Theoretical Computer Science 108 (1993) 251–270
Elsevier

A generalization of Sardinas and Patterson's algorithm to z-codes

M. Madonia, S. Salemi and T. Sportelli

Dipartimento di Matematica ed Applicazioni, Università di Palermo, Italy

Communicated by D. Perrin
Received January 1990
Revised June 1991

Abstract

Madonia, M., S. Salemi and T. Sportelli, A generalization of Sardinas and Patterson's algorithm to z-codes, Theoretical Computer Science 108 (1993) 251–270.

This paper concerns the framework of z-codes theory. The main contribution consists in an extension of the algorithm of Sardinas and Patterson for deciding whether a finite set of words X is a z-code. To improve the efficiency of this test we have found a tight upper bound on the length of the shortest words that might have a double z-factorization over X. Some remarks on the complexity of the algorithm are also given. Moreover, a slight modification of this algorithm allows us to compute the z-deciphering delay of X.

1. Introduction

The theory of z-codes is strictly related to the study of the behaviour of a two-way automaton [5]. Recently, it has been developed, in an independent way, as a non-trivial generalization of theory of codes [1, 2, 4]. In this framework, important properties have been shown; in particular, the fact that z-codes give rise to recognizable sets has been proved in [1]. Another interesting aspect, in investigating properties of such z-codes, consists in the new point of view they introduce in combinatorics on words.

In this context, an algorithm for testing whether a rational set of words X is a z-code was given in [2]. Its implementation requires the construction of an automaton which recognizes the set $X^{\uparrow}$.

The main contribution of this paper is an algorithm which solves the problem in the case where X is a finite set; it is based on a suitable extension of the well-known test on codes due to Sardinas and Patterson [3].

Correspondence to: T. Sportelli, Dipartimento di Matematica ed Applicazioni, Via Archirafi 34, 90123 Palermo, Italy.

The paper is organized as follows: In Section 2, we give some definitions and preliminary results and we present the classical Sardinas and Patterson's algorithm. In Section 3, we describe the new algorithm, whose nature is essentially combinatorial, and we prove a theorem which gives a characterization of the z-codes and shows the correctness of the algorithm.

Section 4 is devoted to the complexity analysis of the algorithm. We find a tight upper bound on the length of the shortest words that might have a double z-factorization. This bound is related to the halt condition of the algorithm. Given a set $X=\{x_1, x_2, \ldots, x_n\}$, it is stated that the complexity of the algorithm is: $O(n^{2L\bar{m}})$, where $L=\sum_{i=1}^{n}|x_i|$ and $\bar{m}=\max\{|x_i| \mid i=1, \ldots, n\}$.

In Section 5, we introduce the new concept of z-deciphering delay and we shortly show that a slight modification of our algorithm allows us to compute the z-deciphering delay of a z-code.

The terminology and the notation adopted here conform to those introduced in previous papers on this topic. Nevertheless, the formal description of the algorithm might appear rather involved: this fact follows from the peculiar structure of z-factorizations, which have a combinatorial nature, apart from the novelty of the subject.

2. Definitions and preliminary results

In this section, some fundamental notations, definitions and general properties of z-codes are given. Moreover, it is recalled, by an example, the behaviour of the Sardinas and Patterson's algorithm applied to a finite set X.

2.1. *On z-codes theory*

Let A be a finite alphabet and A^* the free monoid generated by A. As usual, the elements of A^* are called words and the empty word is denoted by λ. Let $X \subseteq A^*$.

It is possible to define in $A^* \times A^*$ an equivalence relation generated by the set $T=\{((ux, v), (u, xv)) \mid u, v \in A^*, x \in X\}$.

We say that (u, v) *produces in only one step* (u', v'), and we denote this fact by $(u, v) \rightarrow (u', v')$, iff $((u, v), (u', v')) \in T$ or $((u', v'), (u, v)) \in T$. A step is said to be *to the right on* x if: $(u, xv) \rightarrow (ux, v)$; likewise, $(ux, v) \rightarrow (u, xv)$ is said to be a *step to the left on* x.

A *path* is a sequence of steps.

We denote the equivalence class of the pair (u, v) with $u \circledR v$. Given a set $X \subseteq A^*$, let $X^{\uparrow}=\{w \in A^* \mid \lambda \circledR w = w \circledR \lambda\}$.

In other words, $w \in A^*$ belongs to $X^{\uparrow}$ if there exists at least one finite path between the pairs (λ, w) and (w, λ). Notice that the first step, and the last step in the path are both steps to the right.

Definition 2.1. Given a word $w \in X^{\uparrow}$, a *z-factorization of w over X*, of length m, is a sequence of steps $(u_i, v_i) \to (u_{i+1}, v_{i+1})$, for $i = 1, 2, \ldots, m$, which verifies the following conditions:

(1) $u_1 = v_{m+1} = \lambda$,

(2) $v_1 = u_{m+1} = w$,

(3) $(u_j, v_j) \neq (u_k, v_k)$ for $j \neq k$.

Condition (3) is necessary to exclude the presence of "cycles" in the z-factorization.

Definition 2.2. A set $X \subseteq A^*$ is a *z-code* iff any word $w \in A^*$ has at most one z-factorization over X.

Remark 1. *The family of z-codes is strictly included in the family of codes*: indeed, if $X \subseteq A^*$ is a z-code, trivially it is a code; the converse is false: it suffices to consider $X_1 = \{b, a^2b, b^2a, a^2b^2a\}$; X_1 is a code on A^*, but it is not a z-code (see Example 1).

Trivially, *the family of z-codes is not empty*: indeed it contains the families of prefix and suffix codes and this is a strict inclusion, because there exist z-codes that are neither prefix nor suffix, as the set $X_2 = \{a^3ba^4, a^2b, b, ba\}$.

Moreover, z-codes may be regarded as basis of rational sets; this relevant property of z-codes has been stated by the theorem [1]: *for any recognizable $X \subseteq A^*$ there exists a deterministic automaton which recognizes $X^{\uparrow}$.*

2.2. *Sardinas and Patterson's algorithm*

Example 1. Let $A = \{a, b\}$ and let $X = \{b, aab, bba, aabba\}$. We test whether X is a code, by using the Sardinas and Patterson's algorithm.

We start by considering those words of X which are prefix of other words of X. In this way, we build a set $U_1 = X^{-1}X - \lambda$ that contains the suffixes (in bold in Fig. 1) which are usually called remainders.

In this case, any attempt to discover a double factorization, must take into account only the words of X^* that begin with "*bba*" and with "*aabba*"; therefore $U_1 = \{\boldsymbol{ba}\}$.

Now, the attempts to find a word w that might have a double factorization on X^* must be continued by checking:

(1) if the remainders (in U_1) are prefix of some words of X;

(2) if some words of X are prefix of the remainders (in U_1).

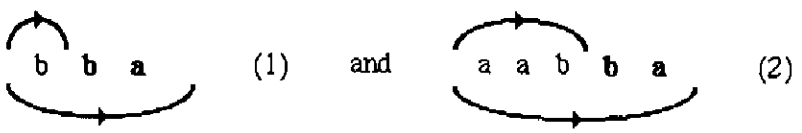

Fig. 1.

