Maximal Repetitions in Words or How to Find all Squares in Linear Time

Roman Kolpakov, Gregory Kucherov

Maximal Repetitions in Words or How to Find all Squares in Linear Time *

Roman Kolpakov† and Gregory Kucherov

INRIA-Lorraine/LORIA
B.P. 101
54602 Villers-lès-Nancy, France
e-mail: {roman,kucherov}@loria.fr.

Abstract

A (fractional) repetition in a word w is a subword with the period of at most half of the subword length. We study maximal repetitions occurring in w, that is those for which any extended subword of w has a bigger period. The set of such repetitions represents in a compact way all repetitions in w.

We first count the exact number of maximal repetitions in Fibonacci words. Then we prove our main result asserting that the maximal number of such repetitions in general words (on arbitrary alphabet) is linear in the word length. We then show how this result implies a linear-time algorithm for finding all maximal repetitions.

1 Introduction

1.1 Repetitions and their classification

Repetitions (called also periodicities) play a fundamental role in many topics of word combinatorics, formal language theory and applications. Several notions of repetition has been used in the literature. In its simplest form, a repetition is a word of the form uu, commonly called a *square*. A natural generalization is to consider, instead of squares, arbitrary powers, that is words of the form u^n for $n \geq 2$. We call such repetitions *integer repetitions*. If a word is not an integer repetition, it is called *primitive*. Integer repetitions can be further generalized to *fractional repetitions*, that is words of the form $w = u^n v$, where $n \geq 2$ and v is a proper prefix of u. u is called a *root* of w. If u is primitive, quantity $n + \frac{|v|}{|u|}$ is called the *exponent* of w, and $|u|$ is the *period* of w. Considering repetitions with fractional exponent may turn to be very useful and may provide a deeper insight of combinatorial properties of words [MP92, MRS95, CS96].

Depending on the problem, the difference between the above three notions of repetition may not be relevant (for example if one wants to check whether a word is repetition-free) but, as will be seen below, may be important. Besides, if one wants to find (or to count) all repetitions in a word, it must be specified whether all *distinct* repetitions are looked for (that is, their position

*The work has been done within a joint project of the French-Russian A.M.Liapunov Institut of Applied Mathematics and Informatics at Moscow University

†On leave from the French-Russian Institute for Informatics and Applied Mathematics, Moscow University, 119899 Moscow, Russia. Supported by a grant from the French Ministry of Public Education and Research.

in the word is not relevant) or all the occurrences of (possibly syntactically equal) repetitions. In this paper we will be mainly concerned with the latter case, and we will sometimes say *positioned repetitions* to underline this meaning.

Considering positioned integer or fractional repetitions in a word leads to the notion of *maximal repetition*. Informally, a maximal repetition is one which cannot be extended to a bigger repetition with the same period. In case of integer repetitions, this amounts to those repetitions u^k, $k \geq 2$, which are not followed or preceeded by another occurrence of u. In case of fractional repetitions, a maximal repetition is a subword $u^n v$ (v a prefix of u, $n \geq 2$) which cannot be extended *by one letter* to the right or to the left without changing (increasing) the period. For example, the subword 10101 in the word $w = 1011010110110$ is a maximal repetition (with period 2), while the subword 1010 is not. Another maximal repetitions of w are prefix 10110101101 (period 5), suffix 10110110 (period 3), prefix 101101 (period 3), and the three occurrences of 11 (period 1).

In this paper we study maximal positioned fractional repetitions that, for the sake of shortness, we will call *m-repetitions*.[1]. m-repetitions are very important objects as they completely characterize in a compact way all the repetitions in the word. For example, if we know all m-repetitions in a word, we can easily obtain all squares in this word, with both primitive and non-primitive roots.

1.2 Number of repetitions in a word

The question "How many repetitions can a word contain?" is interesting from both theoretical and applicative perspective. However, one must specify carefully which repetitions are counted.

A word of length n contains $O(n \log n)$ positioned primitively-rooted squares. This follows, in particular, from Lemma 10 of [CR95] which asserts that a word cannot contain in its prefixes more than $\log_\phi n$ primitive-rooted squares which immediately implies the $n \log_\phi n$ upper bound (ϕ is the golden ratio). On the other hand, in [Cro81] it was shown that Fibonacci words contain $\Omega(n \log n)$ positioned squares. Since all squares in Fibonacci words are primitively-rooted, this proves that $O(n \log n)$ is the asymptotically tight bound. A formula for the exact number of squares in Fibonacci words has been obtained in [FS98a].

The situation is different if only distinct squares are counted. In [FS98a], it is shown that the k-th Fibonacci word f_k contains $2(|f_{k-2}| - 1) = 2(2 - \phi)|f_k| + o(1)$ distinct squares. The number of distinct squares in general words of length n is bounded by $2n$ (for an arbitrary alphabet) which was shown in [FS98b] using a result from [CR95]. It is conjectured that this number is actually smaller than n, at least for the binary alphabet. Thus, in contrast to positioned squares, the maximal number of distinct squares is linear.

In [Cro81], Crochemore studies positioned primitively-rooted maximal integer repetitions. Similar to positioned squares, the maximal number of such repetitions is $\Theta(n \log n)$. The lower bound easily follows from the $\Omega(n \log n)$ bound for positioned squares in Fibonacci words, as Fibonacci words don't contain 4-powers, and an occurrence of a 3-power is an extension of two square occurrences. Therefore, the number of maximal integer repetitions in Fibonacci words is at least half the number of positioned squares, and is then $\Theta(n \log n)$.

