > 1$, it is known that *DAWG*(P) has at most $2|P| - 1$ states (see [3]). Therefore, we have that $|MF(Fact(P))| \leq (|A| - |A_P|) + r|A_P| + (2|P| - 2 - r)(|A_P| - 1) = (2|P| - 2)(|A_P| - 1) + |A| - |A_P| + r$. When $|P| \leq 1$, it can be checked directly that $|MF(Fact(P))| = |A|$. □

5. Checking whether a regular language is factorial or anti-factorial

In this section we address the problem of checking whether a regular language given by a deterministic finite state automaton is a factorial language or an anti-factorial language.

Proposition 5.1. Let $\mathcal{A} = (Q, A, i, T, \delta)$ be a trim finite deterministic automaton recognizing a regular language L. One can check whether L is a factorial language in time $O(|Q|^2 \times |A|)$.

Proof:
A trivial necessary condition for L to be factorial is that $T = Q$. We assume now that this is the case. Let p be any state of Q and $\mathcal{A}_p = (Q, A, p, Q, \delta)$. The automaton $\mathcal{A}_p$ recognizes a regular language L_p. It is straightforward that L is factorial if and only if $L_p \subset L$ for each $p \neq i \in Q$. Let us consider the product automaton $\mathcal{A}'$ of $\mathcal{A}$ by itself which set of states is $Q \times Q$ and initial states are $\{(i, p) \mid p \neq i \in Q\}$. The automaton $\mathcal{A}'$ has a deterministic labelling. The construction of $\mathcal{A}'$ can be performed in time $O(|Q|^2 \times |A|)$. Then L is not factorial if and only there exists a state (p, q) accessible from an initial state, and a letter a, such that $\delta(q, a)$ is defined and $\delta(p, a)$ is not. This takes again $O(|Q|^2 \times |A|)$ time. □

Proposition 5.2. Let $\mathcal{A} = (Q, A, i, T, \delta)$ be a trim finite deterministic automaton recognizing a regular language L. One can check whether L is an anti-factorial language in time $O(|Q|^2 \times |A|)$.

Proof:
Let p be any state of Q and $\mathcal{A}_p = (Q, A, p, Q, \delta)$. The automaton $\mathcal{A}_p$ recognizes a regular language L_p. It is straightforward that L is anti-factorial if and only if $L_p \cap L = \emptyset$ for each $p \neq i \in Q$ and L_i is a prefix language (no word of L_i is a prefix of another word of it). We consider again the product automaton $\mathcal{A}'$ of $\mathcal{A}$ by itself. Then L is anti-factorial if and only if there is no path in $\mathcal{A}'$ from a state (i, p) with $p \neq i$ to a state (q, r) with $q \in T$, and if each state $p \in T$ has no outgoing edges in $\mathcal{A}$. This takes $O(|Q|^2 \times |A|)$ time. □

6. An application to symbolic dynamics

We mention here an application of Section 2 to the study of sofic shifts. Sofic shifts are symbolic dynamical systems that can be described as the set of bi-infinite sequences labels of paths in a finite state automaton [12]. Thus a natural presentation of a sofic shift is a cover automaton. Two shifts are isomorphic, or conjugate, if and only if there is a bi-continuous one-to-one map from one shift onto the other. The decidability of the conjugacy of two sofic shifts is a well known open problem. A dynamic invariant related to the growth rate of the minimal forbidden words of the set of factors of the shifts has been described in [1]. If two sofic shifts have not the same growth rate of their minimal forbidden words, one can say that the two shifts are not conjugate. But one cannot conclude anything if the values of the two invariants are equal since the invariant is not complete.

We briefly recall the precise definition of this invariant and we show how Section 2 can be used to compute it, and to check whether two sofic shifts have the same invariant.

The enumerative sequence s_S of minimal forbidden words of the set of factors of a sofic shift S is a regular (or $\mathbb{N}$-rational) sequence. The corresponding series is then a rational series that can be written as a quotient of two polynomials P_S, Q_S with coefficients in $\mathbb{Z}$ and such that the $\gcd$ of P_S and Q_S is 1. If s_S is not a polynomial, the root of Q_S of minimal modulus is a positive real number denoted by $r(s_S)$.

The exponential growth rate of an enumerative sequence s is defined as the nonnegative real number $\limsup s_n^{1/n}$. The invariant is then the following. Two conjugate sofic shifts S and T have enumerative sequences s_S and s_T with the same exponential growth rate, i.e. satisfy $r(s_S) = r(s_T)$.

