Theoretical Computer Science 82 (1991) 35-49
Elsevier

An algorithm for distinguishing efficiently bit-strings by their subsequences

Jean-Jacques Hebrard

LIUC, Université de Caen, 14032 Caen Cedex, France

Communicated by D. Perrin
Received May 1988
Revised February 1989

Abstract

Hebrard, J.-J., An algorithm for distinguishing efficiently bit-strings by their subsequences, Theoretical Computer Science 82 (1991) 35-49.

A linear on-line algorithm for computing a shortest subsequence that distinguishes two different bit-strings is presented. The method is based on a special way of factorizing strings.

0. Introduction

A string h divides a string u if it can be obtained from u by deleting zero or more symbols. If a string h divides u (resp. v) and does not divide v (resp. u) we say that h distinguishes u and v. The similarity of two strings u and v can be studied by comparing the strings they are divided by. For example several similarity measures are based on the computation of a longest string dividing u and v [2, 4, 5, 6]. One can also consider as a measure of similarity the greatest integer $d(u, v)$ such that no string of length $\leqslant d(u, v)$ can distinguish u and v. This paper is devoted to the computation of $d(u, v)$.

Various algorithms have been proposed for this problem. Simon [7] presented an algorithm with time and space complexity $O(|A|\,|uv|)$, where A is the alphabet. Unfortunately this algorithm is not on-line and requires a large pre-processing needing a lot of space. Another method uses the finite automaton which accepts the set of all the strings that divide a given string. This leads to an almost linear algorithm [1].

We present a new method based on a special factorization of u and v which we call the arch factorization. If u and v are bit-strings the arch factorization provides an efficient linear algorithm to compute $d(u, v)$. This algorithm is on-line and only requires a constant amount of extra space. Moreover the method gives the construction of a shortest string that distinguishes u and v.

1. Basic definitions and notations

Given a finite set of symbols A, a string u is a finite sequence $u(1)\dots u(n)$ of elements of A; the length of u is denoted by $|u|$. The empty string is denoted by ε and the set of all strings over A by A^*. By $\text{alph}(u)$ we mean the set of symbols which occur in u. The concatenation of two strings u and v is denoted by u.v.

Given a string $u(1)\dots u(n)$, $\text{pref}(u, i)$ and $\text{suf}(u, i)$ denote respectively $u(1)\dots u(i)$ and $u(i+1)\dots u(n)$. We have $u = \text{pref}(u, i).\text{suf}(u, i)$.

A string h *divides* u if there exists a subsequence of u $u(s(1))\dots u(s(m))$ such that $h = u(s(1))\dots u(s(m))$. s is said to be the *first occurrence* of h in u if for every subsequence $u(t(1))\dots u(t(m))$ such that $h = u(t(1))\dots u(t(m))$, we have $s(i) \leq t(i)$ $(1 \leq i \leq m)$. A string h *distinguishes* two strings u and v if it divides one of them and does not divide the other. $D(u, v)$ denotes the set of the shortest strings that distinguish u and v.

Given a string u and an integer l let $S(u, l) = \{h \in A^* \mid h \text{ divides } u \text{ and } |h| \leq l\}$. Given two strings u and v, $d(u, v)$ is defined by

$$d(u, v) = \begin{cases} \max\{l \mid S(u, l) = S(v, l)\} & \text{if } u \neq v, \\ \infty & \text{otherwise.} \end{cases}$$

$d(u, v)$ is the greatest integer such that no string of length $\leq d(u, v)$ can distinguish u and v. One can show that $\delta(u, v) = 2^{-d(u,v)}$ is an ultrametric distance [3]. If h is member of $D(u, v)$ then $|h| = d(u, v) + 1$. The following sections deal with the computation of $d(u, v)$ and $D(u, v)$.

2. Arch factorization

Our method is based on a special way of factorizing strings.

Definition. Let u be a string over A. $u = \text{ar}_u(1)\dots \text{ar}_u(n).r(u)$ is the *arch factorization* of u if:

(i) for every $I \in \{1, \dots, n\}$, $\text{ar}_u(I) = c_u(I).u(p_u(I))$ with $c_u(I) \in A^*$, $p_u(I) \in \{1, \dots, |u|\}$, $\text{alph}(c_u(I)) \neq A$ and $\text{alph}(\text{ar}_u(I)) = A$.

(ii) $\text{alph}(r(u)) \neq A$.

The strings $\text{ar}_u(I)$ will be called the *archs* of u and $r(u)$ the *rest*. The string $u(p_u(1)).u(p_u(2))\dots u(p_u(n))$ will be called the *model* of u and denoted by $m[u]$.

Example

$A = \{0, 1, 2\}, \quad u = 1022011210010210,$

$u = 102.201.1210.0102.10,$

$\mathrm{ar}_u(1) = 102, \quad u(p_u(1)) = u(3) = 2,$

$\mathrm{ar}_u(2) = 201, \quad u(p_u(2)) = u(6) = 1,$

$\mathrm{ar}_u(3) = 1210, \quad u(p_u(3)) = u(10) = 0,$

$\mathrm{ar}_u(4) = 0102, \quad u(p_u(4)) = u(14) = 2,$

$r(u) = 10$ and $m[u] = 2102.$

We show that every string shorter than $|m[u]|$ divides u and that it is always possible to construct a string of length $|m[u]|+1$ which does not divide u.

Proposition 1. *Let u and h be two strings over A. If $|h| \leq |m[u]|$ then h divides u.*

Proof. $\forall i \in \{1, \ldots, |h|\}$, $h(i)$ divides $\mathrm{ar}_u(i)$. □

Proposition 2. *Let u be a string over A and $a \in A \setminus \mathrm{alph}(r(u))$. Then $m[u].a$ does not divide u.*

Proof. We have $m[u] = u(p_u(1)) \ldots u(p_u(n))$. The result follows from the fact that p_u is the first occurrence of $m[u]$ in u. □

The two following propositions show that the arch factorization provides an easy tool to compute $d(u, v)$ when the models $m[u]$ and $m[v]$ have different lengths or $\mathrm{alph}(r(u))$ is different from $\mathrm{alph}(r(v))$. In both cases we have $d(u, v) = \min(|m[u]|, |m[v]|)$.