What happens if we count the number of m-repetitions instead of integer powers or just squares? Note that a word can contain much less m-repetitions than maximal integer powers – it is easy to conceive a word of the form u^3 which contains $\Theta(|u|)$ integer powers but only one m-repetition. What is the maximal number of m-repetitions in a word? The results of [IMS97] imply that Fibonacci words contain a linear number of m-repetitions. This is showed, however, in an indirect way by presenting a linear-time algorithm which enumerates all m-repetitions in a Fibonacci word.

[1] m-repetitions have been called *runs* in [IMS97] and *maximal periodicities* in [Mai89]

In this paper we first obtain directly the exact number of m-repetitions in Fibonacci words. Incidentally (or maybe not?), a Fibonacci word contains one less maximal repetitions than distinct squares.

We then prove a main result asserting that a word of length n over an arbitrary alphabet contains $O(n)$ m-repetitions, which contrasts to the above results about the $O(n \log n)$ number of repeated squares or integer repetitions.

1.3 Searching for repetitions

The problem of searching for repetitions in a string is a classical pattern matching problem (see [CR94]) which has been studied by many authors. In early 80s, Slisenko [Sli83] claims a linear (real-time) algorithm for finding all *distinct* m-repetitions in a word. Independently, Crochemore [Cro83] described a simple linear algorithm for finding a square in a word (and thus checking if a word is repetition-free). Another linear algorithm checking whether a word contains a square was proposed in [ML85].

However, as the counting results imply, there is no hope to construct a linear algorithm to explicitely find all positioned squares in a word as their number is super-linear. There are several different $O(n \log n)$ algorithms finding all occurrences of repetitions in a string. Note however that each of these algorithms uses its own notion of repetition. In 1981, Crochemore [Cro81] proposed an $O(n \log n)$ algorithm for finding all occurrences of primitively-rooted maximal integer repetitions in a word. Using a suffix tree technique, Apostolico and Preparata [AP83] described an $O(n \log n)$ algorithm for finding all positioned *right-maximal* fractional repetitions, which are fractional repetitions that cannot be extended *to the right* without increasing the period. Finally, Main and Lorentz [ML84] proposed another algorithm which actually finds all m-repetitions in $O(n \log n)$ time. They also point out the optimality of this bound under the assumption of unbounded alphabet and under the restriction that the algorithm is based only on symbol comparisons.

As far as other related works are concerned, Kosaraju [Kos94] describes an $O(n)$ algorithm which, given a word, finds for each position the shortest square starting at this position. He also claims a generalization which finds all primitively-rooted squares in time $O(n + S)$ where S is the number of such squares. Very recently, Stoye and Gusfield [SG98] proposed several algorithms that achieve known bounds, but are based on a unified suffix tree framework. Their results are based on an algorithm which finds in time $O(n \log n)$ all "branching tandem repeats" (cf Section 6). Based on this result, they show how to find all positioned primitively-rooted squares, and more generally all primitively-rooted right-maximal integer repetitions, in $O(n \log n)$ time, and how to find all positioned (not necessarily primitively-rooted) squares in time $O(n \log n + T)$ where T is the number of such squares.

Using an idea of Crochemore [Cro81], Main [Mai89] proposed a *linear* algorithm which finds all *leftmost* occurrences of distinct m-repetitions in a word. However, so far it has been an open question whether a *linear* algorithm for finding *all* m-repetitions exists. In the concluding section of [Mai89], Main speculates that such an algorithm might exist. The same question is raised in [IMS97] where the authors showed that all maximal repetitions in a Fibonacci word can be listed in linear time.

Using our result about the linear maximal number of m-repetitions in a word, we show that a minor modification of Main's algorithm [Mai89] gives a linear algorithm finding all m-repetitions.

1.4 Paper organization

The paper is organized as follows. In Section 2 we define main notions and give basic results that will be used later. In Section 3 we count the exact number of m-repetitions in Fibonacci words. This result is independent from the rest of the paper. Section 4 is devoted to our main result asserting that the number of m-repetitions in general words can be bounded by a linear function in the length. The proof occupies the whole section. In Section 5 we apply this result to derive a linear-time algorithm for finding all m-repetitions in a word. Finally, in Section 6 we make some concluding remarks and computer experiments.

2 Further definitions and basic results

Consider a word $w = a_1 \dots a_n$. Any word $a_i \dots a_j$ for $i \leq j$, which we denote $w[i..j]$, is a *subword* of w. A position in w is an integer number between 0 and n. Each position π in w defines a decomposition $w = w_1 w_2$ where $|w_1| = \pi$. The position of letter a_i in w is $i - 1$. If $v = w[i..j]$, we denote $initpos(v) = i - 1$ and $endpos(v) = j$. We say that subword $v = w[i..j]$ *crosses* a position π in w, if $i \leq \pi < j$.

