INFORMATIQUE THÉORIQUE ET APPLICATIONS

A. J. KFOURY

A linear-time algorithm to decide whether a binary word contains an overlap

Informatique théorique et applications, tome 22, n° 2 (1988), p. 135-145

<http://www.numdam.org/item?id=ITA_1988__22_2_135_0>

Informatique théorique et Applications/Theoretical Informatics and Applications
(vol. 22, n° 2, 1988, p. 135 à 145)

A LINEAR-TIME ALGORITHM TO DECIDE WHETHER A BINARY WORD CONTAINS AN OVERLAP (*)

by A. J. KFOURY (1)

Communicated by J. E. PIN

Abstract. – *We present two new results. First, we give a new linear-time algorithm to test whether an arbitrary binary word contains an overlap. Second, we show that the number of overlap-free binary words of length n does not exceed* $\mathcal{O}(n^e)$ *for some* $e<2$.

Résumé. – *Nous établissons deux nouveaux résultats. Premièrement, nous présentons un nouvel algorithme qui décide en temps linéaire si un mot binaire arbitraire contient un chevauchement. Deuxièmement, nous démontrons que le nombre de mots binaires de longueur n ne contenant pas de chevauchements ne dépasse pas* $\mathcal{O}(n^e)$ *où* $e<2$.

1. INTRODUCTION

Crochemore was the first to develop a $\mathcal{O}(n)$ algorithm to test whether a word of length n is square-free [2]; his algorithm can be adapted to decide in linear time whether a binary word is overlap-free. Here we take a fresh look at the latter problem and develop another linear-time algorithm for it (with a relatively small proportionality constant). Furthermore, an analysis of our algorithm allows us to determine a subquadratic bound on the number of overlap-free words over a binary alphabet which is tighter than the previously known bound; ours is $\mathcal{O}(n^e)$ for some $e<2$, whereas Restivo's and Salemi's is $\mathcal{O}(n^{\log 1.5})$ [5].

Concerning notation and terminology, we will generally follow Chapter 1 of [6]. In addition, late Greek letters (π, ρ, σ and τ) will denote variables ranging over the set of possible words, and late Roman letters (x, y and z) will stand for individual symbols from finite alphabets.

(*) Received July 1986, revised June 1987.
(1) Laboratoire de Génie Informatique, Institut I.M.A.G., Grenoble, France.
Permanent address: Computer Science Department, Boston University, Boston, Mass, U.S.A., 02215.

Informatique théorique et Applications/Theoretical Informatics and Applications
0296-1598 88/02 135 11/$3.10/

Formally, a word σ contains a *square* if it contains a finite subword of the form $\tau\tau$, where τ is non-empty. A word σ contains an *overlap* if it contains a finite subword of the form $\rho\tau\rho'$ such that $\rho\tau=\tau\rho'$, where ρ, τ, and ρ' are non-empty. Square-freeness implies overlap-freeness, but not the other way around. No word of length $\geqq 4$ over a binary alphabet can be square-free; on the other hand, there are overlap-free words of unbounded length over a binary alphabet [6].

If σ is a word over a binary alphabet, $\bar{\sigma}$ will denote the complement of σ, i. e., the word obtained from σ by replacing every 0 by 1 and every 1 by 0.

The following is a useful characterization of overlap-freeness we shall use repeatedly in the sequel.

1.1. LEMMA: *A word σ is overlap-free $\Leftrightarrow$ σ does not contain a finite subword of the form $x\tau x\tau x \Leftrightarrow \sigma$ does not contain a finite subword of the form $\tau\tau x$ (or $x\tau\tau$) where τ is non-empty and x is the first (or last) symbol of τ.*

Proof: Straightforward. ■

2. SOME PROPERTIES OF OVERLAP-FREE WORDS

Before giving our algorithm in Section 3, we state and prove a few technical results about overlap-free words, from which will follow the correctness of the algorithm. Many of these are implicit in [5].