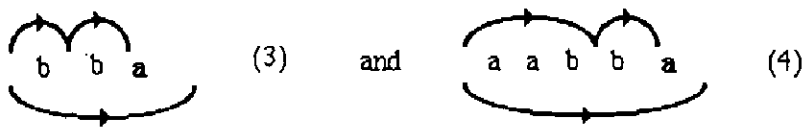

Fig. 2.

So, a sequence of sets of remainders is defined by induction whose general term is

$$U_{m+1} = X^{-1}U_m \cup U_m^{-1}X \quad \text{for } m \geqslant 1.$$

In our example, the decompositions (1) and (2) can be continued as shown in Fig. 2.

The second step of the implementation of the Sardinas and Patterson's algorithm gives rise to the set of remainders

$$U_2 = X^{-1}U_1 \cup U_1^{-1}X = \{\boldsymbol{a}\}.$$

In the same way, it is possible to find $U_3 = \{\boldsymbol{ab}, \boldsymbol{abba}\}$ and $U_4 = \emptyset$.

In [3] it is shown that the Sardinas and Patterson's algorithm always ends, because one of the three following cases must occur after a finite number of steps:

(a) $\exists m$ such that $\lambda \in U_m$. In other words, $\exists x \in U_{m-1}$ such that $x \in X$. In this case, X is not a code.

(b) $\exists m$ such that $U_m = \emptyset$. We can conclude that X is a code. This is the case of our example because $U_4 = \emptyset$.

(c) $\exists i$ such that $U_m = U_{m-i}$. Then $U_{m+1} = U_{m-i+1}$, and so on. Indeed, if $\lambda \notin U_h$ for any $h \leqslant m$, it follows that $\lambda \notin U_k$ for any $k > m$ and, also in this case, we can conclude that X is a code.

3. An algorithm for testing whether a set of words is a z-code.

In this section, a formal description of the algorithm for testing whether a finite set $X = \{x_1, x_2, \ldots, x_n\}$ of words is a z-code is given. This algorithm carries on all the attempts to find words with a double z-factorization over X.

In the following, it will be shown that while the implementation of the Sardinas and Patterson's algorithm produces a sequence of sets of words, the implementation of our algorithm produces a sequence of sets Q_m whose elements are tuples in $A^* \times A^* \times \{1, 2, \ldots, n\}^3$. Therefore, we characterize those sets X that are codes by considering some peculiarities of the sets Q_m.

3.1. Algorithm description

To formalize our algorithm, we need some new definitions and notations.

Definition 3.1. Let $X = \{x_1, x_2, \ldots, x_n\}$. We call *configuration* any tuple in $A^* \times A^* \times \{1, 2, \ldots, n\}^3$.

Definition 3.2. We say that a configuration $q=(l,r,i,j,k)$ produces on X a configuration $q'=(l_1,r_1,i_1,j_1,k_1)$, and we write $q_X \Rightarrow q'$, if there exists $x_h \in X$ such that

$$h \neq j, \quad r = x_h r_1, \quad l_1 = l x_h,$$
$$i_1 = h, \quad j_1 = 0, \quad k_1 = k;$$

or

$$h \neq j, \quad x_h = r r_1, \quad l_1 = l r,$$
$$i_1 = k, \quad j_1 = 0, \quad k_1 = h;$$

or

$$h \neq i, \quad r_1 = x_h r, \quad l_1 = l x_h^{-1},$$
$$i_1 = 0, \quad j_1 = h, \quad k_1 = k.$$

The sequence (Q_m) is defined by induction:

- Q_1 is the set of all configurations $(x_h, r, h, 0, k)$ such that $\exists x_h, x_k \in X$, with $x_k = x_h r$.
- for any integer m, Q_{m+1} is the set of the configurations produced on X by the elements of Q_m.

For any integer m, we denote by C_m the set $C_m = \{lr \in A^* \mid \exists (l,r,i,j,k) \in Q_m\}$ and by W_m the set $W_m = \{r \in A^* \mid \exists (l,r,i,j,k) \in Q_m\}$. We call W_m the set of the remainders of mth level.

Now we can give a module for the program development which shortly describes the tasks to be done by the proposed algorithm. The value K that occurs in the module is the upper bound on the length of the shortest words in A^* that might have two distinct z-factorizations over X. Notice that, for a given set X, this value is known [2].

In the following section we will find a tight upper bound K on the length of shortest words that might have a double z-factorization over X, and this improves the efficiency of our algorithm.

Algorithm

```
Begin
Read(X);
m ← 1;
build Q1, C1, W1;   {first step}
While (λ ∉ Wm) and (Qm ≠ ∅) and (for any lr ∈ Cm, |lr| < K) do
  begin
  m ← m + 1;
  build Qm, Cm, Wm;   {mth step}
  end;
If λ ∈ Wm then X is not a z-code
  else X is a z-code;
end.
```

Example 1 (Continued). Assume that the elements of X are numbered as follows:

$$x_1 = b, \qquad x_2 = aab, \qquad x_3 = bba, \qquad x_4 = aabba.$$

First step: Using the previous notation we obtain

$$Q_1 = \{(b, \boldsymbol{ba}, 1, 0, 3), (aab, \boldsymbol{ba}, 2, 0, 4)\},$$

corresponding to the attempts of z-factorizations as shown in Fig. 3. Therefore, $C_1 = \{\boldsymbol{bba}, \boldsymbol{abba}\}$ and $W_1 = \{\boldsymbol{ba}\}$.

Second step: The decompositions (1) and (2) may be continued as shown in Fig. 4. Formally,

$$(b, ba, 1, 0, 3) \underset{X}{\Rightarrow} (bb, a, 1, 0, 3),$$

$$(aab, ba, 2, 0, 4) \underset{X}{\Rightarrow} (aabb, a, 1, 0, 4),$$

$$(aab, ba, 2, 0, 4) \underset{X}{\Rightarrow} (aa, bba, 0, 1, 4).$$

Thus, we obtain

$$Q_2 = \{(bb, \boldsymbol{a}, 1, 0, 3), (aabb, \boldsymbol{a}, 1, 0, 4), (aa, \boldsymbol{bba}, 0, 1, 4)\},$$

$$C_2 = \{\boldsymbol{bba}, \boldsymbol{aabba}, \boldsymbol{aabba}\} \quad \text{and} \quad W_2 = \{\boldsymbol{a}, \boldsymbol{bba}\}.$$

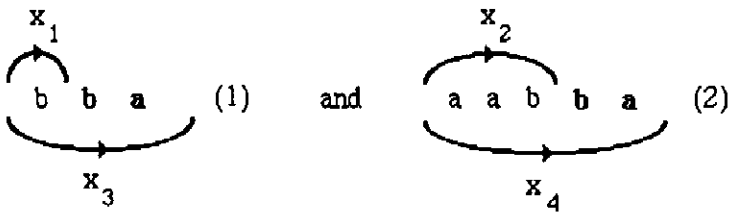

Fig. 3.

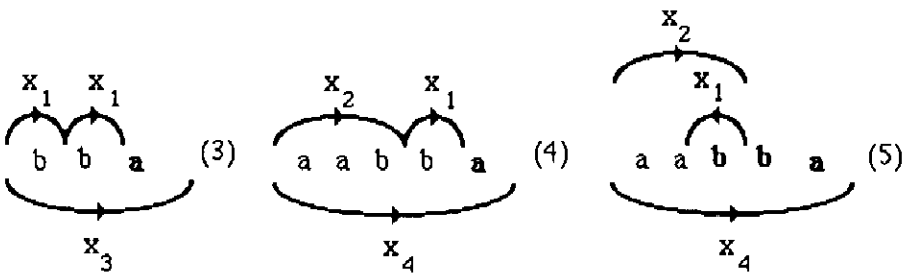

Fig. 4.