If w is a subword of u^n for some natural n, $|u|$ is called a *period* of w, and word u is a *root* of w. Clearly, p is a period of $w = a_1 \dots a_n$ iff $a_i = a_{i+p}$ whenever $1 \leq i, i + p \leq n$. Another equivalent definition is (see [Lot83]): p is a period of $w = a_1 \dots a_n$ iff $w[1..n - p] = w[p + 1..n]$. The last definition shows that each word w has the minimal period that we will denote $p(w)$ and call often simply *the* period of w. The ratio $\frac{|w|}{p(w)}$ is called the *exponent* of w and denoted $e(w)$. Clearly, a root u of w such that $|u| = p(w)$, is *primitive*, that is u cannot be written as v^n for $n \geq 2$. Following [Lot83, Chapter 8], we call the roots u with $|u| = p(w)$ *cyclic roots*. We also call the cyclic root $w[1..p(w)]$ the *prefix cyclic root* of w, and the cyclic root $w[n - p(w) + 1..n]$ the *suffix cyclic root* of w.

Consider $w = a_1 \dots a_n$. A *repetition* in w is any subword $r = w[i..j]$ with $e(r) \geq 2$. A *maximal repetition* in w, called for short an *m-repetition*, is a repetition $r = w[i..j]$ such that

(i) if $i > 1$, then $p(w[i - 1..j]) > p(w[i..j])$,

(ii) if $j < n$, then $p(w[i..j + 1]) > p(w[i..j])$.

In other words, an m-repetition is a repetition $r = w[i..j]$ such that no subword of w which contains r as a proper subword has the same minimal period as r. Note that any repetition in a word can be extended to a unique m-repetition, that we will call the *corresponding* m-repetition. For example, the repetition 1010 in word $w = 1011010110110$ corresponds to the m-repetition 10101 obtained by one letter extension to the right.

A basic result about periods is the Fine and Wilf's theorem (see [Lot83]):

Theorem 1 (Fine and Wilf) *If w has periods p_1, p_2, and $|w| \geq p_1 + p_2 - gcd(p_1, p_2)$, then $gcd(p_1, p_2)$ is also a period of w.*

The following Lemma states some useful facts about m-repetitions that will be used in the sequel.

Lemma 1 *(i) Two distinct m-repetitions with the same period p cannot have an overlap of length greater than or equal to p,*

(ii) Two m-repetitions with minimal periods p_1, p_2, $p_1 \neq p_2$, cannot have an overlap of length greater than or equal to $2\max\{p_1, p_2\}$.

Proof: Part (i) is easily proved by analyzing relative positions of two repetitions of period p and showing that if they intersect on at least p letters, at least one of them is not maximal. Part (ii) is a consequence of Fine and Wilf's theorem. If the intersection is at least $(p_1 + p_2 - gcd(p_1, p_2))$ long, then at least one of the cyclic roots of the two repetitions is not primitive, which is a contradiction. □

A repetition r is said to have a period in some subword of w if r overlaps with this subword on at least $p(r)$ letters. Also, we say that a repetition r has a period on the right (respectively on the left) of a position π with the meaning that $w[\pi + 1..\pi + p(r)]$ (respectively $w[\pi - p(r) + 1..\pi]$) is a subword of r.

The following reformulation of Lemma 1(i) will be often used in Section 4.

Corollary 1 *If two m-repetitions of w have a period on the right (on the left) of the same position π, then either these m-repetitions have different periods or they coincide.*

We will also use the following known Proposition which is also a consequence of Fine and Wilf theorem.

Proposition 1 *If u is a primitive word, then u cannot be an internal subword of uu (that is, a subword which is not a prefix or suffix).*

Proof: If u is an internal subword of uu, then $u = v_1v_2 = v_2v_1$ where both v_1 and v_2 are non-empty. This means that both v_1 and v_2 are roots of u, that is both $|v_1|$ and $|v_2|$ are periods of u. Since $|u| \geq |v_1| + |v_2| - gcd(|v_1|, |v_2|)$, $gcd(|v_1|, |v_2|)$ is also a period of u, which implies that u is an integer power. This contradicts the condition that u is primitive. □

We conclude this section with the following simple Lemma which will be also often used in Section 4. $\#S$ denotes the cardinality of a set S.

Lemma 2 *Let $S \subseteq \mathbb{R}$ be a set of real numbers, and each number of S belongs to interval $[a, b]$ (that is, $S \subseteq [a, b]$). Assume that there is $\Delta \in \mathbb{R}$ such that for every $x, y \in S$, $|x - y| \geq \Delta$. Then $\#S \leq (b - a)/\Delta + 1$.*

All logarithms are binary unless the base is indicated.

3 The number of m-repetitions in Fibonacci words

Fibonacci words are words over the binary alphabet $\{0, 1\}$ defined recursively by $f_0 = 0$, $f_1 = 1$, $f_n = f_{n-1}f_{n-2}$ for $n \geq 2$. The length of f_n, denoted F_n, is the n-th Fibonacci number. Fibonacci words have numerous interesting combinatorial properties and often provide a good example to test conjectures and analyse algorithms on words.

As it was noted in the Introduction, the Fibonacci word f_n contains $O(F_n \log F_n)$ squares all of which are primitively-rooted. In [FS98a], the exact number of squares in Fibonacci words has been obtained, which is asymptotically $\frac{2}{5}(3 - \phi)nF_n + O(F_n)$. Since general words of length n contain $O(n \log n)$ primitively-rooted squares [CR95], Fibonacci words contain asymptotically maximal number of primitively-rooted squares.

In this section, we count the number of m-repetitions in Fibonacci words. Let R_n be the number of m-repetitions in f_n. We prove the following

Theorem 2 *For all* $n \geq 4$, $R_n = 2F_{n-2} - 3$.