2.1. LEMMA: *Let $\sigma=x_1 x_2 \ldots x_n \in \{0,1\}^*$ be an overlap-free word of length $n\geqq 5$.*

(1) *If $x_1 x_2 x_3 x_4 \in \{0110, 1001, 1010, 0101\}$ or if $x_1 x_2 x_3 x_4 x_5 \in \{00100, 11011\}$, then for all even j, $4\leqq j<n$, $x_j=\bar{x}_{j-1}$.*

(2) *If $x_1 x_2 x_3 x_4 \in \{0011, 1100, 0100, 1011\}$ or if $x_1 x_2 x_3 x_4 x_5 \in \{00101, 11010\}$, then for all odd j, $3\leqq j<n$, $x_j=\bar{x}_{j-1}$.*

It is easy to check that all prefixes $x_1 x_2 x_3 x_4$ not mentionned in (1) *and* (2) *contain an overlap.*

Proof: (1) By induction on even $j\geqq 4$. The result is clear for $j=4$. Let then k be an even integer, $4<k<n$, and assume the result is already proved for every even j, $4\leqq j<k$. With no loss of generality, suppose $x_{k-3}x_{k-2}=01$. This forces $x_{k-1}x_k\neq 11$, otherwise $x_{k-2}x_{k-1}x_k$ would be an overlap. We next show that $x_{k-1}x_k\neq 00$.

By the introduction hypothesis, $x_{k-5}x_{k-4}=01$ or 10, (In case $k=6$, $x_{k-5}x_{k-4}$ may also be 11, but then in this case $x_{k-1}x_k=10$). If $x_{k-5}x_{k-4}=01$

and $x_{k-1}x_k=00$ then $x_{k-5}\ldots x_{k-1}$ would be an overlap. If $x_{k-5}x_{k-4}=10$ and $x_{k-1}x_k=00$ (the latter condition forcing $x_{k+1}=1$), then $x_{k-5}\ldots x_{k+1}$ would be an overlap.

(2) By induction on odd $j\geqq 3$. The proof is similar to (1) above, and we omit it. ■

2.2. LEMMA: *If* $\sigma\in\{0,1\}^*$ *is overlap-free of length* $\geqq 7$, *there is a unique way of decomposing* σ *into three parts:* $\sigma=\pi\rho\pi'$ *where* π, $\pi'\in\{\lambda, 0, 1, 00, 11\}$ *and* $\rho\in\{01, 10\}^+$.

Proof: This easily follows from the preceding lemma. ■

The π and π' in 2.2 are determined according to the 12 patterns mentioned in Lemma 2.1. Clearly, once π is determined, so is π'. If σ has as a prefix one of the first four patterns in 2.1.1:

$$0110, 1001, 1010, 0101.$$

then $\pi=\lambda$; if σ has as a prefix one of the remaining two patterns in 2.1.1:

$$00100, 11011,$$

then $\pi=00$ or 11; and if σ has as a prefix one of the six patterns in 2.1.2:

$$0011, 1100, 0100, 1011, 00101, 11010.$$

then $\pi=0$ or 1.

For the case of overlap-free words σ of length $\leqq 6$ not in $\{\lambda, 0, 1, 00, 11\}$, a decomposition of σ in the form prescribed by Lemma 2.2 is always possible, although it may not be unique (e.g., let $\sigma=001011$ which admits two such decompositions, according to whether $\pi=0$ or $\pi=00$.) Also, a unique decomposition in the form prescribed by Lemma 2.2 is sometimes possible for words σ that are not overlap-free (for example, let $\sigma=001100110$).

2.3. LEMMA: *Let* π, $\pi'\in\{\lambda, 0, 1, 00, 11\}$, $\rho\in\{01, 10\}^+$ *and* $|\rho|>4$. *Then* $\pi\rho\pi'$ *is overlap-free* $\Leftrightarrow$ *both* $\pi\rho$ *and* $\rho\pi'$ *are overlap free.*

Proof: The left-to-right implication is immediate. For the converse, assume both $\pi\rho$ and $\rho\pi'$ are overlap-free but that $\pi\rho\pi'$ is not, and we shall get a contradiction. Under this assumption, $|\pi|\neq 0$ and $|\pi'|\neq 0$. Because ρ is of even length, not both $|\pi|=1$ and $|\pi'|=1$, otherwise $\pi\rho\pi'$ would not be of the form $\tau\tau x$ where x is the leftmost symbol of τ. With no loss of generally, assume $|\pi|=2$. This implies that $\pi=xx$ and $\pi'=x$ (or xx), where $x\in\{0,1\}$, with the shortest overlap in $\pi\rho\pi'$ being $xx\rho x$ (or $x\rho xx$). But if $\rho\in\{01, 10\}^+$,

it is now easily checked that both $\pi\rho$ and $\rho\pi'$ contain an overlap, contradicting the initial assumption. ■

In the preceding lemma we cannot ignore the condition $|\rho|>4$ in the hypothesis. For example, if $\pi=00$, $\rho=1001$, and $\pi'=00$, both $\pi\rho$ and $\rho\pi'$ are overlap-free but $\pi\rho\pi'$ is not.