Proposition 3. *Let u and v be two strings over A such that $|m[u]| < |m[v]|$ and let $a \in A \setminus \mathrm{alph}(r(u))$. Then $d(u, v) = |m[u]|$ and $m[u].a \in D(u, v)$.*

Proof. Let h be a string over A. If $|h| \leq |m[u]|$, h divides u and v. The string $m[u].a$ does not divide u (Proposition 2), it divides v since $|m[u].a| \leq |m[v]|$. □

Proposition 4. *Let u and v be two strings over A such that $|m[u]| = |m[v]| = n$ and $\mathrm{alph}(r(v)) \setminus \mathrm{alph}(r(u)) \neq \emptyset$. Let $a \in \mathrm{alph}(r(v)) \setminus \mathrm{alph}(r(u))$. Then $d(u, v) = n$ and $m[u].a \in D(u, v)$.*

Proof. Let h be a string over A. If $|h| \leq |m[u]|$, h divides u and v. The string $m[u].a$ does not divide u, it divides v since $m[u]$ divides $\mathrm{ar}_v(1) \ldots \mathrm{ar}_v(n)$ and a divides $r(v)$. □

We must now show how to compute $d(u, v)$ when $|m[u]| = |m[v]|$ and $\text{alph}(r(u)) = \text{alph}(r(v))$. This will only be done for bit-strings.

3. The case of bit-strings

In this section u and v are strings over $\{0, 1\}$. We first examine the situation where $|m[u]| = |m[v]|$, $\text{alph}(r(u)) = \text{alph}(r(v))$ and $m[u] \neq m[v]$. The following proposition says that in this case $d(u, v) = |m[u]| = |m[v]|$. This result is proved by considering the greatest k such that $u(p_u(k)) \neq v(p_v(k))$. To make things more concrete let us examine the following examples:

(1) $u = 01011100010$ and $v = 10001010100$

$$u = 0\ \underline{1}.\quad 0\ \underline{1}.\ 1\ 1\ \underline{0}.\ 0\ 0\ \underline{1}.\ 0 \qquad m[u] = 1101,$$

$$v = \underline{1}\ 0.\ 0\ 0\ \underline{1}.\quad \underline{0}\ \underline{1}.\quad 0\ \underline{1}.\ 0\ 0 \qquad m[v] = 0111.$$

We have $k = 3$. The string $w = m[u].1 = 11011$ divides v and does not divide u. The third arch enables w to run more quickly through u than through v.

(2) $u = 001100$ and $v = 01010$

$$u = 0\ 0\ \underline{1}.\ 1\ \underline{0}.\ 0 \qquad m[u] = 10,$$

$$v = \quad 0\ \underline{1}.\ \underline{0}\ \underline{1}.\ 0 \qquad m[v] = 11.$$

We have $k = 2 = |m[u]|$ and $m[u].1 = 101$ distinguishes u and v.

Proposition 5. *Let u and v be two strings over $\{0, 1\}$ such that $|m[u]| = |m[v]| = n$, $\text{alph}(r(u)) = \text{alph}(r(v)) = \mathcal{R}$ and $m[u] \neq m[v]$. Then $n > 0$ and $d(u, v) = n$. Moreover if k $(1 \leqslant k \leqslant n)$ is the greatest integer such that $u(p_u(k)) \neq v(p_v(k))$ we have:*

(i) *if $k < n$ and $a \in \{0, 1\} \setminus \mathcal{R}$:*
- *either $u(p_u(k)) \neq u(p_u(k+1))$ and $m[u].a \in D(u, v)$,*
- *or $v(p_v(k)) \neq v(p_v(k+1))$ and $m[v].a \in D(u, v)$.*

(ii) *if $k = n$:*
- *either $v(p_v(n)) \notin \mathcal{R}$ and $m[u].v(p_v(n)) \in D(u, v)$,*
- *or $u(p_u(n)) \notin \mathcal{R}$ and $m[v].u(p_u(n)) \in D(u, v)$.*

Proof. (i) $k < n$. We have $u(p_u(k)) \neq v(p_v(k))$ and $u(p_u(k+1)) = v(p_v(k+1))$. Suppose $u(p_u(k)) = 0$, $v(p_v(k)) = 1$, $u(p_u(k+1)) = 1$ and $v(p_v(k+1)) = 1$. The string $m[u].a$ distinguishes u and v. It does not divide u (Proposition 2). $\text{pref}(m[u], k-1)$ divides $\text{ar}_v(1) \ldots \text{ar}_v(k-1)$ (Proposition 1), $u(p_u(k)).u(p_u(k+1))$ divides $\text{ar}_v(k)$ since $\text{ar}_v(k) = 0^q 1 (q > 0)$, and $\text{suf}(m[u], k+1).a$ divides $\text{ar}_v(k+1) \ldots \text{ar}_v(n)$ (Proposition 1). Consequently $m[u].a$ divides v.

(ii) $k = n$. We may suppose $u(p_u(n)) = 0$ and $v(p_v(n)) = 1$. Either $u(p_u(n))$ or $v(p_v(n))$ does not belong to $\mathcal{R}$. Suppose $v(p_v(n))$ does not belong to $\mathcal{R}$. The string $m[u].v(p_v(n))$ distinguishes u and v. It does not divide u (Proposition 2).

pref($m[u], n-1$) divides $\mathrm{ar}_v(1)\ldots\mathrm{ar}_v(n-1)$ (Proposition 1), and $u(p_u(n)).v(p_v(n))$ divides $\mathrm{ar}_v(n)$ since $\mathrm{ar}_v(n)=0^q.1$ ($q>0$). Therefore $m[u].v(p_v(n))$ divides v. □

Remark. Note that Proposition 5 does not hold if card(A) > 2. For example when $A=\{0,1,2\}$, $u=20101012$ and $v=21010102$ we have $|m[u]|=|m[v]|=2$, $m[u]=12$, $m[v]=02$, but $d(u,v)=3$ ($S(u,3)=S(v,3)$ and 1110 distinguishes u and v).

In order to study the only case that has not yet been considered, namely $m[u]=m[v]$ and $\mathrm{alph}(r(u))=\mathrm{alph}(r(v))$, we need some new definitions and notations.

Definition. Given two bit-strings u and v such that $|m[u]|=|m[v]|=n$ with $n>0$ and $|\mathrm{ar}_u(I)|<|\mathrm{ar}_v(I)|$ for some $I\in\{1,\ldots,n\}$, disting(I) is the string: $\mathrm{pref}(m[u],I-1).c_u(I).m[\mathrm{suf}(u,p_u(I)-1)].a$ with $a\in\{0,1\}\setminus\mathrm{alph}(u(|u|))$.