We follow the general proof scheme used in [FS98a] for counting the number of repeated squares. Consider the decomposition $f_n = f_{n-1}f_{n-2}$ and call the position between f_{n-1} and f_{n-2} the *boundary*. Clearly, the m-repetitions in f_n are divided into those which lie entirely in f_{n-1} or f_{n-2} and those which cross the boundary, that is intersect with f_{n-1} (call this intersection the left part) and with f_{n-2} (right part). Note first that the left part and the right part of an m-repetition cannot be both of exponent ≥ 2, since Fibonacci words don't have subwords of exponent 4. If either the left or the right part is of exponent ≥ 2, then the m-repetition is an extension of an m-repetition of respectively f_{n-1} or f_{n-2}. This implies that the only new m-repetitions of f_n that should be counted are those that cross the boundary but don't have their right and left part of exponent ≥ 2. Denote $c(n)$ the number of such m-repetitions that we will call *composed* m-repetitions of f_n. Then

$$R_n = R_{n-1} + R_{n-2} + c(n).$$

The following argument gives the solution.

Lemma 3 *For all* $n \geq 8$, $c(n) = c(n-2)$.

Consider the representation

$$f_n = f_{n-1}|f_{n-2} = f_{n-2}f_{n-3}|f_{n-3}f_{n-4} = f_{n-2}[f_{n-3}|f_{n-4}]f_{n-5}f_{n-4} \tag{1}$$

where $|$ denotes the boundary, $n \geq 5$, and square brackets delimit the occurrence of f_{n-2} with the same boundary as for the whole word f_n. It is known that every m-repetition in Fibonacci words has the period F_k for some k (this is mentioned in [FS98a] as a "folklore" result, proved in [Séé85]). Since $F_{n-3} > F_{n-4} > 2F_{n-6}$, it follows from (1) that if a composed m-repetition of f_n has the period F_k for $k \leq n-6$, then it is also a composed m-repetition of f_{n-2} and therefore is counted in $c(n-2)$. Vice versa, every composed m-repetition of f_{n-2} with period F_k for $k \leq n-6$, is also a composed m-repetition of f_n. We now examine the m-repetitions of f_n with periods F_{n-2}, F_{n-3}, F_{n-4}, F_{n-5} which cross the boundary.

m-repetitions with period F_{n-2}. The last term of (1) shows that square $(f_{n-2})^2$ is a prefix of f_n that crosses the boundary. As $F_{n-1} < 2F_{n-2}$, the corresponding m-repetition does not have a square in its left or right part and therefore is composed for f_n. Since $F_{n-2} > F_n/3$, any two m-repetitions of f_n with period F_{n-2} intersect by more than F_{n-2} letters. By Lemma 1(i), this shows that f_n has only one m-repetition with period F_{n-2}. Trivially, the m-repetition under consideration is not an m-repetition of f_{n-2}.

m-repetitions with period F_{n-3}. From the decomposition $f_n = f_{n-2}f_{n-3}|f_{n-3}f_{n-4}$ (see (1)), there is a square $(f_{n-3})^2$ with the root length F_{n-3} crossing the boundary. The corresponding m-repetition does not extend to the left of the left occurrence of f_{n-3}, as the last letters of f_{n-3} and f_{n-2} are different (the last letters of f_i's alternate). Therefore, this m-repetition does not have a square in its left or right part, and thus is composed for f_n. As this m-repetition has a period both on the left and on the right of the boundary, it is the only m-repetition with period F_{n-3} crossing the boundary (see Corollary 1). Again, from length considerations, it is not an m-repetition of f_{n-2}.

m-repetitions with period F_{n-4}. Since $f_n = f_{n-1}|f_{n-4}f_{n-5}f_{n-4} = f_{n-1}|f_{n-4}f_{n-5}f_{n-5}f_{n-6} = f_{n-1}|f_{n-4}f_{n-5}f_{n-6}f_{n-7}f_{n-6} = f_{n-1}|(f_{n-4})^2 f_{n-7}f_{n-6}$, there is a square $(f_{n-4})^2$ on the right of the boundary. However, this m-repetition does not extend to the left (i.e. does not cross the boundary), since the last letters of f_{n-4} and f_{n-1} are different.

On the other hand, $f_n = f_{n-2}[f_{n-3}|f_{n-4}]f_{n-5}f_{n-4} = f_{n-3}f_{n-4}[f_{n-4}f_{n-5}|f_{n-5}f_{n-6}]f_{n-5}f_{n-4} = f_{n-3}f_{n-4}[f_{n-4}\underbrace{f_{n-5}|f_{n-6}}_{f_{n-4}}f_{n-7}f_{n-6}]f_{n-5}f_{n-4}$ for $n \geq 6$. This reveals an m-repetition of period F_{n-4} which crosses the boundary. However, this is not a composed m-repetition of f_n, as it has a square on the left of the boundary. On the other hand, the restriction of this m-repetition to f_{n-2} (subword in square brackets) *is* a composed m-repetition for f_{n-2} (exactly in the same way as the m-repetition with period F_{n-2} is a composed m-repetition of f_n in the first case above).