2.4. LEMMA: *Let* $x, y\in\{0,1\}$ *and* $\rho\in\{01,10\}^+$. *Then:*

(1) $x\rho$ *overlap-free* $\Leftrightarrow \bar{x}x\rho$ *overlap-free,*

(2) ρy *overlap-free* $\Leftrightarrow \rho y\bar{y}$ *overlap-free.*

Proof: The right-to-left implications are trivially true. We prove the left-to-right implication in (1) only; the same proof applies to (2). Assume then that $x\rho$ is overlap-free, $\bar{x}x\rho$ is not, and we get a contradiction. Under this assumption, the shortest overlap in $\bar{x}x\rho$ contains the leftmost $\bar{x}$; i.e., $\bar{x}x\rho$ contains a prefix $\tau\tau\bar{x}$ with $\tau\neq\lambda$. Given that $\rho\in\{01,10\}^+$, it is easily seen τ cannot be of odd length. If τ is of even length, $x\rho$ has already a prefix $\tau'\tau' x$, with $|\tau'|=|\tau|$, contradicting the assumption that $x\rho$ is overlap-free. ■

2.5. LEMMA: *Let* $x, y\in\{0,1\}$ *and* $\rho\in\{01,10\}^+$. *Then:*

(1) $xx\rho$ *overlap-free* $\Rightarrow \bar{x}x\rho$ *overlap-free,*

(2) ρyy *overlap-free* $\Rightarrow \rho y\bar{y}$ *overlap-free,*

(3) $\bar{x}x\rho$ *overlap-free* $\Rightarrow$ *if* $xx\rho$ *has an overlap then* xxx *or* $xx\bar{x}xx\bar{x}x$ *is a prefix of* $xx\rho$,

(4) $\rho y\bar{y}$ *overlap-free* $\Rightarrow$ *if* ρyy *has an overlap then* yyy *or* $y\bar{y}yy\bar{y}yy$ *is a suffix of* ρyy.

Proof: The proofs for (1) and (2) are similar to those of the left-to-right implications in the preceding lemma. We prove (3) only; the proof for (4) is similar.

For (3), assume $\bar{x}x\rho$ is overlap-free and $xx\rho$ is not. Hence $xx\rho$ contains a prefix $\tau\tau x$, with $\tau\neq\lambda$. Given that $\rho\in\{01,10\}^+$ and that τ is x or starts with xx, the length of τ must be odd. Given that $x\rho$ is overlap-free, this forces the prefix $\tau\tau x$ to be xxx or $xx\bar{x}xx\bar{x}x$. ■

We define a partial function φ from $\{0,1\}^+$ to $\{01,10\}^+$. φ is defined for all words of the form $\pi\rho\pi'$ where $\pi, \pi'\in\{\lambda,0,1,00,11\}$ and $\rho\in\{01,10\}^+$ by:

$$\varphi(\pi\rho\pi')=\begin{cases}\rho, & \text{if } \pi=\pi'=\lambda;\\ \bar{x}x\rho, & \text{if } \pi=x \text{ or } xx, \text{ with } x\in\{0,1\}, \text{ and } \pi'=\lambda;\\ \rho y\bar{y}, & \text{if } \pi'=y \text{ or } yy, \text{ with } y\in\{0,1\}, \text{ and } \pi=\lambda;\\ \bar{x}x\rho y\bar{y}, & \text{if } \pi=x \text{ or } xx, \pi'=y \text{ or } yy, \text{ with } x,y\in\{0,1\}.\end{cases}$$

2.6. LEMMA: *Let* $\pi, \pi' \in \{\lambda, 0, 1, 00, 11\}$, *and* $\rho \in \{01, 10\}^+$. *Hypothesis*:

(1) *If* ($\pi = x$ or xx) *and* ($\pi' = y$ or yy), *then* $x\rho y \neq \tau\tau$ *for all* $\tau \in \{0, 1\}^+$;

(2) *If* $\pi = xx$ *then neither* xxx *nor* $xx\bar{x}xx\bar{x}x$ *is a prefix of* $\pi\rho$;

(3) *If* $\pi' = yy$ *then neither* yyy *nor* $y\bar{y}yy\bar{y}yy$ *is a prefix of* $\rho\pi'$.