Third step: In the same way, the following sets are constructed:

$$Q_3 = \{(bba, \mathbf{ab}, 3, 0, 2), (bba, \mathbf{abba}, 3, 0, 4), (aabba, \lambda, 3, 0, 4), (aabba, \lambda, 4, 0, 3),$$
$$(aabba, \mathbf{ab}, 4, 0, 2), (aabba, \mathbf{abba}, 4, 0, 4)\};$$

$$C_3 = \{bba\mathbf{ab}, bba\mathbf{abba}, aabba, aabba\mathbf{ab}, aabba\mathbf{abba}\};$$

$$W_3 = \{\mathbf{ab}, \mathbf{abba}, \lambda\}.$$

Then, since $\lambda \in W_3$, we conclude that X is not a z-code.

3.2. The correctness of the algorithm

In order to prove Theorem 3.6, which states the correctness of the algorithm, we now give some definitions.

Definition 3.3. Given $v \in X^+$ and $w \in A^*$ such that w is a factor of v (i.e. $v = xwy$ with $x, y \in A^*$), a *partial z-factorization of v over X of length l* starting from the right (resp. left) of w is a sequence of steps $(u_i, v_i) \to (u_{i+1}, v_{i+1})$ $i = 1, \ldots, l$ such that

(1) $u_1 = xw$ and $v_1 = y$ (resp. $u_1 = x$ and $v_1 = wy$);
(2) $u_{l+1} = v$ and $v_{l+1} = \lambda$;
(3) $(u_j, v_j) \neq (u_k, v_k)$ for $j \neq k$.

Definition 3.4. Let $X = \{x_1, x_2, \ldots, x_n\}$. We say that the tuple (c, w, i, j, k), where $c \in A^*$, $w \in W_k$ and i, j, k are integers ≥ 0, satisfies condition (C1) on v if there exists a tuple $(c, w, f, s, t) \in Q_k$ such that the following three conditions hold:

(1) cw is a prefix of v;
(2) there exists a partial z-factorization of v over X of length i starting from the right of w and beginning with a step on x_r such that
   - if the step on x_r is a step to the left, then $t \neq r$;
(3) there exists a partial z-factorization of v over X of length j starting from the left of w and beginning with a step on x_l such that
   - if the step on x_l is a step to the left (resp. to the right), then $f \neq l$ (resp. $s \neq l$).

Lemma 3.5. *Let $X = \{x_1, x_2, \ldots, x_n\}$. For all $m \geq 1$, $\lambda \in W_m$ iff there exist a word $v \in C_m$ and a tuple (c, w, i, j, k) satisfying condition* (C1) *on v, with $k = m - i - j$.*

Proof. We prove the statement of the lemma by descending induction on k. First assume $k = m$. If $\lambda \in W_m$, then the tuple $(v, \lambda, 0, 0, m)$ satisfies condition (C1) on $v \in C_m$, with v the context of λ.

Conversely, if there exist $v \in C_m$ and a tuple (c, w, i, j, m) satisfying condition (C1) on v, then, since $K = m - i - j$, $i = j = 0$. This implies $w = \lambda$ and, consequently, $\lambda \in W_m$.

Now, let $m > k \geq 1$, and suppose that the sufficient condition of the lemma holds for $m, m-1, \ldots, k+1$.

If $\lambda \in W_m$, then, by induction hypothesis, there exist a $v \in C_m$ and a tuple $(c_1, u, i, j, k+1)$ satisfying condition (C1) on v. Therefore, a tuple $(c_1, u, f, s, t) \in Q_{k+1}$ exists and three cases may occur.

Case 1: There exists a word $x_h \in X$ and a tuple $(c, w, f', s', t') \in Q_k$ such that

$$x_h u = w \in W_k \quad \text{and} \quad h \neq s'.$$

In this case, the tuple $(c, w, i, j+1, k)$ satisfies condition (C1) on v. Indeed, as far as condition (2) of the Definition 3.4 is concerned, it suffices to consider the partial z-factorization of v over X of length i, starting from the right of u, taking into account that $t \neq t'$.

Moreover, as far as condition (3) of the Definition 3.4 is concerned, it suffices to add a step to the right on x_h at the left of the partial z-factorization of v of length j starting from the left of u.

Case 2: There exist an $x_h \in X$, a $w \in W_k$ and a tuple $(c, w, f', s', t') \in Q_k$ such that

$$wu = x_h \quad \text{and} \quad h \neq s'.$$

In this case, by using analogous considerations, we find that the tuple $(c, w, j, i+1, k)$ satisfies condition (C1) on v.

Case 3: there exist an $x_h \in X$, a $w \in W_k$ and a tuple $(c, w, f', s', t') \in Q_k$ such that

$$x_h w = u \quad \text{and} \quad h \neq f'.$$

Also in this case, the tuple $(c, w, i, j+1, k)$ satisfies condition (C1) on v.

Thus, the first part of the lemma is proved.

Conversely, suppose that there exist a word $v \in C_m$ and a tuple (c, w, i, j, k) satisfying condition (C1) on v. So, there exists a tuple $(c, w, f', s', t') \in Q_k$.

Without loss of generality, suppose that, in condition (3) of the Definition 3.4, the partial z-factorization of v begins with a step on $x_h \in X$. We shall prove that $\lambda \in W_m$.

If $j = 0$, then $i = 0$ and $k = m$, $w = \lambda$.

Thus, $j \geqslant 1$. Once more, we distinguish three cases.

Case 1: The step on x_h is a step to the right, and x_h is a prefix of w.

In this case, $w = x_h u$, with $u \in A^*$, and $h \neq s'$; then $u \in W_{k+1}$ and the tuple $(cx_h, u, i, j-1, k+1)$ satisfies condition (C1) on v.

Thus, $\lambda \in W_m$, by the induction hypothesis.

Case 2: The step on x_h is a step to the right and w is a prefix of x_h.

In this case, $x_h = wu$, with $u \in A^*$, and $h \neq s'$; then $u \in W_{k+1}$ and the tuple $(cx_h, u, j-1, i, k+1)$ satisfies condition (C1) on v.

Again, $\lambda \in W_m$ by induction hypothesis.

Case 3: The step on x_h is a step to the left, and x_h is a suffix of the context of w.

In this case, $cw = v' x_h w$, with $v' \in A^*$, and $f' \neq h$, then $x_h w \in W_{k+1}$ and the tuple $(cx_h^{-1}, x_h w, i, j-1, k+1)$ satisfies condition (C1) on v and, thus, $\lambda \in W_m$ by the induction hypothesis.

The proof is concluded. □

Now we can prove the following theorem.

Theorem 3.6. *The set X is a z-code iff none of the sets W_m defined above contains λ.*

Proof. If X is not a z-code, then there exists a word $v \in X^+$ such that v has two distinct z-factorizations over X. Let them be

$$(\lambda, v) \to (u_1, y_1) \to \cdots \to (u_i, y_i) \to (v, \lambda),$$

$$(\lambda, v) \to (z_1, t_1) \to \cdots \to (z_j, t_j) \to (v, \lambda),$$

with $u_h, y_h, z_k, t_k \in A^*$ for $h = 1, \ldots, i$, $k = 1, \ldots, j$.