There is no other m-repetition of period F_{n-4} crossing the boundary, since such would intersect with one of the two above by more than a period length, which would contradict Lemma 1(i). In conclusion, there is one composed m-repetition of period F_{n-4} in f_{n-2} and no such m-repetition in f_n.

m-repetitions with period F_{n-5}. Rewrite $f_n = f_{n-2}[f_{n-4}f_{n-5}|f_{n-5}f_{n-6}]f_{n-5}f_{n-4}$ which shows that there is a square of root length F_{n-5} crossing the boundary. Since the boundary is the center of this square, the latter corresponds to the only m-repetition with period F_{n-5} crossing the boundary. However, this m-repetition is not a composed m-repetition for f_n, as it has a square in its right part, as shown by the following transformation: $f_n = f_{n-2}[f_{n-4}f_{n-5}|f_{n-5}f_{n-6}]f_{n-6}f_{n-7}f_{n-4} = f_{n-2}[f_{n-4}f_{n-5}|f_{n-5}\underbrace{f_{n-6}]f_{n-7}}_{f_{n-5}}f_{n-8}f_{n-7}f_{n-4}$ for $n \geq 8$. On the other hand, the restriction of this m-repetition to f_{n-2} (subword in square brackets) *is* a composed m-repetition for f_{n-2} (in the same way as the m-repetition with period F_{n-3} is composed for f_n in the second case above). Thus, there is one composed m-repetition of the period F_{n-5} in f_{n-2} and no such m-repetition in f_n.

In conclusion, two new composed m-repetitions arise in f_n in comparison to f_{n-2}, but two composed m-repetitions of f_{n-2} are no more composed in f_n, as they extend in f_n to form a square in its right or left part. This shows that $c(n) = c(n-2)$ for $n \geq 8$ and proves the Lemma.

A direct counting shows that $R_0 = 0,\ R_1 = 0,\ R_2 = 0,\ R_3 = 0,\ R_4 = 1,\ R_5 = 3,\ R_6 = 7,\ R_7 = 13$. Therefore, $c(3) = 0,\ c(4) = 1,\ c(5) = 2,\ c(6) = 3,\ c(7) = 3$. Since $c(n) = c(n-2)$ for all $n \geq 8$, then $c(n) = 3$ for all $n \geq 6$. We then have the recurrence relation $R_n = R_{n-1} + R_{n-2} + 3$ for $n \geq 6$ with boundary conditions $R_4 = 1,\ R_5 = 3$. Substituting $R_n = 2R'_{n-2} - 3$, we get the relation $R'_n = R'_{n-1} + R'_{n-2}$ for $n \geq 4$, where $R'_2 = 2,\ R'_3 = 3$. This defines exactly the Fibonacci numbers. Thus, $R'_n = F_n$ for $n \geq 2$, and $R_n = 2F_{n-2} - 3$ for $n \geq 4$. Theorem 2 is proved.

4 Maximal number of m-repetitions in a word

Fibonacci words are known to contain "many" repetitions. The result of Section 3 suggests the following question: Is it true that general words contain only a linear number of m-repetitions? In this section we answer this question affirmatively. We prove that the maximal number of m-repetitions in words of length n is a linear function on n, regardless of the underlying alphabet. Denote by $Rep(n)$ the maximal number of m-repetitions in words of length n (the alphabet is not fixed).

Theorem 3 *$Rep(n) = O(n)$.*

The proof of Theorem 3 occupies the rest of this section. Actually, we prove the following statement.

Theorem 4 *There exist absolute positive constants C_1, C_2 such that*

$$Rep(n) \leq C_1 n - C_2\sqrt{n}\log n \tag{2}$$

We assume, without loss of generality, that C_1 is sufficiently bigger than C_2, say $C_1 \geq 2C_2$, so that the function $C_1 x - C_2\sqrt{x}\log x$ is monotonically increasing for all $x \geq 1$. In the proof, we use induction over n. For technical reasons we assume that $n \geq 81$ ($\sqrt{n} \geq 9$). The base cases ($n < 81$) are trivially satisfied, as constant C_1 can be chosen as large as we need.

Recall that $e(w)$ and $p(w)$ denote respectively the exponent and the period of w.

Take any $n \geq 81$, and a word $w = a_1 \ldots a_n$ of length n. We split the proof into two major cases depending on whether or not w contains an m-repetition of "big" exponent. Formally, denote $\mathcal{BR}$ the set of words w containing an m-repetition of exponent $\lceil\sqrt{n}\rceil$ or more.

Case 1 ($w \notin \mathcal{BR}$): Let $w \notin \mathcal{BR}$. Consider the letter $a_{\lceil\frac{n}{2}\rceil}$, and write $w = w_1 a_{\lceil\frac{n}{2}\rceil} w_2$, where $|w_1|, |w_2| \leq \frac{n}{2}$. Then $Rep(w) \leq Rep(|w_1|) + Rep(|w_2|) + \#CR(w)$, where $CR(w)$ is the set of m-repetitions in w containing $a_{\lceil\frac{n}{2}\rceil}$. By induction,

$$Rep(|w_1|) + Rep(|w_2|) \leq 2Rep(\lfloor\frac{n}{2}\rfloor) \leq C_1 n - \sqrt{2}C_2\sqrt{n}\log\frac{n}{2}. \tag{3}$$

We now turn to estimating $\#CR(w)$. Our goal is to prove that $\#CR(w) = O(\sqrt{n}\log n)$.

We decompose $CR(w) = CRL(w) \cup CRR(w)$, where $CRL(w)$ (respectively $CRR(w)$) are those m-repetitions of $CR(w)$ which have a period on the left (respectively on the right) of $a_{\lceil\frac{n}{2}\rceil}$. Note that $CRL(w)$ and $CRR(w)$ are generally not disjoint.