Conclusion: $\pi\rho\pi'$ *overlap-free* $\Leftrightarrow$ $\varphi(\pi\rho\pi')$ *overlap-free.*

Proof: We consider the case when $|\rho| > 4$, so that Lemma 2.3 can be used. For the case when $|\rho| \leqq 4$, the lemma is established exhaustively.

By Lemma 2.3, 2.4, and 2.5, it is easy to see that all we need to prove is the implication $\bar{x}x\rho$ and $\rho y\bar{y}$ overlap-free $\Rightarrow$ $\bar{x}x\rho y\bar{y}$ overlap-free. Assume that $\bar{x}x\rho$ and $\rho y\bar{y}$ are overlap-free, but that $\bar{x}x\rho y\bar{y}$ is not. A shortest overlap (an expression of the form $\tau\tau z$ with z the first symbol of τ) in $\bar{x}x\rho y\bar{y}$ is therefore $x\rho y\bar{y}$, or $\bar{x}x\rho y$, or $\bar{x}x\rho y\bar{y}$. Because $\bar{x}x\rho y\bar{y}$ is of even length, it cannot be a shortest overlap. Hence, $x\rho y\bar{y}$ or $\bar{x}x\rho y$ is a shortest overlap. But in both cases, this contradicts the fact that $x\rho y \neq \tau\tau$ for all τ. ■

We cannot omit condition (1) in the hypothesis of the preceding lemma. For example, let $\pi = 0$, $\pi' = 1$, and $\rho = 011001$, so that $\pi\rho\pi' = \tau\tau$ with $\tau = 0011$. Here $\pi\rho\pi'$ is overlap-free, but $\varphi(\pi\rho\pi')$ is not.

We define another partial function ψ from $\{0, 1\}^+$ to $\{01, 10\}^+$. ψ is defined for all words of the form $\pi\rho\pi'$ where $\pi, \pi' \in \{\lambda, 0, 1, 00, 11\}$ and $\rho \in \{01, 10\}^+$, by:

$$\psi(\pi\rho\pi') = \begin{cases} \rho, & \text{if } \pi = \pi' = \lambda; \\ \bar{x}x\rho, & \text{if } \pi = x \text{ or } xx, \text{ with } x \in \{0, 1\}, \text{ and } \pi' = \lambda; \\ \rho y\bar{y}, & \text{if } \pi = \lambda \text{ or } x \text{ or } xx, \pi' = y \text{ or } yy, \text{ with } x, y \in \{0, 1\}. \end{cases}$$

Observe that $\varphi(\pi\rho\pi')$ and $\psi(\pi\rho\pi')$ are words in $\{01, 10\}^+$, and $\varphi(\pi\rho\pi') = \psi(\pi\rho\pi')$ whenever $\pi = \lambda$ or $\pi' = \lambda$.

2.7. LEMMA: *Let* $\pi, \pi' \in \{\lambda, 0, 1, 00, 11\}$, *and* $\rho \in \{01, 10\}^+$. *Hypothesis*:

(1) *If* ($\pi = x$ *or* xx) *and* ($\pi' = y$ *or* yy), *then* $x\rho y = \tau\tau$ *for some* $\tau \in \{0, 1\}^+$;

(2) *If* $\pi = xx$ *then neither* xxx *nor* $xx\bar{x}xx\bar{x}x$ *is a prefix of* $\pi\rho$;

(3) *If* $\pi' = yy$ *then neither* yyy *nor* $y\bar{y}yy\bar{y}yy$ *is a suffix of* $\rho\pi'$.

Conclusion: $\pi\rho\pi'$ *overlap-free* $\Leftrightarrow$ $\psi(\pi\rho\pi')$ *overlap-free.*

Proof: We consider the case when $|\rho| > 4$, so that Lemma 2.3 can be used. For the case $|\rho| \leqq 4$, the lemma is established exhaustively.