Without loss of generality we assume that $z_1 = x_p$ and $u_1 = x_q$ with $x_p, x_q \in X$, and that $|z_1| < |u_1|$. Then $u_1 = z_1 w$ for some $w \in A^+$. Consequently, the tuple $(z_1, w, p, 0, q) \in Q_1$.

Moreover, the tuple $(z_1, w, i-1, j-1, 1)$ satisfies condition (C1) on $v \in C_{i+j-1}$. According to Lemma 3.5, $\lambda \in W_{i+j-1}$.

Conversely, if $\lambda \in W_m$, take, in Lemma 3.5, $k = 1$. Then, there exist a $v \in C_m$, a tuple $(c, w, i, j, 1)$ satisfying condition (C1) on v, and a tuple $(c, w, p, 0, q) \in Q_1$; so $x_p w = x_q$, for some $x_p, x_q \in X$, and v has two distinct z-factorizations over X. The first one begins with a step to the right on x_p, and it goes on with the partial z-factorization of v, of length j, starting from the left of w (note that, if the first step of this partial z-factorization is a step to the left on x_l, then $l \neq p$); the other one begins with a step to the right on x_q, and it goes on with the partial z-factorization of v, of length i, starting from the right of w (note that, if the first step of this partial z-factorization is a step to the left on x_r, then $r \neq q$).

This establishes the theorem. □

Remark 2. Notice that the algorithm always ends, after a finite number of steps.

Let us remember that the algorithm stops when the execution of the "while loop" terminates. Therefore, one of the three following conditions must fail:

(1a) $\lambda \notin W_m$;

(2a) $Q_m \neq \emptyset$;

(3a) for any $lr \in C_m$, $|lr| < K$.

If X is not a z-code, Theorem 3.6 assures us that:

(1b) $\exists$ an integer m such that $\lambda \in W_m$ and the execution of the "while loop" stops because condition (1a) fails. In particular, the context of λ provides a word which has a double z-factorization over X.

This is the case of our example: indeed, $\lambda \in W_3$ and the word $w = aabba$ has two distinct z-factorization over X.

If X is a z-code, either

(2b) $\exists$ an integer m such that $Q_m = \emptyset$, and the execution of the "while loop" stops because condition (2a) fails; or

(3b) $\exists$ an integer m such that, for any $n < m$, $\lambda \notin W_n$ and for any $l_m r_m \in C_m$, $|l_m r_m| \geq K$, where K is the upper bound on the length of the shortest words in A^* that might have two distinct z-factorization over X (see [2]).

Indeed, we will show (see the lemmas of the following section) that after m executions of the "while loop" body, if condition (2a) has not failed, the length of any word $w \in Q_m$ is $\geqslant m$.

This assures us that, in the worst case, after K steps condition (3a) fails, and then the execution of the "while loop" stops.

4. Complexity of the algorithm

A new upper bound on the length of the shortest words that might have a double z-factorization has been found (see Proposition 4.6), in order to improve the efficiency of the algorithm. This bound is tight. It has been found by taking into account not only the lengths of the words of the finite set X, but also their alphabetic structure.

At the end of this section we derive the complexity of the algorithm.

4.1. Some new definitions

From now on, let $X=\{x_1, x_2, \ldots, x_n\} \subseteq A^*$ be a finite set and let $\mathrm{Card}(X)=n$.

For any $w \in A^*$ and $a \in A$, $|w|_a$ denotes the number of occurrences of the letter a in the word w.

For any $X=\{x_1, x_2, \ldots, x_n\}$ and $a \in A$, we set

$$|X|_a = \sum_{i=1}^{n} |x_i|_a.$$

Definition 4.1. Let $X=\{x_1, x_2, \ldots, x_n\}$, let $w \in X^+$ and let f be a z-factorization of w over X. Then for $k=1, \ldots, |w|$, we say that *a step of f on x_i*, $(u', x_i v') \to (u' x_i, v')$ or $(u' x_i, v') \to (u', x_i v')$, *crosses the kth position of w*, if $|u'| < k$ and $|u' x_i| \geqslant k$. In this case we also say that x_i crosses the kth position of w in f in its $(K-|u'|)$th position.

Definition 4.2. Let $X=\{x_1, x_2, \ldots, x_n\}$, let $w \in X^+$ and let f be a z-factorization of w over X. Then for $k=1, \ldots, |w|$ we define

$C_f(k)=\{(i,j) \mid$ at least one of the following steps occurs in f:

- $(u, x_i v) \to (u x_i, v)$,
- $(u x_i, v) \to (u, x_i v)$, where $u, v \in A^*$ and $|u|=k-j\}$.

Example 2. Let $X=\{x_1, x_2, x_3, x_4\}=\{a^3ba^4, a^2b, b, ba\}$. Let us consider the word $w=aaba \in X^+$ and its z-factorization f:

$$(\lambda, w)=(\lambda, aaba) \to (aab, a) \to (aa, ba) \to (aaba, \lambda)=(w, \lambda).$$

Then:

$$C_f(1)=\{(2,1)\}, \qquad C_f(2)=\{(2,2)\},$$

$$C_f(3)=\{(2,3), (3,1), (4,1)\}, \qquad C_f(4)=\{(4,2)\}.$$

Remark 3. If $w \in X^+$ and $\mathcal{f}$ is a z-factorization of w, then, for any $k = 1, \ldots, |w|$, $\mathrm{Card}(C_{\mathcal{f}}(k))$ is equal to the number of steps of $\mathcal{f}$ that cross the kth position of w. Therefore, $\mathrm{Card}(C_{\mathcal{f}}(k))$ is an odd number.

4.2. An upper bound on the length of the shortest ambiguous words

From now on, let $X = \{x_1, x_2, \ldots, x_n\}$ be a set that is not a z-code and let $w \in X^+$ be a word of minimal length that has two distinct z-factorizations over X, $\mathcal{f}_1$ and $\mathcal{f}_2$.

The following three lemmas state some conditions that are essential for the proof of Proposition 4.6, which gives an upper bound on $|w|$.

In particular, Lemma 4.3 shows that $\mathcal{f}_1$ and $\mathcal{f}_2$ do not have "coinciding cuts".

In Lemma 4.4, an upper bound is given to the number of times in which any letter of w can be crossed in $\mathcal{f}_1$ and $\mathcal{f}_2$.

Lemma 4.5 states that the sets $C_{\mathcal{f}_1}(k) \cup C_{\mathcal{f}_2}(k)$ are different from each other for every position k of w ($1 \leqslant k \leqslant |w|$).

The proofs of these lemmas are rather technical; therefore, in order to avoid the thread of the problem to be lost, we choose to postpone them to the end of the paper (see the Appendix).

Lemma 4.3. *If the pair (u, v) occurs in $\mathcal{f}_1$, with $u, v \in A^+$, then (u, v) does not occur in $\mathcal{f}_2$.*

Based on this lemma, we have the following remark.

Remark 4. For $k = 1, \ldots, |w|$ we have $C_{\mathcal{f}_1}(k) \cap C_{\mathcal{f}_2}(k) = \emptyset$.