Case 1.1 ($CRL(w)$): Consider $CRL(w)$. Observe that by Corollary 1, no two distinct m-repetitions from $CRL(w)$ have the same period. We further split $CRL(w)$ into two non-intersecting subsets, $CRL1(w)$ and $CRL2(w)$, where $CRL1(w)$ (respectively $CRL2(w)$) consists of those m-repetitions $r \in CRL(w)$ which have $p(r)/2$ or more (respectively, less than $p(r)/2$) letters on the right of $a_{\lceil\frac{n}{2}\rceil}$.

Case 1.1.1 ($CRL1(w)$): Consider $r_1, r_2 \in CRL1(w)$ with periods $p(r_1)$, $p(r_2)$ respectively. By the remark above, $p(r_1) \neq p(r_2)$, and assume that $p(r_1) > p(r_2)$, and $\Delta = p(r_1) - p(r_2)$. Consider the (non-empty) word $v = w[\pi_b + 1..\pi_e]$, where $\pi_b = \max\{initpos(r_1) + p(r_1), initpos(r_2) + p(r_2)\}$ and $\pi_e = \min\{endpos(r_1), endpos(r_2)\}$. Observe that v is a subword of both r_1 and r_2 which occurs, in each of them, at least one period away from the beginning. Then v has two other occurrences at positions $\pi_b - p(r_1)$ and $\pi_b - p(r_2)$. Consider word $v' = w[\pi_b - p(r_1) + 1..\pi_b - p(r_2) + |v|]$. Observe that $|v'| = |v| + \Delta$, and v' has a period Δ, as v occurs both as a prefix and a suffix of v'. The situation is illustrated in Figure 1.[2] Since $w \notin \mathcal{BR}$, we can bound $\frac{|v'|}{\Delta} = \frac{|v|}{\Delta} + 1 \leq \lceil\sqrt{n}\rceil$. Since v contains the subword $w[\lceil\frac{n}{2}\rceil..\pi_e]$ of length at least $p(r_2)/2$, we have $|v| \geq p(r_2)/2$. We then have $\frac{p(r_2)}{2\Delta} \leq \sqrt{n}$ which implies $\frac{p(r_2)}{p(r_1)} \leq 1 - \frac{1}{2\sqrt{n}+1}$. Turning to logarithms, $\log p(r_2) - \log p(r_1) \leq \log(1 - \frac{1}{2\sqrt{n}+1}) \leq$

[2] Note that Figure 1 depicts only one of possible situations. E.g., the end position of r_1 may be smaller than that of r_2, left occurrences of v may not intersect, etc.

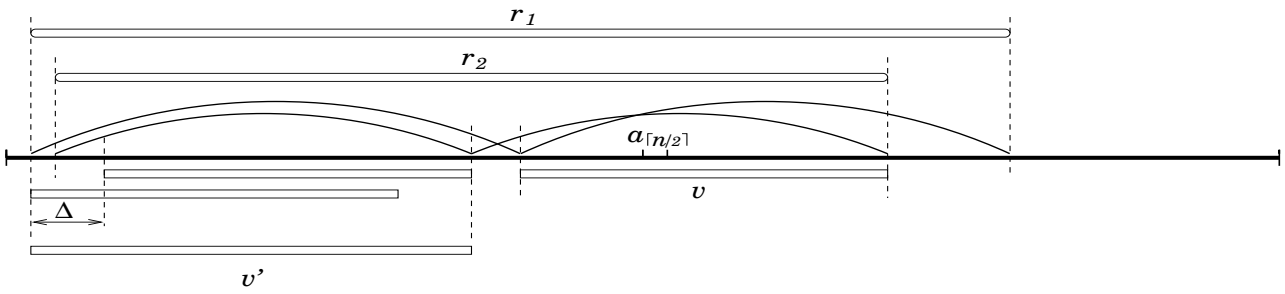

Figure 1: Illustration to Case 1.1.1

$-\frac{1}{2\sqrt{n}+1}$, as $\log(1-x) \leq -x$ for $0 \leq x < 1$. Therefore, $\log p(r_1) - \log p(r_2) \geq \frac{1}{2\sqrt{n}+1}$. Recall that each m-repetition r from $CRL1(w)$ has a distinct period $p(r)$ and hence a distinct value $\log p(r)$. On the other hand, $\log p(r)$ can vary from 0 to $(\log n - 1)$. By Lemma 2, there are at most $(\log n - 1)(2\sqrt{n}+1) + 1 = O(\sqrt{n}\log n)$ distinct values $\log p(r)$, and therefore that many m-repetitions in $CRL1(w)$.

Case 1.1.2 ($CRL2(w)$): For $CRL2(w)$, the proof is exactly the same as in Case 1.1.1 except that here v contains the subword $w[\max\{endpos(r_1) - p(r_1), endpos(r_2) - p(r_2)\} + 1..\lceil \frac{n}{2} \rceil]$ of length at least $p(r_2)/2$ which implies that $|v| \geq p(r_2)/2$. Thus, $CRL2(w)$ contains at most $O(\sqrt{n}\log n)$ m-repetitions too.

We obtain that $CRL(w)$ contains at most $O(\sqrt{n}\log n)$ m-repetitions.

Case 1.2 ($CRR(w)$): By symmetry, $CRR(w)$ contains also at most $O(\sqrt{n}\log n)$ m-repetitions.