For the right-to-left implication, there are several subcases to consider, some trivial. We consider only one of the non-trivial subcases here (the others being treated similarly). Suppose that $\rho y\bar{y}$ is overlap-free, and we want to

show that $xx\rho y$ is also overlap-free. This is established by the following sequence of implications:

$\rho y\bar{y}$ overlap-free $\Rightarrow$ ρy overlap-free

$\Rightarrow$ $x\rho y$ overlap-free (by (1)) $\Rightarrow$ $x\rho$ overlap-free

$\Rightarrow$ $x\rho y\bar{y}$ overlap-free (by 2.3) $\Rightarrow$ $\bar{x}x\rho y\bar{y}$ overlap-free (by 2.4)

$\Rightarrow$ $xx\rho y\bar{y}$ overlap-free (by 2.5) $\Rightarrow$ $xx\rho y$ overlap-free.

For the left-to-right implication, it readily follows by Lemmas 2.3, 2.4, and 2.5, that we only need to prove that: $x\rho y$ contains an overlap $\Rightarrow$ both $x\rho$ and ρy contain an overlap.

Assume $x\rho y$ contains an overlap, but not ρy, and we get a contradiction. (The proof that $x\rho$ contains an overlap is similar.) By hypothesis, $x\rho y = \tau\tau$ so that $x\rho y = x\tau' yx\tau' y$ for some $\tau' \in \{0,1\}^+$. Because ρy is overlap-free, $x\rho y$ has a prefix $x\tau''\tau''$ for some $\tau'' \in \{0,1\}^+$ which ends with symbol x. Hence $\tau''\ \tau''$ is a non-empty prefix of $\tau' yx\tau' y = \rho y$. If $\tau''\tau''$ is also a prefix of $\tau' y$ then ρy contains the sub-expression $x\tau''\tau''$, which is an overlap, contradicting the initial assumption. If $\tau' yx$ is a prefix of $\tau''\tau''$, itself a prefix of $\tau' yx\tau' y$, it is not difficult to see that $\tau' xy\tau' y$ contains an overlap-another contradiction. ■

We need one more fact about overlap-free words. This is a classical result due to Thue [6].

2.8. LEMMA: *Let* $\sigma \in \{01, 10\}^+$ *and* σ' *be obtained from* σ *by mapping consecutive occurrences of* 01 *and* 10 *into* 0 *and* 1, *respectively. Then* σ *is overlap-free* $\Leftrightarrow$ σ' *is overlap-free.*

Proof: Straightforward. ■

3. THE ALGORITHM

The input to the algorithm is an arbitrary $\sigma \in \{0,1\}^+$. At the first iteration, the algorithm sets $\sigma_1 = \sigma$. At the n-th iteration, $n \geqq 1$, it carries out the following steps:

1. If $\sigma_n \in \{0, 1, 00, 11\}$, terminate successfully.

2. Decompose σ_n as $\pi_n \rho_n \pi_n'$ with $\pi_n, \pi_n' \in \{\lambda, 0, 1, 00, 11\}$ and $\rho_n \in \{01, 10\}^+$. If this is not possible, terminate unsucessfully. If σ_n has more than one such decomposition, take π_n as short as possible.

3. If $\pi_n = xx$, with $x \in \{0,1\}$, and xxx or $xx\bar{x}xx\bar{x}x$ is a prefix of $\pi_n \rho_n$, terminate unsuccessfully.

4. If $\pi'_n = yy$, with $y \in \{0,1\}$, and yyy or $y\bar{y}yy\bar{y}yy$ is a suffix of $\rho_n \pi'_n$, terminate unsuccessfully.

5. If ($\pi_n = x$ or xx) and ($\pi'_n = y$ or yy) and ($x\rho_n y = \tau\tau$ for some $\tau \in \{0,1\}^*$) then goto 6 else goto 7.

6. Define σ_{n+1} from $\psi(\pi_n \rho_n \pi'_n)$ by mapping consecutive occurrences of 01 and 10 into 0 and 1, respectively, and go to the $(n+1)$-st iteration.

7. Define σ_{n+1} from $\varphi(\pi_n \rho_n \pi'_n)$ by mapping consecutive occurrences of 01 and 10 into 0 and 1, respectively, and go to the $(n+1)$-st iteration.