Example

$u=110.01.1110.1$, $m[u]=010$, $\mathrm{alph}(r(u))=\{1\}$,

$v=1110.01.110.11$, $m[v]=010$, $\mathrm{alph}(r(v))=\{1\}$,

$|\mathrm{ar}_u(1)|<|\mathrm{ar}_v(1)|$, $\mathrm{pref}(m[u],0)=\varepsilon$, $c_u(1)=11$,

$m[\mathrm{suf}(u,2)]=m[00111101]=10$, $a=0$,

disting(1) = 11100.

$|\mathrm{ar}_v(3)|<|\mathrm{ar}_u(3)|$, $\mathrm{pref}(m[v],2)=01$, $c_v(3)=11$,

$m[\mathrm{suf}(v,9)]=m[011]=1$, $a=0$,

disting(3) = 011110.

We shall see (Proposition 9) that disting(I) distinguishes u and v. In order to prove this result we must first thoroughly examine the string $m[\mathrm{suf}(u,p_u(I)-1)]$. The following lemma shows that its properties depend on whether conditions $u(p_u(I)-1)\neq u(p_u(I+1)-1)$ and $|\mathrm{ar}_u(I+1)|>2$ hold.

Lemma 6. *Let u be a string over $\{0,1\}$ such that $|m[u]|=n$ and $n>0$. Let $I<n$.*

(i) *If $u(p_u(n)-1)\notin\mathrm{alph}(r(u))$ then $m[\mathrm{suf}(u,p_u(n)-1)]=\varepsilon$.*

(ii) *If $u(p_u(n)-1)\in\mathrm{alph}(r(u))$ then $m[\mathrm{suf}(u,p_u(n)-1)]=u(p_u(n)-1)$.*

(iii) *If $u(p_u(I)-1)\neq u(p_u(I+1)-1)$ then*

$$m[\mathrm{suf}(u,p_u(I)-1)]=u(p_u(I)-1).m[\mathrm{suf}(u,p_u(I+1))].$$

(iv) *If $u(p_u(I)-1)=u(p_u(I+1)-1)$ and $|\mathrm{ar}_u(I+1)|=2$ then*

$$m[\mathrm{suf}(u,p_u(I)-1)]=u(p_u(I)-1).m[\mathrm{suf}(u,p_u(I+1)-1)].$$

(v) *If $u(p_u(I)-1)=u(p_u(I+1)-1)$ and $|\mathrm{ar}_u(I+1)|>2$ then*

$$m[\mathrm{suf}(u,p_u(I)-1)]=u(p_u(I)-1).u(p_u(I)).m[\mathrm{suf}(u,p_u(I+1))].$$

Proof. (i) Suppose $u(p_u(n)-1)=0$. If $u(p_u(n)-1)\notin \mathrm{alph}(r(u))$ then $\mathrm{suf}(u, p_u(n)-1)=1^k (k>0)$, and $m[\mathrm{suf}(u, p_u(n)-1)]=\varepsilon$.

(ii) Suppose $u(p_u(n)-1)=0$. If $u(p_u(n)-1)\in \mathrm{alph}(r(u))$ then $\mathrm{suf}(u, p_u(n)-1)=10^k (k>0)$, and $m[\mathrm{suf}(u, p_u(n)-1)]=0$.

(iii) Suppose $u(p_u(I)-1)=0$ and $u(p_u(I+1)-1)=1$. We have $u(p_u(I))=1$, $\mathrm{ar}_u(I+1)=1^k 0 (k>0)$, and $\mathrm{suf}(u, p_u(I)-1)=1^{k+1}0.\mathrm{suf}(u, p_u(I+1))$. Then $m[\mathrm{suf}(u, p_u(I)-1)]=0.m[\mathrm{suf}(u, p_u(I+1))]$.

(iv) Suppose $u(p_u(I)-1)=u(p_u(I+1)-1)=0$. Then $\mathrm{ar}_u(I+1)=01$, $\mathrm{suf}(u, p_u(I)-1)=10.\mathrm{suf}(u, p_u(I+1)-1)$ and

$$m[\mathrm{suf}(u, p_u(I)-1)]=0.m[\mathrm{suf}(u, p_u(I+1)-1)].$$

(v) Suppose $u(p_u(I)-1)=u(p_u(I+1)-1)=0$. Then $\mathrm{ar}_u(I+1)=00^k 1$ $(k>0)$, $\mathrm{suf}(u, p_u(I)-1)=100^k 1.\mathrm{suf}(u, p_u(I+1))$ and

$$m[\mathrm{suf}(u, p_u(I)-1)]=01.m[\mathrm{suf}(u, p_u(I+1)-1]. \quad \square$$

Examples. Here $I=1$, $p_u(I)-1=2$, $u(p_u(I)-1)=0$ and $w\in\{0,1\}^*$.

(i) $u=001$, $m[\mathrm{suf}(u,2)]=m[1]=\varepsilon$; $u=001.1$, $m[\mathrm{suf}(u,2)]=m[11]=\varepsilon$.

(ii) $u=001.0$, $m[\mathrm{suf}(u,2)]=m[10]=0$.

(iii) $u=001.110.w$, $m[\mathrm{suf}(u,2)]=m[1110.w]=0.m[w]$.

(iv) $u=001.01.w$, $m[\mathrm{suf}(u,2)]=m[101.w]=0.m[1.w]$.

(v) $u=001.001.w$, $m[\mathrm{suf}(u,2)]=m[1001.w]=01.m[w]$.

Notations. Given $I\in\{1,\dots,|m[u]|\}$, $F_u(I)$ (resp. $G_u(I)$) denotes the smallest J such that $J>I$ and $u(p_u(J)-1)\neq u(p_u(I)-1)$ (resp. $|\mathrm{ar}_u(J)|>2$). For every $I\in\{1,\dots,|m[u]|\}$, let $\mathcal{F}_u(I)=\{I<J\leq|m[u]| \,/\, u(p_u(J)-1)\neq u(p_u(I)-1)\}$ and $\mathcal{G}_u(I)=\{I<J\leq|m[u]| \,/\, |\mathrm{ar}_u(J)|>2\}$. If $\mathcal{F}_u(I)\neq\emptyset$ then $F_u(I)=\min \mathcal{F}_u(I)$ otherwise $F_u(I)=\infty$, if $\mathcal{G}_u(I)\neq\emptyset$ then $G_u(I)=\min \mathcal{G}_u(I)$ otherwise $G_u(I)=\infty$.