We conclude that $\#CR(w) \leq \#CRL(w) + \#CRR(w) = O(\sqrt{n}\log n)$. Therefore,

$$Rep(w) \leq C_1 n - \sqrt{2}C_2\sqrt{n}\log\frac{n}{2} + O(\sqrt{n}\log n).$$

To complete the induction step and verify that $Rep(w)$ satisfies inequation (2), we have to make the expression $\sqrt{2}C_2\sqrt{n}\log\frac{n}{2} - O(\sqrt{n}\log n)$ bigger than $C_2\sqrt{n}\log n$. This can be done by picking a sufficiently large constant C_2. The proof of Case 1 is completed.

Case 2 ($w \in \mathcal{BR}$): Let us now turn to the case $w \in \mathcal{BR}$. Let $w = w_1 r w_2$, where r is an m-repetition in w with period $p_r = p(r)$ and exponent $e(r) = \frac{|r|}{p_r} \geq \lceil\sqrt{n}\rceil$. Denote $e_r = e(r)$. Since $n \geq 81$, then $e_r \geq 9$. Denote $\pi_{init} = initpos(r)$, $\pi_{end} = endpos(r)$. We now split r into three approximately equal parts, each having at least 3 periods. Formally, we find positions $\pi_{left} = \pi_{init} + \lfloor\frac{|r|}{3}\rfloor$, $\pi_{right} = \pi_{end} - \lfloor\frac{|r|}{3}\rfloor$. We now classify all m-repetitions into four non-intersecting classes:

$LR(w)$: m-repetitions entirely lying in $w[1..\pi_{left}]$,

$RR(w)$: m-repetitions entirely lying in $w[\pi_{right}+1..n]$,

$BCR(w)$: m-repetitions not contained in $LR(w), RR(w)$ and having a period p_r or more,

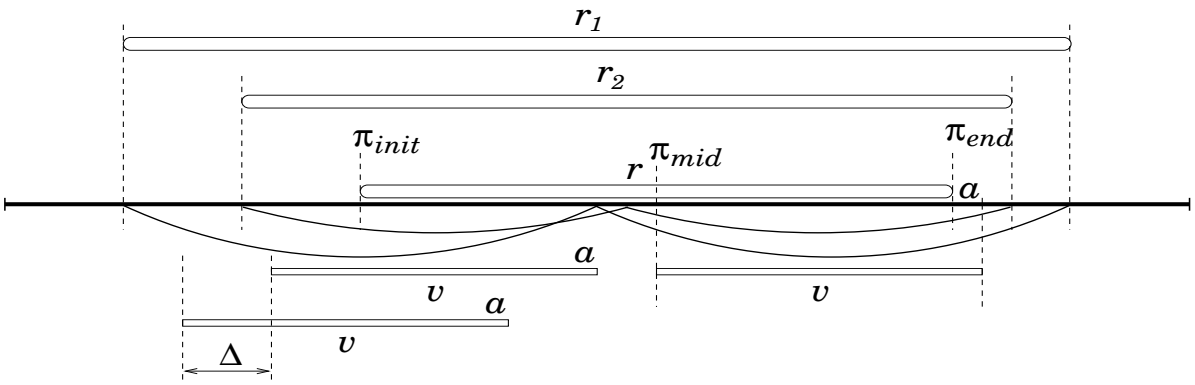

Figure 2: Illustration to Case 2.1.1.1

$SCR(w)$: m-repetitions not contained in $LR(w), RR(w)$ and having a period smaller than p_r.

The goal is now to estimate the cardinality of each of these classes. As for $LR(w), RR(w)$, they will be estimated later using the induction hypothesis. Let us focus on $BCR(w)$.

Case 2.1 ($BCR(w)$): Our goal is to prove that $\#BCR(w) = O(e_r)$. The first observation is that m-repetitions of $BCR(w)$, except for r itself, cannot lie entirely inside r as this would contradict Lemma 1(ii). Thus, any m-repetition of $BCR(w)$ contains at least one of the letters $a_{\pi_{init}}, a_{\pi_{end}+1}$. We further split $BCR(w)$ according to different possibilities:

$BCR0(w) = \{u \in BCR(w) | initpos(u) < \pi_{init} \text{ and } endpos(w) > \pi_{end}\}$,

$BCR1(w) = \{u \in BCR(w) | initpos(u) \geq \pi_{init} \text{ and } endpos(w) > \pi_{end}\}$,

$BCR2(w) = \{u \in BCR(w) | initpos(u) < \pi_{init} \text{ and } endpos(w) \leq \pi_{end}\}$.

Then $\#BCR(w) = \#BCR0(w) + \#BCR1(w) + \#BCR2(w) + 1$, the one coming from the m-repetition r itself. We proceed by analyzing each of this classes separately.

Case 2.1.1 ($BCR0(w)$): Let us pick the position $\pi_{mid} = \pi_{init} + \lfloor |r|/2 \rfloor$ in the middle of r. We futher divide $BCR0(w)$ into two (possibly intersecting) subsets. Let $BCR0'(w)$ consist of those m-repetitions of $BCR0(w)$ that have a period on the left of π_{mid}, and $BCR0''(w)$ of those that have a period on the right of π_{mid}.