For later reference, we call the above algorithm $\mathscr{A}$. Without giving the details, let us note that $\mathscr{A}$ can be easily modified into another algorithm $\mathscr{A}'$ which executes the same steps as $\mathscr{A}$, except that if $\mathscr{A}$ terminates unsuccessfully then $\mathscr{A}'$ in addition returns a subword of the input word which is an overlap. More specifically, if $\mathscr{A}$ terminates unsuccessfully at Step 2 (resp. Step 3, respec. Step 4) at the n-th iteration, then σ_n contains an overlap of the form xxx or $x\bar{x}x\bar{x}x$ or $xx\bar{x}xx\bar{x}x$ (respec. xxx or $xx\bar{x}xx\bar{x}x$ as a prefix, respec. xxx or $x\bar{x}xx\bar{x}xx$ as a suffix) with $x \in \{0,1\}$. This last assertion is immediately verified for Step 3 and Step 4; to verify it for Step 2 also, one may use the reasoning of the proof of 2.1 (left to the reader). An overlap in σ_n must finally be translated into an overlap in σ_1, the original input word, and it is not difficult to see that this can be done without exceeding a linear time complexity (established for $\mathscr{A}$ in the next theorem).

3.1. THEOREM: *Algorithm $\mathscr{A}$ terminates successfully on input $\sigma \in \{0,1\}^+ \Leftrightarrow \sigma$ is overlap-free. If $|\sigma| = n$, then $\mathscr{A}$ executes at most $\mathcal{O}(n)$ steps.*

Proof: The correctness of Step 2 in $\mathscr{A}$ follows from Lemma 2.2. Steps 3 and 4 in $\mathscr{A}$ test whether conditions (2) and (3) of lemmas 2.6 and 2.7 are satisfied. The correctness of Step 5 in $\mathscr{A}$ follows from Lemmas 2.6 and 2.7. The correctness of Steps 6 and 7 follows from Lemma 2.8.

For the time complexity of $\mathscr{A}$, observe that on the first iteration $\mathscr{A}$ processes a word of length n, on the second iteration it processes a word of length $n/2$, on the third iteration it processes a word of legnth $n/4$, etc., from which we deduce that $\mathscr{A}$ runs in a linear number of steps. ■

We can get an estimate for the proportionality constant in the time complexity of algorithm $\mathscr{A}$ as follows. First of all, note that on each iteration $\mathscr{A}$ makes at most 7 passes of the word it processes (at most 5 passes in Step 2, 1 pass in Step 5, and 1 pass in Step 6 or 7). Since $n + (n/2) + (n/4) + \ldots \leqq 2n$,

the time complexity of $\mathscr{A}$ is therefore $\leqq 14n+c$, where c is the cost incurred in Steps 3 and 4 which is $\leqq 14\log(n)$ (7 comparisons for each of Step 3 and Step 4, multiplied by the maximum number of iterations).

3.2. THEOREM: *There are at most $\mathcal{O}(n^e)$ overlap-free words of length n, for some $e<2$.*

Proof: The analysis is simpler if we modify Algorithm $\mathscr{A}$. Let $U \subset \{0,1\}^{\leqq 6}$ be the set of overlap-free words of length $\leqq 6$. We replace Step 1 in $\mathscr{A}$ by the following:

1. If $\sigma_n \in U$, terminate successfully; and if $\sigma_n \in \{0,1\}^{\leqq 6} - U$, terminate unsuccessfully.

We call $\mathscr{B}$ the algorithm obtained from $\mathscr{A}$ after this modification. Clearly, $\mathscr{B}$ terminates successfully if and only if the input is overlap-free.

The bound mentioned in the statement of the theorem is a bound on the number of words of length $n \geqq 7$ on which $\mathscr{B}$ terminates successfully. If $\mathscr{B}$ terminates successfully on $\sigma \in \{0,1\}^+$ and $|\sigma| = n \geqq 7$, then $\mathscr{B}$ executes k iterations for some $2 \leqq k \leqq \lceil \log(n-2) \rceil - 1$. Indeed, it is not difficult to check that $|\sigma_i| \leqq (n+2^i-2)/2^{i-1}$ for $i \geqq 1$, so that the largest possible value of i such that $|\sigma_i| \leqq 6$ is $\lceil \log(n-2) \rceil - 1$.

Assume $\mathscr{B}$ executes $k \geqq 2$ iterations and terminates successfully. Hence, the first test of Step 2, and the tests of Steps 3 and 4, are always false in the course of this execution. The only test which may switch from false to true, or vice-versa, is that of Step 5. (The test of Step 1 is false throughout except in the k-th iteration.)