Suppose $C_{\mathcal{f}_1}(k) \cap C_{\mathcal{f}_2}(k) \neq \emptyset$ and let $(i, j) \in C_{\mathcal{f}_1}(k) \cap C_{\mathcal{f}_2}(k)$.

Then, in both the z-factorizations $\mathcal{f}_1$ and $\mathcal{f}_2$, there is at least one of the following steps:

- $(u, x_i v) \to (u x_i, v)$,
- $(u x_i, v) \to (u, x_i v)$, where $u, v \in A^*$ and $|u| = k - j$.

But this implies that there exists a pair ($(u, x_i v)$ or $(u x_i, v)$) that occurs both in $\mathcal{f}_1$ and in $\mathcal{f}_2$, contradicting Lemma 4.3. From Remark 3, and from $C_{\mathcal{f}_1}(k) \cap C_{\mathcal{f}_2}(k) = \emptyset$, it follows that $\mathrm{Card}(C_{\mathcal{f}_1}(k) \cup C_{\mathcal{f}_2}(k))$, for $k = 1, \ldots, |w|$, is an even number.

Lemma 4.4. *For $k = 1, \ldots, |w|$, we have*

$$\mathrm{Card}(C_{\mathcal{f}_1}(k) \cup C_{\mathcal{f}_2}(k)) \leqslant 2\bar{m} - 2,$$

where $\bar{m} = \max\{|x_i| \mid i = 1, \ldots, n\}$.

Remark 5. For $k = 1, \ldots, |w|$, we have

$$\mathrm{Card}(C_{\mathcal{f}_1}(k) \cup C_{\mathcal{f}_2}(k)) \leqslant |X|_a,$$

where a is the letter which occurs in the kth position of w.

Lemma 4.5. *For $h, k = 1, \ldots, |w|$, we have*

$$C_{f_1}(h) \cup C_{f_2}(h) = C_{f_1}(k) \cup C_{f_2}(k) \Leftrightarrow h = k.$$

Now we can state the following proposition.

Proposition 4.6. *Let $X = \{x_1, x_2, \ldots, x_n\}$ be a subset of A^* that is not a z-code and let w be a word of minimal length that has two distinct z-factorizations over X. Then*

$$|w| \leqslant \sum_{a \in A} \sum_{i=1}^{p_a} \binom{|X|_a}{2i} = K,$$

where $p_a = \min\{(\lfloor |X|_a/2 \rfloor), (\bar{m}-1)\}$ and $\bar{m} = \max\{|x_i| \mid i = 1, \ldots, n\}$.

Proof. From Remarks 4 and 5, and from Lemma 4.4, it follows that if k is a position of w in which the letter a occurs, then the maximum number of different sets $\{C_{f_1}(k) \cup C_{f_2}(k)\}$ is

$$\sum_{i=1}^{p_a} \binom{|X|_a}{2i},$$

where $p_a = \min\{(\lfloor |X|_a/2 \rfloor), (\bar{m}-1)\}$ and $\bar{m} = \max\{|x_i| \mid i = 1, \ldots, n\}$.

Then, from Lemma 4.5, the proposition holds. □

Remark 6. The previous upper bound on the length of shortest words of X^1, that might have two distinct z-factorizations over X, is tight; indeed, it is actually reached in some particular cases; for example, let $X = \{ab, abc, def, cdef\}$. X is not a z-code (in particular it is not a code) and $w = abcdef$ is a word of minimal length that has two distinct z-factorizations over X. Indeed,

$$\sum_{a \in A} \sum_{i=1}^{p_a} \binom{|X|_a}{2i} = 6 \sum_{i=1}^{1} \binom{2}{2i} = 6 = |w|.$$

4.3. A result of complexity

Given $X = \{x_1, x_2, \ldots, x_n\}$, let $L = \sum_{i=1}^{n} |x_i|$ the length of X. The implementation of our algorithm on X goes on by construction of a sequence of sets Q_m of configurations. In order to give a brief analysis of the algorithm, we choose to represent its computation by a tree; then, we give an upper bound on the number of nodes of this tree. To go further into details:

- the root of the tree is the set X;
- each node corresponds to a configuration;
  all the nodes of mth level, taken as a whole, represent the set Q_m of configurations that are generated at the mth step of the algorithm; indeed, the sons of a node q are all the configurations produced by q in one step; in particular, if q is a leaf of the tree, this means that no configuration can be derived from q.

It follows that, the depth d of the tree is equal to the number of executions of the "while loop" of the algorithm.

In the worst case (see Remark 2) the "while loop" body is executed K times, where K is the previous upper bound.

From Section 4.2 we can derive:

$$d \leqslant K \leqslant L^2 + L^4 + L^6 + \cdots + L^{b_x} = p(L),$$

where

$$b_x = \min\{2\bar{m}, h\},\ \bar{m} = \max\{|x_i| \mid i = 1, \ldots, n\} \text{ and } h = \max\{|X|_a \mid a \in A\}.$$

Notice that the number of nodes of the first level of the tree is at most $O(n^2)$ and that, starting from this level, any node of the tree has at most $2n$ sons (n corresponding to possible steps to the right and n corresponding to possible steps to the left). Therefore, in the worst case, that corresponds to a complete tree of degree $2n$, the number of nodes is $O(n^2((2n)^{p(L)} - 1)/(2n - 1))$, i.e. $O(2^{p(L)-1} n^{p(L)+1})$.

Moreover, by considering the relations $2^{p(L)} \leqslant n^{p(L)}$ and $p(L) \sim O(L^{\bar{m}})$, it is possible to conclude that the number of nodes is $O(n^{2L^{\bar{m}}})$.

In the construction of the sets Q_m, we now consider as elementary operation the comparison between two strings. Then, we state the following theorem.

Theorem 4.7. *Given* $X = \{x_1, x_2, \ldots, x_n\}$, *the complexity of the algorithm is* $O(n^{2L^{\bar{m}}})$, *where* $L = \sum_{i=1}^{n} |x_i|$ *and* $\bar{m} = \max\{|x_i| \mid i = 1, \ldots, n\}$.

Notice that if X is not a code, then the algorithm stops after the same number of steps that are requested in Sardinas and Patterson's algorithm, although, in the generalized algorithm, more complications are involved.

5. Further development

In this section, we first define the new concept of z-deciphering delay for z-codes; this notion is analogous to the one regarding codes (see [3]): given $X \subseteq A^*$, its z-deciphering delay may be finite or infinite. In the first case, the "delay" between the moment when a possible step of a z-factorization over X is discovered, and the moment when these steps are definitively valid, is bounded.

We give a method to compute the z-deciphering delay for a given z-code X by the implementation of our extension of Sardinas and Patterson's algorithm.

Definition 5.1. Let $X \subseteq A^*$. Given a word $w \in X^{\uparrow}$, a *quasi z-factorization of w over X*, of length m, is a sequence of steps $(u_i, v_i) \to (u_{i+1}, v_{i+1})$ for $i = 1, 2, \ldots, m$ which satisfies the following conditions:

(1) $u_1 = \lambda$;

(2) $v_1 = w$;

(3) $(u_j, v_j) \neq (u_k, v_k)$ for $j \neq k$.