Example

$$u=01.001.110.10.0$$

$$F_u(1)=3, \quad G_u(1)=2, \quad F_u(2)=3, \quad G_u(2)=3,$$

$$F_u(3)=G_u(3)=\infty, \quad F_u(4)=G_u(4)=\infty.$$

The next lemma shows that $m[\mathrm{suf}(u, p_u(I)-1)]$ can be written $a^p b^q.\mathrm{suf}(m[u], K)$ with $a, b\in\{0,1\}$ and $a=u(p_u(I)-1)$. The values of p, q and K depend on $F_u(I)$ and $G_u(I)$. Let us see that first on examples.

Here $I=1$, $p_u(1)-1=1$ and $u(p_u(1)-1)=0$:

(1) $u=01.01.110.w$, $F_u(1)=G_u(1)=3$. We have $F_u(1)\leq G_u(1)$ and

$$m[\mathrm{suf}(u, p_u(1)-1)]=m[10.1110.w]=0^2.m[w]=0^2.\mathrm{suf}(m[u],3).$$

(2) $u=01.01.001.w$, $G_u(1)=3$, $F_u(1)>3$. We have $G_u(1)<F_u(1)$ and

$$m[\mathrm{suf}(u, p_u(1)-1)]=m[10.10.01.w]=0^2 1.\mathrm{suf}(m[u],3).$$

(3) $u = 01.01.111$,

$$F_u(1) = G_u(1) = \infty \quad \text{and} \quad u(p_u(1) - 1) \notin \text{alph}(r(u)),$$

$$m[\text{suf}(u, p_u(1) - 1)] = m[10.1111] = 0.$$

(4) $u = 01.01.000$,

$$F_u(1) = G_u(1) = \infty \quad \text{and} \quad u(p_u(1) - 1) \in \text{alph}(r(u)),$$

$$m[\text{suf}(u, p_u(1) - 1)] = m[10.10.00] = 0^2.$$

Lemma 7. *Let u be a string over $\{0, 1\}$ such that $|m[u]| = n$ and $n > 0$. Let $I \in \{1, \ldots, n\}$ and $a = u(p_u(I) - 1)$.*

(i) *If $F_u(I) \neq \infty$ and $F_u(I) \leq G_u(I)$ then*

$$m[\text{suf}(u, p_u(I) - 1)] = a^{J-I}\text{suf}(m[u], J) \quad \textit{where } J = F_u(I).$$

(ii) *If $G_u(I) < F_u(I)$ then $m[\text{suf}(u, p_u(I) - 1)] = a^{J-I}b.\text{suf}(m[u], J)$ where $J = G_u(I)$ and $b = u(p_u(I))$.*

(iii) *If $F_u(I) = G_u(I) = \infty$ and $a \notin \text{alph}(r(u))$ then $m[\text{suf}(u, p_u(I) - 1)] = a^{n-I}$.*

(iv) *If $F_u(I) = G_u(I) = \infty$ and $a \in \text{alph}(r(u))$ then $m[\text{suf}(u, p_u(I) - 1)] = a^{n-I+1}$.*

Proof

(i) $m[\text{suf}(u, p_u(I) - 1)] = a^{J-I-1}.m[\text{suf}(u, p_u(J - 1) - 1]$ (Lemma 6(iv))

$= a^{J-I}.m[\text{suf}(u, p_u(J))]$ (Lemma 6(iii))

$= a^{J-I}.\text{suf}(m[u], J)$,

(ii) $m[\text{suf}(u, p_u(I) - 1)] = a^{J-I-1}.m[\text{suf}(u, p_u(J - 1) - 1)]$ (Lemma 6(iv))

$= a^{J-I}b.m[\text{suf}(u, p_u(J))]$ (Lemma 6(v))

$= a^{J-I}b.\text{suf}(m[u], J)$,

(iii) $m[\text{suf}(u, p_u(I) - 1)] = a^{n-I}.m[\text{suf}(u, p_u(n) - 1)]$ (Lemma 6(iv))

$= a^{n-I}$, (Lemma 6(i))

(iv) $m[\text{suf}(u, p_u(I) - 1)] = a^{n-I}.m[\text{suf}(u, p_u(n) - 1)]$ (Lemma 6(iv))

$= a^{n-I+1}$ (Lemma 6(ii)). □

Definition. Given two bit-strings u and v such that $|m[u]| = |m[v]| = n$ with $n > 0$ and $|\text{ar}_u(I)| < |\text{ar}_v(I)|$ for some $I \in \{1, \ldots, n\}$, we say that arch I *satisfies* $\mathcal{P}$ if $[F_u(I) \neq \infty$ and $F_u(I) \leq G_u(I)]$ or $[F_u(I) = G_u(I) = \infty$ and $u(p_u(I) - 1) \notin \text{alph}(r(u))]$.

As a consequence of Lemma 7 the length of disting(I) can be easily calculated.

Proposition 8. *Let u and v be two bit-strings such that $|m[u]| = |m[v]| = n$ with $n > 0$ and $|\mathrm{ar}_u(I)| < |\mathrm{ar}_v(I)|$ for some $I \in \{1, \ldots, n\}$. Then*

$$|\mathrm{disting}(I)| = \begin{cases} n + |\mathrm{ar}_u(I)| - 1 & \text{if arch } I \text{ satisfies } \mathcal{P}, \\ n + |\mathrm{ar}_u(I)| & \text{otherwise.} \end{cases}$$

Proof

$$\begin{aligned} |\mathrm{disting}(I)| &= |\mathrm{pref}(m[u], I-1)| + |c_u(I)| + |m[\mathrm{suf}(u, p_u(I)-1)]| + 1 \\ &= I - 1 + |\mathrm{ar}_u(I)| - 1 + |m[\mathrm{suf}(u, p_u(I)-1)]| + 1. \end{aligned}$$

It follows from Lemma 7 that $|m[\mathrm{suf}(u, p_u(I)-1)]| = n - I$ if arch I satisfies $\mathcal{P}$ and $|m[\mathrm{suf}(u, p_u(I)-1)]| = n - I + 1$ otherwise. □