Case 1: The test of Step 5 remains false throughout the execution of $\mathscr{B}$ (the first $k-1$ iterations of $\mathscr{B}$). In this case, the input σ is fully determined by the values of $\pi_1, \pi'_1, \pi_2, \pi'_2, \ldots, \pi_{k-1}, \pi'_{k-1}$, and σ_k; i. e., for a given value of $\sigma_k \in U$ by running the algorithm in "reverse" we uniquely reconstruct the input σ, if we also know the values of $\pi_1, \pi'_1, \ldots, \pi_{k-1}, \pi'_{k-1}$. Although $\pi_i, \pi'_i \in \{\lambda, 0, 1, 00, 11\}$, and therefore the pair (π_i, π'_i) may assume one of 25 values, it is easy to see that if we know the leftmost and rightmost symbols of σ_i—call them x and y respectively—then (π_i, π'_i) may assume no more than 5 distinct values for each $i = 1, 2, \ldots, k-1$. Indeed, if $|\sigma_i|$ is even.

$$(\pi_i, \pi'_i) \in \{(\lambda, \lambda), (x, y), (xx, \lambda), (\lambda, yy), (xx, yy)\},$$

and if $|\sigma_i|$ is odd,

$$(\pi_i, \pi'_i) \in \{(\lambda, y), (x, \lambda), (x, yy), (xx, y)\}.$$

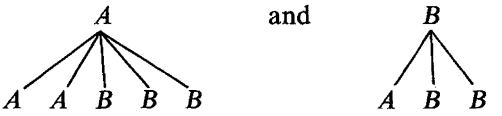
Furthermore, given values for x, y, and (π_i, π_i'), it is also easy to see that the leftmost and rightmost symbols of σ_{i+1} are uniquely determined [and therefore (π_{i+1}, π_{i+1}') may in turn assume no more than 5 values]. From this we conclude that the number of input words $\sigma = \sigma_1$ for which $\mathscr{B}$ will terminate successfully in k iterations, under the assumption that the test of Step 5 remains false throughout, does not exceed

$$4 \cdot 12 \cdot 5^{k-1}$$

where the "4" is the number of possibilities for the pair (lefmost symbol, rightmost symbol) of σ_1, and the "12" the number of possible values for σ_k (an overlap-free word of length 4, 5, or 6, given that we know its leftmost and rightmost symbols). The above bound can be made tighter when we observe that if $\pi_1 = xx$, and therefore if $xx\bar{x}xx\bar{x}\bar{x}$ is a prefix of σ_i then $\pi_{i+1} \neq \lambda$; and likewise, if $\pi_i' = yy$, $\pi_{i+1}' \neq \lambda$. Hence, if we classify the possible values of the pairs (π, π') into two groups, those that do not have xx ou yy as a component and those that do:

$$A = \{(\lambda, \lambda), (x, y), (\lambda, y)(x, \lambda)\},$$
$$B = \{(xx, \lambda), (\lambda, yy), (xx, yy), (x, yy), (xx, y)\},$$

then an instance of A for (π_i, π_i') may be followed by the following instances for (π_{i+1}, π_{i+1}'): 2 instances of A and 3 instances of B (in case $|\sigma_{i+1}|$ is even), or 2 instances of A and 2 instances of B (in case $|\sigma_{i+1}|$ is odd) — in either case, at most 2 A's and 3 B's. On the other hand, an instance of B for (π_i, π_i') may be followed for (π_{i+1}, π_{i+1}') by: at most 1 instance of A and at most 2 instances of B (whether $|\sigma_{i+1}|$ is even or odd). Hence, if $F(h)$ is the number of paths in a tree of height $h \geqq 1$, whose nodes other than the root are labelled with A and B satisfying the conditions:



and the root has 5 successors labelled A, A, B, B, B, then our earlier bound may be reduced to

$$4 \cdot 12 \cdot F(k-1) = 48 \cdot F(k-1).$$

Case 2: The test of Step 5 is true in the i-th iteration, for some $i \leqq k-1$. In this case it is not too difficult to show that in the j-th iteration, for $j = i+1, \ldots, k-1$: the test of Step 5 is false iff $\pi_j = \pi_j' = \lambda$, and the test of