A consequence of Section 2 is that the enumerative sequence of minimal forbidden words of the set of factors of a sofic shift can be computed in quadratic time from a deterministic automaton recognizing the factors of the shift. It can then be checked in polynomial time whether two sofic shifts have the same invariant, i.e. whether $r(s_S) = r(s_T)$ or not. In the case where s_S and s_T are not polynomials, this is achieved by checking whether Q_S and Q_T have the same common maximal root. Without loss of generality, one can assume that the roots of Q_S and Q_T are simple. Since the sequences s_S and s_T are regular sequences, the maximal root of Q_S (resp. Q_T) is a positive real number. A polynomial time algorithm due to Davenport [9] (see [10]) and based on Sturm's theorem, provides, for an input that is a polynomial with integral coefficients, a sequence of rational intervals containing a single real root of the polynomial. Moreover, Sturm's theorem [10] enables a polynomial-time computation of the number of real roots contained in a given rational interval. The result follows from the computation of two sequences of rational intervals, one isolating the roots of Q_S, the other one the roots of Q_T. One then checks whether the $\gcd$ of Q_S and Q_T has a root inside each rightmost interval of the two sequences.

The same argument can be used to check whether two sofic shifts have the same topological entropy in polynomial time.

7. Acknowledgment

The authors thank Florent Hivert and Jean-Yves Thibon for helpful discussions about roots of polynomials. We also thank an anonymous reviewer for improving the proof of Proposition 3.1.

References

[1] Béal, M.-P., Mignosi, F., Restivo, A., Sciortino, M.: Forbidden words in symbolic dynamics, *Adv. in Appl. Math.*, **25**(2), 2000, 163–193.

[2] Béal, M.-P., Senellart, J.: On the bound of the synchronization delay of a local automaton, *Theoret. Comput. Sci.*, (205), 1998, 297–306.

[3] Blumer, A., Blumer, J., Haussler, D., McConnell, R., Ehrenfeucht, A.: Complete Inverted Files for Efficient Text Retrieval and Analysis, *Journal of the ACM*, **34**(3), 1987, 578–595.

[4] Carpi, A., de Luca, A., Varricchio, S.: Words, univalent factors and boxes, *Acta Informatica*, **38**(6), 2002, 409–436.

[5] Černý, J.: Poznámka k. homogénnym experimentom s konečnými automatmi, *Mat. fyz. čas.*, **SAV 14**, 1964, 208–215.

[6] Crochemore, M., Hancart, C., Lecroq, Th.: *Algorithmique du Texte*, Vuibert, Paris, 2001.

[7] Crochemore, M., Mignosi, F., Restivo, A.: Automata and forbidden words, *Inform. Process. Lett.*, **67**, 1998, 111–117.

[8] Crochemore, M., Mignosi, F., Restivo, A., Salemi, S.: Data compression using antidictionaries, in: *Proceedings of the IEEE, Lossless Data Compression* (J. Storer, Ed.), 2000, 1756–1768.

[9] Davenport, J. H.: Computer algebra for cylindrical algebraic decomposition, 1985, TRITA-NA-8511, NADA, KTH, Stockholm.

[10] Davenport, J. H., Siret, Y., Tournier, E.: *Computer Algebra*, Second edition, Academic Press Ltd., London, 1993, Systems and algorithms for algebraic computation, With a preface by Daniel Lazard, Translated from the French by A. Davenport and J. H. Davenport, With a foreword by Anthony C. Hearn.

[11] Hopcroft, J. E., Montwani, R., Ullman, J. D.: *Introduction to Automata Theory, Languages and Computation*, second edition, Addison-Wesley, 2001.

[12] Lind, D., Marcus, B.: *An Introduction to Symbolic Dynamics and Coding*, Cambridge University Press, Cambridge, 1995.

[13] Mignosi, F., Restivo, A., Sciortino, M.: Forbidden factors in finite and infinite words, in: *Jewels are Forever*, Springer, Berlin, 1999, 339–350.

[14] Mignosi, F., Restivo, A., Sciortino, M.: Forbidden Factors and Fragment Assembly, *RAIRO Theoret. Inform. Appl.*, **35**(6), 2001, 565–577, (Journal version).

[15] Raffinot, M.: *Structures pour la localisation de motifs*, Ph.D. Thesis, University of Marne-la-Vallée, France, 1999.