The next proposition shows that two bit-strings u and v such that $m[u] = m[v]$ and $\mathrm{alph}(r(u)) = \mathrm{alph}(r(v))$, are distinguished by disting(I). Consider the following example:

$$u = 110.01.1110.1, \qquad m[u] = 010, \mathrm{alph}(r(u)) = \{1\},$$

$$v = 1110.01.110.11, \qquad m[v] = 010, \mathrm{alph}(r(v)) = \{1\},$$

$|\mathrm{ar}_u(1)| < |\mathrm{ar}_v(1)|$, disting(1) = 11100 divides v and does not divide u:

$$u = \underline{1}\ \underline{1}\ 0\ .\ 0\ \underline{1}\ .\ 1\ 1\ 1\ \underline{0}\ .\ 1\ \_$$

$$v = \underline{1}\ \underline{1}\ \underline{1}\ \underline{0}\ .\ \underline{0}\ 1\ .\ 1\ 1\ 0\ .\ 1\ 1$$

$|\mathrm{ar}_v(3)| < |\mathrm{ar}_u(3)|$, disting(3) = 011110 divides u and does not divide v:

$$u = 1\ 1\ \underline{0}\ .\ 0\ \underline{1}\ .\ \underline{1}\ \underline{1}\ \underline{1}\ \underline{0}\ .\ 1$$

$$v = 1\ 1\ 1\ \underline{0}\ .\ 0\ \underline{1}\ .\ \underline{1}\ \underline{1}\ 0\ .\ \underline{1}\ 1\ \_$$

Notice that u and v can also be distinguished by their rests provided that $|r(u)| \neq |r(v)|$. In the above example we have $|r(u)| < |r(v)|$ and the string $m[u].r(u).1 = 01011$ divides v but does not divide u.

Proposition 9. *Let u and v be strings over $\{0, 1\}$ such that $m[u] = m[v]$ and $\mathrm{alph}(r(u)) = \mathrm{alph}(r(v))$. Let $n = |m[u]| = m|[v]|$.*

(i) *If $n > 0$ and there exists $I \in \{1, \ldots, n\}$ such that $|\mathrm{ar}_u(I)| < |\mathrm{ar}_v(I)|$, then* disting($I$) *divides v but does not divide u.*

(ii) *If $|r(u)| < |r(v)|$ and $a \in \mathrm{alph}(r(v))$, then $m[u].r(u).a$ divides v but does not divide u.*

Proof. (i) $\text{disting}(I) = \text{pref}(m[u], I-1).c_u(I).m[\text{suf}(u, p_u(I)-1)].a$ with $a \in \{0, 1\} \setminus \text{alph}(u(|u|))$. It follows from Proposition 2 that $\text{disting}(I)$ does not divide u since $a \in \{0, 1\} \setminus \text{alph}(r(\text{suf}(u, p_u(I)-1)))$. Let us show that $\text{disting}(I)$ divides v. We can suppose $c_u(I) = 0^k$, $\text{ar}_u(I) = 0^k 1$ and $\text{ar}_v(I) = 0^{k+l}1$ with $l > 0$. We have $\text{pref}(v, p_v(I-1)+k) = \text{ar}_v(1) \ldots \text{ar}_v(I-1).c_u(I)$ and therefore $\text{pref}(m[u], I-1).c_u(I)$ divides $\text{pref}(v, p_v(I-1)+k)$. We must now show that $m[\text{suf}(u, p_u(I)-1)].a$ divides $\text{suf}(v, p_v(I-1)+k)$. Three cases must be considered.

Case 1. arch I satisfies $\mathscr{P}$. It follows from Lemma 7(i) and (iii) that

$$\begin{aligned} |m[\text{suf}(u, p_u(I)-1)].a| = n - I + 1.|m[\text{suf}(v, p_v(I-1)+k)]| \\ = |m[0^l 1.\text{suf}(v, p_v(I))]| = |1.m[\text{suf}(v, p_v(I))]| \\ = |1.\text{suf}(m[v], I)| = n - I + 1. \end{aligned}$$

Then $m[\text{suf}(u, p_u(I)-1)].a$ divides $\text{suf}(v, p_v(I-1)+k)$ (Proposition 1).

Case 2. $G_u(I) < F_u(I)$. It follows from Lemma 7(ii) that

$$m[\text{suf}(u, p_u(I)-1)].a = 0^{J-I}1.\text{suf}(m[u], J).a$$

with

$$J = G_u(I).\text{suf}(v, p_v(I-1)+k) = 0^l 1.\text{ar}_v(I+1) \ldots \text{ar}_v(J-1).\text{suf}(v, p_v(J-1)).$$

For every $K \in \{I+1, \ldots, J-1\}$, $\text{ar}_u(K) = 01$ and $\text{ar}_v(K)$ has the form $0^q 1$ ($q > 0$) since $m[u] = m[v]$. Then $0^{J-I}1$ divides $0^l 1.\text{ar}_v(I+1) \ldots \text{ar}_v(J-1)$. It follows from Proposition 1 that $\text{suf}(m[u], J).a$ divides $\text{suf}(v, p_v(J-1))$ since $|\text{suf}(m[u], J).a| = n - J + 1$ and $|m[\text{suf}(v, p_v(J-1))]| = |\text{suf}(m[v], J-1)| = n - J + 1$.

Case 3. $F_u(I) = G_u(I) = \infty$ and $u(p_u(I)-1) \in \text{alph}(r(u))$. It follows from Lemma 7(iv) that $m[\text{suf}(u, p_u(I)-1)].a = 0^{n-I+1}.a$. Since $u(p_u(I)-1) \in \text{alph}(r(u))$ we have

$$\text{alph}(r(u)) = \text{alph}(r(v)) = \{0\},$$

and

$$\text{suf}(v, p_v(I-1)+k) = 0^l 1.\text{ar}_v(I+1) \ldots \text{ar}_v(n-1).0^q 1 0^r,$$

with $q > 0$ and $r > 0$. The string 0^{n-I+1} divides $0^l 1.\text{ar}_v(I+1) \ldots \text{ar}_v(n-1).0^q$ and a divides 10^r.

(ii) We have $r(v) = r(u).a^k$ with $k > 0$. It is readily seen that $m[u].r(u).a$ does not divide u. It divides v since $m[u] = m[v]$. □

Given u and v ($u \neq v$) such that $m[u] = m[v]$ and $\text{alph}(r(u)) = \text{alph}(r(v))$ we now prove that either there exists an arch I such that $\text{disting}(I)$ belongs to $D(u, v)$, or $D(u, v)$ contains a string which distinguishes u and v by their rests.