Definition 5.2. Let $X \subseteq A^*$. We say that X has a *bounded z-deciphering delay* if there exists $d>0$ such that for any $x_1, x_2, \ldots, x_r, y_1, y_2, \ldots, y_s \in X$ and $w \in X^\uparrow$, *if*:

(1) there exists a z-factorization of w of length s,

$$(\lambda, w)=(u_0, v_0)\to(u_1, v_1)\to\cdots\to(u_s, v_s)=(w, \lambda),$$

where $(u_i, v_i)\to(u_{i+1}, v_{i+1})$ is a step on y_i, for $i=1, \ldots, s$;

(2) there exists a quasi-z-factorization of w of length r,

$$(\lambda, w)=(u'_0, v'_0)\to(u'_1, v'_1)\to\cdots\to(u'_r, v'_r),$$

where $(u'_i, v'_i)\to(u'_{i+1}, v'_{i+1})$ is a step on x_i, for $i=1, \ldots, r$;

(3) $|w|-\max\{|v'_i|\,|\,i=1, \ldots, r\}<|y_s|$; and

(4) $r+s>d$;

then, $x_1=y_1$.

Let $X \subseteq A^*$. The *z-deciphering delay of* X, $d(X)$, is the smallest integer d satisfying the previous conditions, if such a d exists, otherwise it is infinite.

Example 3. Let $X=\{ba, ab^2a, a^2bab^2, ab^2ab^2, bab^2a^2ba\}$. X has an infinite z-deciphering delay. Indeed, it suffices to consider the word $w=abb(abbaab)^\omega$.

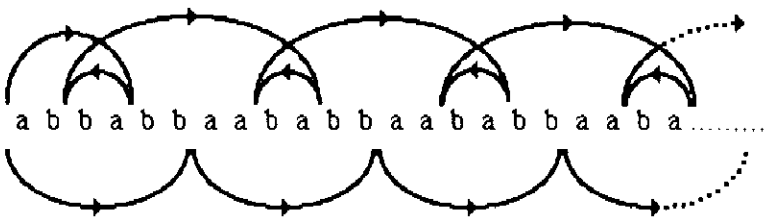

Fig. 5.

Remark 7. It is not very hard to prove that the algorithm allows us to check if a finite z-code $X=\{x_1, x_2, \ldots, x_n\}$ has a finite z-deciphering delay or not. In the first case, it also computes the finite z-deciphering delay $d(X)$. Indeed, if X is a z-code, then, in the generalized Sardinas and Patterson's algorithm, one of the following halt conditions must hold:

(1) $\exists m$ such that $Q_m=\emptyset$.

In this case, $d(X)$ is finite and $d(X)=m$.

(2) $\exists m$ such that for any $lr \in C_m$, $|lr| \geqslant K$, where

$$K=\sum_{a\in A}\sum_{i=1}^{p_a}\binom{|X|_a}{2i} \quad \text{(see Proposition 4.6).}$$

In this case, it suffices to construct the sets Q_i, C_i, W_i until one of the following two cases occurs:

(i) $\exists i>m$ such that $Q_i=\emptyset$.

Also in this case, the z-deciphering delay is finite, in particular $d(X)=i$.

(ii) $\exists i > m$ and $lr \in Ci$ such that $|l| \geqslant K$, where

$$K = \sum_{a \in A} \sum_{i=1}^{p_a} \binom{|X|_a}{2i}$$

In this case $d(X)$ is infinite.

Acknowledgment

The authors are grateful to Prof. D. Perrin for the fruitful discussions and useful comments. They also wish to thank the anonymous referees for their helpful suggestions.

Appendix

Proof of Lemma 4.3. Let us consider the two distinct z-factorizations of w:

$$\mathcal{f}_1: (\lambda, w) = (u_0, v_0) \to (u_1, v_1) \to \cdots \to (u_r, v_r) \to \cdots \to (u_k, v_k) = (w, \lambda),$$

$$\mathcal{f}_2: (\lambda, w) = (u'_0, v'_0) \to (u'_1, v'_1) \to \cdots \to (u'_s, v'_s) \to \cdots \to (u'_h, v'_h) = (w, \lambda),$$

and suppose that $(u, v) = (u_r, v_r) = (u'_s, v'_s)$, with $u, v \in A^+$, $r, s > 0$, $k > r$ and $h > s$.

Moreover, suppose that the following condition holds:

$$r = s \quad \text{and} \quad (u_i, v_i) = (u'_i, v'_i) \quad \text{for } i = 1, \ldots, r \tag{$*$}$$

Let us consider the set $S = S_1 \cup S_2$, where $S_1 = \{v_i \mid r \leqslant i \leqslant k\}$ and $S_2 = \{v'_i \mid s \leqslant i \leqslant h\}$.

Since $r, s > 0$, any element of S is a proper suffix of w. Let v be the longest word of S. Suppose that $v \in S_1$ and $v = v_t$ with $r \leqslant t < k$ (the same considerations hold if $v \in S_2$).

Thus, $v_t = u_t^{-1} w$ and the word v_t has two distinct z-factorizations $\mathcal{f}'_1$ and $\mathcal{f}'_2$: the first one is derived from the last $(k-t)$ steps of $\mathcal{f}_1$, the second one begins with the $(t-r)$ steps of $\mathcal{f}_1$ and goes on with the other $(h-s)$ steps of $\mathcal{f}_2$. Formally, (see also the example in Fig. 6):

$$\mathcal{f}'_1: (\lambda, v_t) = (u_t^{-1} u_t, v_t) \to (u_t^{-1} u_{t+1}, v_{t+1}) \to \cdots \to (u_t^{-1} u_k, v_k) = (u_t^{-1} w, \lambda) = (v_t, \lambda),$$

$$\begin{aligned}\mathcal{f}'_2: (\lambda, v_t) = (u_t^{-1} u_t, v_t) &\to (u_t^{-1} u_{t-1}, v_{t-1}) \to (u_t^{-1} u_{t-2}, v_{t-2}) \to \cdots \\ &\to (u_t^{-1} u_r, v_r) = (u_t^{-1} u'_s, v'_s) \to (u_t^{-1} u'_{s+1}, v'_{s+1}) \to \cdots \\ &\to (u_t^{-1} u'_h, v'_h) = (u_t^{-1} w, \lambda) = (v_t, \lambda).\end{aligned}$$

(Note that, if $(w', w'') \to (w'_1, w''_1)$ is a step to the right (left) on x, then $(w'_1, w''_1) \to (w', w'')$ is a step to the left (right) on x).

In Fig. 6, $\mathcal{f}'_1$ is visualized by the bold line and $\mathcal{f}'_2$ by the dotted line.

Since the steps following the pth step in $\mathcal{f}'_1$ and in $\mathcal{f}'_2$ are surely different (if $p = \max\{t, s\}$), then $\mathcal{f}'_1 \neq \mathcal{f}'_2$ against the minimality of w.

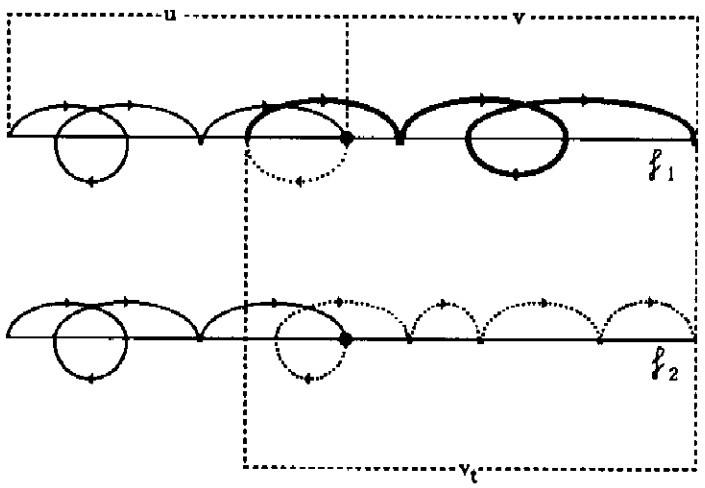

Fig. 6.