Step 5 is true iff $\pi_j = x$ and $\pi'_j = \bar{x}$ for some $x \in \{0, 1\}$. (This last assertion holds provided $|\sigma_j| \geqq 7$, which is the reason for modifying $\mathscr{A}$ into $\mathscr{B}$.) Hence, in this case, the input σ is fully determined by the values of π_1, $\pi'_1, \ldots, \pi_{k-1}, \pi'_{k-1}$, and σ_k, and the iteration number $i \leqq k-1$ for which the test of Step 5 is true for the first time. Moreover, for fixed values of the leftmost and rightmost symbols of $\sigma = \sigma_1$, the sequence of pairs $(\pi_1, \pi'_1), \ldots, (\pi_i, \pi'_i)$ may assume no more than $F(i)$ distinct values (as argued in Case 1), whereas each of $(\pi_{i+1}, \pi'_{i+1}), \ldots, (\pi_{k-1}, \pi'_{k-1})$ may assume one of 3 values, namely (λ, λ), $(0, 1)$, and $(1, 0)$. Hence, the number of input words $\sigma = \sigma_1$ for which $\mathscr{B}$ will terminate successfully in k iterations, under the assumption that the test of Step 5 is true for the first time in the i-th iteration, $1 \leqq i \leqq k-1$, does not exceed

$$4 \cdot 8 \cdot F(i) \cdot 3^{k-1-i}$$

where "4" is the number of possibilities for the pair (leftmost symbol, rightmost symbol) of σ_1, and "8" the number of possible values for σ_k (an overlap-free word of length 4,5, or 6 such that $\sigma_k = \tau\tau$ for some $\tau \in \{0, 1\}^*$).

Our evaluation below of the function F is such that $F(i) \geqq 3^i$. Hence

$$4 \cdot 8 \cdot F(i) \cdot 3^{k-1-i} \leqq 32 \cdot F(k-1).$$

Putting Case 1 and Case 2 together, the number of input words for which $\mathscr{B}$ terminates successfully in $k \geqq 2$ iterations cannot exceed $k \cdot 48 \cdot F(k-1)$. (Case 2 covers $k-1$ subcases, one for every value of $i \in \{1, \ldots, k-1\}$.)

It remains to evaluate the function $F(h)$ for $h \geqq 1$. It is not difficult to prove that this function satisfies the recurrence relation: $F(h+2) = 4F(h+1) - F(h)$, for $h \geqq 1$, with initial conditions $F(1) = 5$ and $F(2) = 19$. Using standard techniques, the solution of this recurrence relation is:

$$F(h) = (1/2)\left[(1+\sqrt{3})(2+\sqrt{3})^h + (1-\sqrt{3})(2-\sqrt{3})^h\right]$$

The second term contributes very little to the rate of growth of $F(2)$. In fact, for all $h \geqq 1$:

$$F(h) < (1.4)(2+\sqrt{3})^h = (1.4)(3.732)^h.$$

Hence, our desired bound $k \cdot 48 \cdot F(k-1)$ does not exceed

$$48 \cdot (\log n) \cdot (1 \cdot 4) \cdot (3.732)^{(\log n)-2} < 5 \cdot (\log n) \cdot n^{1.9},$$

which is $\mathcal{O}(n^e)$ for some $e < 2$. ■

A more careful analysis limits further the rate of growth of the function F in the preceding proof, by considering more closely the relationship between $\pi_i = xx$ and the possible values for π_{i+1}, π_{i+2}, and π_{i+3}. The resulting recurrence for F becomes

$$F(h+4) = 4F(h+3) - 3F(h+2) + 2F(h).$$

The solution of this recurrence in turn allows us to set $e < 1.7$ in the statement of the theorem.

ACKNOWLEDGMENTS

Special thanks are due to Max Crochemore for his useful comments on an earlier version of this report. Jean Berstel guided me through the recent literature on square-free and overlap-free words, in particular [5]. The report [5] investigates the connection between finite and infinite overlap-free binary words, which we have independently carried out and reported elesewhere [3].

REFERENCES

1. J. Berstel, *Some Recent Results on Squarefree Words*, Proc. of Symposium on Theoretical Aspects of Computer Science, 11-13 April 1984, Paris.
2. M. Crochemore, *Linear Searching for a Square in a Word*, Bulletin of EATCS, No. 24, Oct. 1984, pp. 66-72.
3. A. J. Kfoury, *Square-Free and Overlap-Free Words*, Technical Report, Department of Computer Science, Boston University, 1985.
4. M. Lothaire, *Combinatorics on Words*, Addison-Wesley, 1983.
5. A. Restivo and S. Salemi, *Overlap Free Words on Two Symbols*, in *Automata on Infinite Words*, Nivat and Perrin Ed., LNCS 192, Springer-Verlag, 1985, pp. 198-206.
6. A. Salomaa, *Jewels of Formal Language Theory*, Computer Science Press, 1981.