Proposition 10. *Let u and v be two different strings over* $\{0, 1\}$ *such that* $m[u] = m[v]$ *and* $\mathrm{alph}(r(u)) = \mathrm{alph}(r(v)) = \mathcal{R}$. *Let* $n = |m[u]| = |m[v]|$. *At least one of the following conditions holds.*

(i) $n > 0$ *and there exists* $I \in \{1, \ldots, n\}$ *such that* $|\mathrm{ar}_u(I)| \neq |\mathrm{ar}_v(I)|$, $\mathrm{disting}(I)$ *belongs to* $D(u, v)$ *and* $d(u, v) = |\mathrm{disting}(I)| - 1$.

(ii) $|r(u)| \neq |r(v)|$ *and* $d(u, v) = n + \min(|r(u)|, |r(v)|)$. *If* $|r(u)| < |r(v)|$ *then* $m[u].r(u).a \in D(u, v)$ *otherwise* $m[v].r(v).a \in D(u, v)$, *with* $a \in \mathcal{R}$.

Proof. Let $h \in D(u, v)$. Necessarily $|h| \geqslant 2$. The string $h(1) \ldots h(|h|-1)$ divides u and v. Let s (resp. t) be the *first occurrence* of $h(1) \ldots h(|h|-1)$ in u (resp. v); $h(1) \ldots h(|h|-1) = u(s(1)) \ldots u(s(|h|-1)) = v(t(1)) \ldots v(t(|h|-1))$. For every $I \in \{1, \ldots, n\}$ let $N(u, I, h)$, $N(v, I, h)$, $R(u, h)$ and $R(v, h)$ denote the following sets:

$$N(u, I, h) = \{s(1), \ldots, s(|h|-1)\} \cap \{p_u(I-1)+1, \ldots, p_u(I)\},$$

$$N(v, I, h) = \{t(1), \ldots, t(|h|-1)\} \cap \{p_v(I-1)+1, \ldots, p_v(I)\},$$

$$R(u, h) = \{s(1), \ldots, s(|h|-1)\} \cap \{p_u(n)+1, \ldots, |u|\},$$

$$R(v, h) = \{t(1), \ldots, t(|h|-1)\} \cap \{p_v(n)+1, \ldots, |v|\}.$$

$\mathrm{Card}(N(u, I, h))$ (resp. $\mathrm{Card}(R(u, h))$) indicates how many times $h(1) \ldots h(|h|-1)$ "touches" the Ith arch (resp. the rest) of u.

Case 1. $n = 0$ or $[n > 0$ and $\forall I \in \{1, \ldots, n\}$, $\mathrm{Card}(N(u, I, h)) = \mathrm{Card}(N(v, I, h))]$. We have $\mathrm{card}(R(u, h)) = \mathrm{card}(R(v, h))$. $R(u, h)$ and $R(v, h)$ are not empty, otherwise h would not distinguish u and v. Let $R(u, h) = \{s(r), \ldots, s(|h|-1)\}$ and $R(v, h) = \{t(r), \ldots, t(|h|-1)\}$. The string $h(1) \ldots h(r-1)$ divides $\mathrm{pref}(u, p_u(n))$ and $\mathrm{pref}(v, p_v(n))$. The string $h(1) \ldots h(r)$ divides neither $\mathrm{pref}(u, p_u(n))$ nor $\mathrm{pref}(v, p_v(n))$ because s and t are first occurrences. Suppose h divides v and does not divide u. Then $h(r) \ldots h(|h|)$ divides $r(v)$. If we had $|r(u)| \geqslant |r(v)|$, $h(r) \ldots h(|h|)$ would divide $r(u)$ and h would divide u. Hence $|r(u)| < |r(v)|$. Let $a \in \mathcal{R}$. We have $r(v) = r(u).a^q (q > 0)$ and $h(r) \ldots h(|h|) = r(u).a$. Since $h(1) \ldots h(r)$ does not divide $\mathrm{pref}(u, p_u(n))$ we have $r > |m[\mathrm{pref}(u, p_u(n))]|$ (Proposition 1). Now $m[\mathrm{pref}(u, p_u(n))] = m[u]$, then $r > |m[u]|$ and $|h| \geqslant |m[u].r(u).a|$. It follows from Proposition 9(ii) that $m(u).r(u).a$ distinguishes u and v. Then $m(u).r(u).a \in D(u, v)$ since $h \in D(u, v)$, and condition (ii) holds.

Case 2. $n > 0$ and $\{I \in \{1, \ldots, n\} \mid \mathrm{Card}(N(u, I, h)) \neq \mathrm{Card}(N(v, I, h))\} \neq \emptyset$. Let $J = \min\{I \in \{1, \ldots, n\} \mid \mathrm{Card}(N(u, I, h)) \neq \mathrm{Card}(N(v, I, h))\}$, $N(u, J, h) = \{s(r), \ldots, s(r+p)\}$ and $N(v, J, h) = \{t(r), \ldots, t(r+q)\}$. The string $h(1) \ldots h(r-1)$ divides $\mathrm{pref}(u, p_u(J-1))$ and $\mathrm{pref}(v, p_v(J-1))$; $h(1) \ldots h(r)$ divides neither $\mathrm{pref}(u, p_u(J-1))$ nor $\mathrm{pref}(v, p_v(J-1))$ since s and t are first occurrences. Suppose $p < q$. Then $h(r) \ldots h(r+p+1)$ divides $\mathrm{ar}_v(J)$ but does not divide $\mathrm{ar}_u(J)$ since s is the first occurrence of $h(1) \ldots h(|h|-1)$ in u. Therefore we have $|\mathrm{ar}_u(J)| < |\mathrm{ar}_v(J)|$, $h(r) \ldots h(r+p) = c_u(J)$, $s(r+p) = p_u(J) - 1$ and $t(r+p) \leqslant p_v(J) - 2$.

$$\mathrm{disting}(J) = \mathrm{pref}(m[u], J-1).c_u(J).m[\mathrm{suf}(u, p_u(J)-1)].a$$

with $a \in \{0, 1\} \setminus \text{alph}(u(|u|))$. It follows from Proposition 9 that disting(J) distinguishes u and v. We show that $|h| \geqslant |\text{disting}(J)|$.

(a) The string $h(1) \ldots h(r)$ does not divide $\text{pref}(u, p_u(J-1))$. Now $|m[\text{pref}(u, p_u(J-1))]| = |\text{pref}(m[u], J-1)| = J-1$. Therefore $r-1 \geqslant J-1$ (Proposition 1).