If the condition (*) does not hold, let us consider the set $P=P_1\cup P_2$, where

$$P_1=\{u_i \mid 1\leqslant i\leqslant r\} \quad \text{and} \quad P_2=(u'_i \mid 1\leqslant i\leqslant s\}.$$

Since $k>r$ and $h>s$, any element of P is a proper prefix of w. Let u be the longest word of P. Suppose that $u\in P_1$ and $u=u_t$ with $1\leqslant t\leqslant r$ (the same considerations hold if $u\in P_2$).

Thus, $u_t=wv_t^{-1}$ and the word u_t has two distinct z-factorizations f'_1 and f'_2: the first one is derived from the first t steps of f_1, the second one begins with the s steps of f_2 and goes on with the other $(r-t)$ steps of f_1. Formally (see also the example in Fig. 7):

$$\begin{aligned} f'_1\colon\ (\lambda,u_t)&=(\lambda,wv_t^{-1})=(u_0,v_0v_t^{-1})\to(u_1,v_1v_t^{-1})\to\cdots\\ &\to(u_t,v_tv_t^{-1})=(u_t,\lambda),\\ f'_2\colon\ (\lambda,u_t)&=(\lambda,wv_t^{-1})=(u'_0,v'_0v_t^{-1})\to(u'_1,v'_1v_t^{-1})\to\cdots\\ &\to(u'_s,v'_sv_t^{-1})=(u_r,v_rv_t^{-1})\to(u_{r-1},v_{r-1}v_t^{-1})\to(u_{r-2},v_{r-2}v_t^{-1})\to\cdots\\ &\to(u_t,v_tv_t^{-1})=(u_t,\lambda). \end{aligned}$$

In Fig. 7, f'_1 is visualized by the bold line and f'_2 by the dotted line.

Since the first p steps in f'_1 and in f'_2 are surely different (if $p=\min\{t,s\}$), then $f'_1\neq f'_2$ against the minimality of w, and the lemma is proved. □

Proof of Lemma 4.4. Let us first remark that for any z-factorization of w, f, and for any $(i,j)\in C_f(k)$, we have $1\leqslant j\leqslant m$.

If $\mathrm{Card}(C_{f_1}(k)\cup C_{f_2}(k))>2\bar{m}$, then there exist at least three distinct elements (i',j'), $(i'',j''),(i''',j''')\in C_{f_1}(k)\cup C_{f_2}(k)$ such that $j'=j''=j'''$. This implies that, in f_1 and f_2, there exist three steps $(u_1,v_1)\to(u_2,v_2)$, $(u_3,v_3)\to(u_4,v_4)$, $(u_5,v_5)\to(u_6,v_6)$ such that

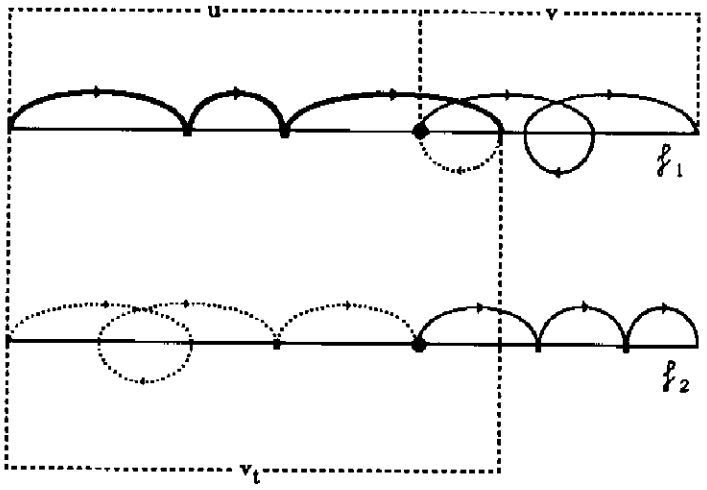

Fig. 7.

$u_r = u_s = u_t$ and $|u_r| = |u_s| = |u_t| = k - j'$ for suitable r, s, t and $1 \leqslant r \neq s \neq t \leqslant 6$. But at most two, among the pairs (u_r, v_r), (u_s, v_s) and (u_t, v_t), can occur in the same z-factorization of w, and this implies that the third pair occurs in the other z-factorization of w, contradicting Lemma 4.3. Thus, $\mathrm{Card}(C_{f_1}(k) \cup C_{f_2}(k)) \leqslant 2\bar{m}$.

Now, if $\mathrm{Card}(C_{f_1}(k) \cup C_{f_2}(k)) = 2\bar{m}$, two cases may occur:

(1) There exist $j \in \{1, \ldots, \bar{m}\}$ and three distinct elements $(i', j'), (i'', j''), (i''', j''') \in C_{f_1}(k) \cup C_{f_2}(k)$ such that $j = j' = j'' = j'''$.

In this case, as we have just seen, we have a contradiction.

(2) For any $j = 1, \ldots, \bar{m}$ there exist two distinct elements (i', j'), (i'', j'') in $C_{f_1}(k) \cup C_{f_2}(k)$ such that $j = j' = j''$. In particular, for $j = \bar{m}$, we are sure that, in $C_{f_1}(k) \cup C_{f_2}(k)$, there are two distinct elements, (i', j') and (i'', j''), such that $j' = j'' = \bar{m}$.

Since $\bar{m} = \max\{|x_i| \mid i = 1, \ldots, n\}$, it follows that:

there are two elements $x_{i'}, x_{i''} \in X$ such that $|x_{i'}| = |x_{i''}| = \bar{m}$;

in f_1 and f_2, there are two steps:

$$(u, x_{i'} v) \to (ux_{i'}, v) \quad (\text{or } (ux_{i'}, v) \to (u, x_{i'} v)) \quad \text{and}$$

$$(u, x_{i''} v) \to (ux_{i''}, v) \quad (\text{or } (ux_{i''}, v) \to (u, x_{i''} v)),$$

such that $|u| = k - \bar{m}$ and $|ux_{i'}| = |ux_{i''}| = k$.

But this implies that $x_{i'} = x_{i''}$ and, therefore, there exists a pair occurring twice in f_1 and f_2 contradicting Lemma 4.3. □

Proof of Lemma 4.5. Suppose that there exists $w = uyv$, with $u, v \in A^*$, $y \in A^+$, $|u| = h$, $|uy| = k$, such that: $C_{f_1}(h) \cup C_{f_2}(h) = C_{f_1}(k) \cup C_{f_2}(k)$.

Now we show that the word $w_1 = uv_1$, where v_1 is a prefix (proper or not) of v, has two distinct z-factorizations against the hypothesis that w is the shortest word which has a double z-factorization.