(b) Let us show now that $|h(r+p+1) \ldots h(|h|)| > |m[\text{suf}(u, p_u(J)-1)]|$:
If h does not divide u then $h(r+p+1) \ldots h(|h|)$ does not divide $\text{suf}(u, p_u(J)-1)$, and therefore $|h(r+p+1) \ldots h(|h|)| > |m[\text{suf}(u, p_u(J)-1)]|$ (Proposition 1).
If h divides u then it does not divide v and $h(r+p+1) \ldots h(|h|)$ does not divide $\text{suf}(v, t(r+p))$. Then $|h(r+p+1) \ldots h(|h|)| > |m[\text{suf}(v, t(r+p))]|$ (Proposition 1). We have

$$m[\text{suf}(v, t(r+p))] = m[\text{suf}(v, p_v(J-1))]$$

since

$$t(r+p) \leqslant p_v(J)-2.$$

$$m[\text{suf}(v, p_v(J-1))] = m[\text{suf}(u, p_u(J-1))]$$

since

$$m[u] = m[v].$$

Now

$$|m[\text{suf}(u, p_u(J-1))]| \geqslant |m[\text{suf}(u, p_u(J)-1)]|,$$

thus

$$|h(r+p+1) \ldots h(|h|)| > |m[\text{suf}(u, p_u(J)-1)]|.$$

Finally we have $r-1 \geqslant J-1$, $h(r) \ldots h(r+p) = c_u(J)$ and $|h(r+p+1) \ldots h(|h|)| \geqslant |m[\text{suf}(u, p_u(J)-1)].a|$. Thus $|h| \geqslant \text{disting}(J)|$, $\text{disting}(J) \in D(u, v)$ and condition (i) holds. □

4. Algorithm

In this section u and v are bit-strings. From the above propositions we obtain a linear on-line algorithm which computes $d(u, v)$. It only requires one reading of u and v and a constant amount of extra space. Let

$$n = \min(|m[u]|, |m[v]|), \qquad r = \min(|r(u)|, |r(v)|),$$

$$\text{Diff} = \{I \in \{1, \ldots, n\} \mid |\text{ar}_u(I)| \neq |\text{ar}_v(I)|\},$$

$$M = \{I \in \text{Diff} \mid \forall J \in \text{Diff}, \min(|\text{ar}_u(I)|, |\text{ar}_v(I)|) \leqslant \min(|\text{ar}_u(J)|, |\text{ar}_v(J)|)\},$$

$$\text{lmin} = \min(|\text{ar}_u(I)|, |\text{ar}_v(I)|) \quad \text{with } I \in M,$$

$$P = \{I \in M \mid \text{arch } I \text{ satisfies } \mathscr{P}\}.$$

We can summarize the results of the preceding sections by the following functional statement:

$d(u, v) \equiv$

if $(m[u] \neq m[v]$ **or** $\text{alph}(r(u)) \neq \text{alph}(r(v)))$ **then** n (Propositions 3, 4, 5)

else if $\text{Diff} = \emptyset$ **then** (**if** $|r(u)| = |r(v)|$ **then** ∞ **else** $n + r$) (Proposition 10)

else if $(|r(u)| \neq |r(v)|$ **and** $r \leq \text{lmin} - 2)$ **then** $n + r$ (Propositions 8, 10)

else if $P \neq \emptyset$ **then** $n + \text{lmin} - 2$ (Propositions 8, 10)

else $n + \text{lmin} - 1$. (Propositions 8, 10)

Examples. (i) $u = 110.01.1110.1$, $v = 1110.01.110.11$, $m[u] = m[v] = 010$, $\text{alph}(r(u)) = \text{alph}(r(v)) = \{1\}$, $n = 3$, $r = 1$, $\text{Diff} = \{1, 3\}$, $M = \{1, 3\}$, $\text{lmin} = 3$, $r \leq \text{lmin} - 2$, $d(u, v) = n + r = 4$, $m[u].r(u).1 = 01011 \in D(u, v)$.

(ii) $u = 110.01.1110.11$, $v = 1110.01.110.111$, $m[u] = m[v] = 010$, $\text{alph}(r(u)) = \text{alph}(r(v)) = \{1\}$, $n = 3$, $r = 2$, $\text{Diff} = \{1, 3\}$, $M = \{1, 3\}$, $\text{lmin} = 3$, $r > \text{lmin} - 2$, $P = \{1\}$, $d(u, v) = n + \text{lmin} - 2 = 4$, $\text{disting}(1) = 11100 \in D(u, v)$.

(iii) $u = 10.01.001.10.00$, $v = 10.001.0001.10.0$, $m[u] = m[v] = 0110$, $\text{alph}(r(u)) = \text{alph}(r(v)) = \{0\}$, $n = 4$, $r = 1$, $\text{Diff} = \{2, 3\}$, $M = \{2\}$, $\text{lmin} = 2$, $r > \text{lmin} - 2$, $P = \emptyset$, $d(u, v) = n + \text{lmin} - 1 = 5$, $\text{disting}(2) = 000101 \in D(u, v)$.

In order to compute $d(u, v)$ we must merely calculate n, r, lmin, and determine whether $m[u] = m[v]$, $\text{alph}(r(u)) = \text{alph}(r(v))$, $|r(u)| = |r(v)|$, $\text{Diff} \neq \emptyset$ and $P \neq \emptyset$. This is done by the following algorithm.

The algorithm uses the procedure NEXTARCH. When NEXTARCH(u) is called it reads the next arch of u. If $a^k b$ $(k > 0)$ is read, the variables bit-arch[u], length-arch[u] and bool-arch[u] are respectively bound to a, $k + 1$ and true. If the rest a^k $(k \geq 0)$ is read the variables bit-arch[u], length-arch[u] and bool-arch[u] are respectively bound to a, k and false (if $k = 0$ the value of bit-arch[u] is indeterminate). Notice that procedure NEXTARCH can be implemented in such a way that it only needs a constant memory. The algorithm runs through u and v using NEXTARCH.

The variable lmin is initially bound to ∞. If immediately after arch I has been read the conditions min(length-arch[u],length-arch[v]) < lmin and length-arch[u] $\neq$ length-arch[v] hold, the value of lmin is changed by the assignment lmin := min(length-arch[u],length-arch[v]). In order to determine whether arch I satisfies $\mathscr{P}$ we use the variables inf, bit and p which are updated by calling procedure UPDATE. Each time UPDATE is called the following assignments are performed: if length-arch[u] < length-arch[v] then inf := u else inf := v, bit := bit-arch[inf] and p := ind (indeterminate). The value of p remains "ind" until it can be decided whether arch I satisfies $\mathscr{P}$, "true" (resp. "false") is then assigned to p if arch I does satisfy $\mathscr{P}$ (resp. does not satisfy $\mathscr{P}$). This is done by procedure COMPUTE_P which compares for each arch $J > I$ the current value of bit-arch[inf] with that of bit, and tests whether length-arch[inf] > 2.

The procedure UPDATE is also called when the value of p is false and conditions min(length-arch[u],length-arch[v]) = lmin and length-arch[u] $\neq$ length-arch[v] hold, because even if lmin is not changed, the values of inf and bit might be different and p might become true.

When the algorithm terminates we obtain the following bindings:

- n is bound to $\min(|m[u]|,|m[v]|)$,
- bool-m is bound to the boolean value of $m[u]=m[v]$.

If $m[u]=m[v]$ then:

- r is bound to $\min(|r(u)|,|r(v)|)$,
- bool-rl is bound to the value of $|r(u)|=|r(v)|$,
- bool-r is bound to the value of $\mathrm{alph}(r(u))=\mathrm{alph}(r(v))$,
- bool-Diff is bound to the value of Diff $\neq \emptyset$,

If $m[u]=m[v]$ and Diff $\neq \emptyset$ then:

- lmin is bound to $\min(|\mathrm{ar}_u(I)|,|\mathrm{ar}_v(I)|)$ with $I \in M$,
- p is bound to the value of $P \neq \emptyset$.

```
procedure UPDATE;
begin
  if length-arch[u] < length-arch[v] then inf := u else inf := v;
  bit := bit-arch[inf]; p := ind
end; {UPDATE}

procedure COMPUTE_P;
begin
  if bit-arch[inf] ≠ bit then p := true
  else if length-arch[inf] > 2 then p := false;
end; {COMPUTE_P}

Algorithm
begin
n := 0; lmin := ∞; p := false; NEXTARCH(u); NEXTARCH(v);

while bool-arch[u] and bool-arch[v] and bit-arch[u] = bit-arch[v] do begin
    n := n + 1;
    if p = ind then COMPUTE_P;
(1) if length-arch[u] ≠ length-arch[v] then
      if min(length-arch[u], length-arch[v]) < lmin then
        begin lmin := min(length-arch[u], length-arch[v]);
          UPDATE
        end
      else if min(length-arch[u], length-arch[v]) = lmin and p = false
        then UPDATE;
    NEXTARCH(u);NEXTARCH(v)
end; {while}
```

```
if not bool-arch[u] and not bool-arch[v] then {here m[u] = m[v]}
  begin
    bool-m := true;
    r := min(length-arch[u], length-arch[v]);
    if length-arch[u] = length-arch[v]
      then bool-rl := true else bool-rl := false;
    if (r = 0 and bool-rl) or (r ≠ 0 and bit-arch[u] = bit-arch[v])
      then bool-r := true else bool-r := false;
    if lmin ≠ ∞ then bool-Diff := true else bool-Diff := false;
    if p = ind then
      if length-arch[inf] = 0 or bit ≠ bit-arch[inf] then p := true else p := false
  end

else
  begin {here m[u] ≠ m[v]}
    bool-m := false;
    while bool-arch[u] and bool-arch[v] do
      begin n := n + 1; NEXTARCH(u); NEXTARCH(v) end
  end
end.{algorithm}
```

Remark. Line (1): if the value of p is still "ind" necessarily bit-arch[inf] = bit and length-arch[inf] = 2 (otherwise COMPUTE_P would have assigned "true" or "false" to p). Therefore if length-arch[u] ≠ length-arch[v], min(length-arch[u],length-arch[v]) = lmin and p = ind it is useless to call UPDATE (inf, bit and p would not be changed).

It follows from Propositions 3, 4, 5, 10 and Lemma 7 that the above algorithm can be straightforwardly adapted to compute a string of $D(u, v)$. The main difference is that the models $m[u]$ and $m[v]$ must be explicitly computed and the required amount of extra space becomes $O(d(u, v))$.

5. Conclusion

The arch factorization provides an efficient method to compute $d(u, v)$ when u and v are bit-strings. One may ask whether this method could be generalized to any pair of strings. We first notice that Propositions 3 and 4 hold even if u and v are not bit-strings, thus $d(u, v)$ can always be computed if $|m[u]| \neq |m[v]|$ or $\text{alph}(r(u)) \neq \text{alph}(r(v))$. Difficulties arise when $|m[u]| = |m[v]|$ and $\text{alph}(r(u)) = \text{alph}(r(v))$. For example let us merely consider the case $|m[u]| = |m[v]| = n$, $\text{alph}(r(u)) = \text{alph}(r(v))$ and $m[u] \neq m[v]$. If u and v are bit-strings we can prove that $d(u, v) = n$ (Proposition 5), but this result does not hold if $\text{card}(A) > 2$ (see the remark following Proposition 5). In fact the proof of Proposition 5 strongly depends on the cardinality of A. In order to compute $d(u, v)$ when $\text{card}(A) > 2$, the analysis of u and v must be less coarse than the one provided by the mere application of arch factorization. The way for future work could be to compute the arch factorization of u and v, and then of every arch of u and v, and so on.

References

[1] J.J. Hebrard and M. Crochemore, Calcul de la distance par les sous-mots, *RAIRO Inform Théor. Appl.* **20**(4) (1986) 441-456.
[2] S.K. Kumar and C.P. Rangan, A linear space algorithm for the LCS problem, *Acta Inform.* **24** (1987) 353-362.
[3] Lothaire, *Combinatorics on Words* (Cambridge University Press, 1983).
[4] W.J. Masek and M.S. Paterson, A faster algorithm computing string edit distances, *J. Comput. System Sci.* **20** (1980) 18-31.
[5] N. Nakatsu, Y. Kambayashi and S. Yajima, A longest common subsequence algorithm suitable for similar test strings, *Acta Inform.* **18** (1982) 171-179.
[6] D. Sankoff and J.B. Kruskal, *Time Warps, String Edits and Macromolecules: the Theory and Practice of Sequence Comparison* (Addison-Wesley, Reading, MA, 1983).
[7] I. Simon, An algorithm to distinguish words efficiently by their subwords, 1984, unpublished.