Let $x_{i_1}, x_{i_2}, \ldots, x_{i_r}$ be the sequence of words of X that cross the hth position of w in f_1; let $x_{i_{r+1}}, x_{i_{r+2}}, \ldots, x_{i_s}$ be the sequence of words of X that cross the hth position of w in f_2; let $x_{j_1}, x_{j_2}, \ldots, x_{j_p}$ be the sequence of words of X that cross the kth position of w in f_1; let $x_{j_{p+1}}, x_{j_{p+2}}, \ldots, x_{j_q}$ be the sequence of words of X that cross the kth position of w in f_2. Note that:

- for any z, such that $1 \leqslant z < r$ or $(r+1) \leqslant z < s$, there exists, in f_1 or in f_2, a path from the step on x_{i_z}, that crosses the hth position of w, to the step on $x_{i_{z+1}}$, that crosses the hth position of w and
- for any z, such that $1 < z \leqslant r$ or $(r+1) < z \leqslant s$ there exists a path from the step on x_{i_z}, that crosses the hth position of w, to the step on $x_{i_{z-1}}$, that crosses the hth position of w.

  Likewise, in f_1 or in f_2:

  for any t, such that $1 \leqslant t < p$ or $(p+1) \leqslant t < q$, there exists a path from the step on x_{j_t}, that crosses the kth position of w, to the step on $x_{j_{t+1}}$, that crosses the kth position of w and

  for any t, such that $1 < t \leqslant p$ or $(p+1) < t \leqslant q$, there exists a path from the step on x_{j_t}, that crosses the kth position of w, to the step on $x_{j_{t-1}}$, that crosses the kth position of w.

Moreover, in f_1 (f_2), there exists a path from the first step of f_1 (f_2) to the step on x_{i_1} ($x_{i_{r+1}}$), that crosses the hth position of w, and there exists a path from the step on x_{j_p} (x_{j_q}), that crosses the kth position of w, to the last step of f_1 (f_2).

Since $C_{f_1}(h) \cup C_{f_2}(h) = C_{f_1}(k) \cup C_{f_2}(k)$, we have:

- $q = s$ and the sequence $j_1, j_2, \ldots, j_p, j_{p+1}, \ldots, j_q$ is a permutation of the sequence $i_1, i_2, \ldots, i_r, i_{r+1}, \ldots, i_s$;
- for any step on x_i, $1 \leqslant i \leqslant n$, that crosses the hth position of w in f_1 or in f_2 (for example $(u', x_i v') \to (u' x_i, v')$ with $|u' x_i| = h + j$ for a suitable $j > 0$), there exists, in f_1 or in f_2, a corresponding step on x_i, that crosses the kth position of w (for example $(u'' x_i, v'') \to (u'', x_i v'')$, such that $|u'' x_i| = k + j$).

Now, let us give another definition: for any pair $(u', u''yv)$, that occurs in f_1 or in f_2, we call the pair $(u', u''v)$ the reduction on y of $(u', u''yv)$ and, likewise, for any pair (uyv', v'') that occurs in f_1 or in f_2, we call the pair (uv', v'') the reduction on y of (uyv', v''). In other words, when we consider the reduction on y of a pair that occurs in f_1 or f_2, we "rub out" the factor y from w.

In order to find two distinct z-factorizations of w_1, we shall construct two paths $\not{f}_1$ and $\not{f}_2$ as follows.

To construct $\not{f}_1$, we consider the first steps of f_1 until we find the step on x_{i_1} that crosses the hth position of w. At this point, we look for the step corresponding to this one and that crosses the kth position of w. Let us suppose that it is a step on x_{j_t}. First, we consider the two following cases:

(1) If $j_t = j_p$ then $\not{f}_1$ goes on with the path from the step on x_{j_p} that crosses the kth position of w to the last step of f_1.

(2) If $j_t = j_q$ then $\not\mu_1$ goes on with the path from the step on x_{j_q} that crosses the kth position of w to the last step of $\not\mu_2$.

In these two cases, we find one z-factorization of uv, by the reductions on y of any pair that occurs in $\not\mu_1$ and then we can continue by the construction of the path $\not\mu_2$.

If $j_t \neq j_p$ and $j_t \neq j_q$, then the path $\not\mu_1$ goes on, either:

(3) with the path from the step on x_{j_t} that crosses the kth position of w to the step on $x_{j_{t+1}}$ that crosses the kth position of w, if the step on x_{i_1} (crossing the hth position of w) and the corresponding step on x_{j_t} (crossing the kth position of w) have both the same direction; or:

(4) with the path from the step on x_{j_t} that crosses the kth position of w to the step on $x_{j_{t-1}}$ that crosses the kth position of w, if the step on x_{i_1} (crossing the hth position of w) and the corresponding step on x_{j_t} (crossing the kth position of w) have different directions.

In the case (3), we look for the step, that crosses the hth position of w, and that corresponds to the step on $x_{j_{t+1}}$; let us suppose that it is a step on x_{i_z};

In the case (4) we look for the step, that crosses the hth position of w, and that corresponds to the step on $x_{j_{t-1}}$; let us suppose that it is a step on x_{i_z}.

First, we consider the case with $i_z = i_{r+1}$.

(5) If $i_z = i_{r+1}$, then $\not\mu_1$ goes on with the path from the step on $x_{i_{r+1}}$ to the first step of $\not\mu_2$.

In this case, we find two distinct z-factorizations of $w_1 = uv_1$, where v_1 is a proper prefix of v, by the reductions on y of any pair that occurs in $\not\mu_1$ and we can stop.

If $i_z \neq i_{r+1}$, then the path $\not\mu_1$ goes on either:

(3b) with the path from the step on x_{i_z}, that crosses the hth position of w, to the step on $x_{i_{z+1}}$, that crosses the hth position of w, if the step on x_{i_z} (crossing the hth position of w) and its corresponding step (crossing the kth position of w) have both the same direction; or:

(4b) with the path from the step on x_{i_z}, that crosses the hth position of w, to the step on $x_{i_{z-1}}$, that crosses the hth position of w, if the step on x_{i_z} (crossing the hth position of w) and its corresponding step (crossing the kth position of w) have different directions. We continue the construction of $\not\mu_1$ in this way, every time looking for corresponding steps, until case (1) or case (2) or case (5) occurs.

At this point, if we have not stopped, we construct $\not\mu_2$ as $\not\mu_1$, starting from the first step of $\not\mu_2$, and again, either:

- we arrive to the first step of $\not\mu_1$ and, in this case, we find, by the reductions on y of any pair that occurs in $\not\mu_2$, two distinct z-factorizations of $w_1 = uv_1$, where v_1 is a proper prefix of v; or:
- we find, by the reductions on y of any pair that occurs in $\not\mu_2$, another z-factorization of uv. This other z-factorization of uv is distinct from the previous one (it suffices to note that, at least, the first steps of these z-factorizations of uv are different).

Thus, the lemma is proved. ||

References

[1] M. Anselmo, Automates et codes zigzag, *RAIRO Inform. Théor. Appl.* **25**(1) (1991).
[2] M. Anselmo, Sur les codes zigzag et leur décidabilité, Report LITP 89.36.
[3] J. Berstel and D. Perrin, *Theory of Codes* (Academic Press, New York, 1985).
[4] M. Madonia, S. Salemi and T. Sportelli, On z-submonoids and z-codes, *RAIRO Inform. Théor. Appl.* **25**(4) (1991).
[5] J.P. Pécuchet, Automates boustrophédons, langages reconnaisables de mots infinis et variétés de semigroupes, Thèse d'Etat, LITP, 1986.