Case 2.1.1.1 ($BCR0'(w)$): Consider two m-repetitions $r_1, r_2 \in BCR0'(w)$. By Corollary 1, $p(r_1) \neq p(r_2)$. Assume $p(r_1) > p(r_2)$. Consider the word $v = w[\pi_{mid}+1..\pi_{end}+1]$. Note that $a_{\pi_{end}+1}$ is the letter right after the end of the m-repetition r, which implies that $a_{\pi_{end}+1} \neq a_{\pi_{end}+1-p_r}$. Note also that any proper prefix of v is a part of r and then has a period p_r. Word v belongs to both r_1 and r_2 and starts, in each of them, at least one period away from the beginning. Then v has two other occurrences starting at positions $\pi_{mid} - p(r_1)$ and $\pi_{mid} - p(r_2)$ (see Figure 2). The shift between these occurrences is $\Delta = p(r_1) - p(r_2)$ and we claim that $\Delta \geq |v| - p_r$. Otherwise, if $\Delta < |v| - p_r$, then the two occurrences of v have an overlap of length at least $p_r + 1$. Since this overlap is a prefix of the occurrence of v starting at $\pi_{mid} - p(r_2)$, it has a period p_r. Since the

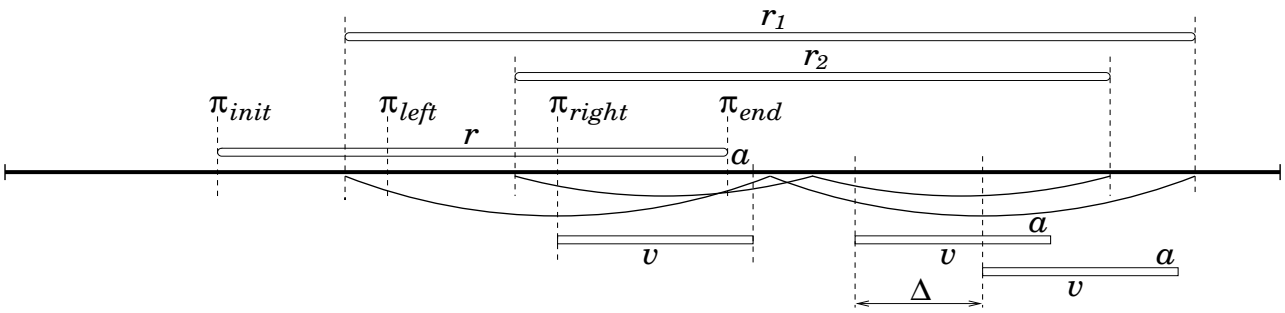

Figure 3: Illustration to Case 2.1.2.1

overlap is also a suffix of the occurrence of v starting at $\pi_{mid} - p(r_1)$ (see Figure 2), we have that $a_{\pi_{end}+1} = a_{\pi_{end}+1-p_r}$ which is a contradiction.

Thus, $\Delta = p(r_1) - p(r_2) \geq |v| - p_r \geq \frac{|r|}{2} - p_r$. As for any $u \in BCR0'(w)$, $p_r < p(u) \leq \frac{n}{2}$, by Lemma 2 we have that $BCR0'(w)$ contains at most $\frac{n/2-p_r}{|r|/2-p_r} + 1$ m-repetitions.

Case 2.1.1.2 ($BCR0''(w)$): This case is symmetric to Case 2.1.1.1. $BCR0''(w)$ contains at most $\frac{n/2-p_r}{|r|/2-p_r} + 1$ m-repetitions too.

Summing up $\#BCR0'(w)$ and $\#BCR0''(w)$, we get $\#BCR0(w) \leq \frac{n-2p_r}{|r|/2-p_r} + 2 \leq \frac{n}{p_r(e_r/2-1)} + 2 \leq \frac{{e_r}^2}{e_r/2-1} + 2 = O(e_r)$.

Case 2.1.2 ($BCR1(w)$): Recall that $BCR1(w)$ consists of the m-repetitions of period p_r or more, starting at a position between π_{init} and π_{right}, and ending at a position greater than π_{end}. We divide the m-repetitions of $BCR1(w)$ into two subsets according to their overlap with r. Let $BCR1'(w)$ be the set of those m-repetitions r' of $BCR1(w)$ which overlap with r on less than $p(r')$ letters, and $BCR1''(w)$ the set of those m-repetitions r' of $BCR1(w)$ which overlap with r on at least $p(r')$ letters.

Case 2.1.2.1 ($BCR1'(w)$): Clearly, any m-repetition $u \in BCR1'(w)$ has at least $p(u)+1$ letters on the right of π_{end}. This implies, by Corollary 1, that no two m-repetitions of $BCR1'(w)$ have the same period. Consider two m-repetitions $r_1, r_2 \in BCR1'(w)$, and assume $p(r_1) > p(r_2)$. Consider the word $v = w[\pi_{right} + 1..\pi_{end} + 1]$ that is contained in both r_1 and r_2. Similar to Case 2.1.1.1, observe that any proper prefix of v has period p_r, and $a_{\pi_{end}+1} \neq a_{\pi_{end}+1-p_r}$. Since v is located in the prefix cyclic root of both r_1 and r_2, it has two copies in w at positions $\pi_{right} + p(r_2)$ and $\pi_{right} + p(r_1)$ (see Figure 3). The shift between these occurrences is $\Delta = p(r_1) - p(r_2)$. Similar to Case 2.1.1.1, if $\Delta < |v| - p_r$, we obtain the contradiction with $a_{\pi_{end}+1} \neq a_{\pi_{end}+1-p_r}$, and therefore we conclude that $\Delta \geq |v| - p_r$.

Since for $u \in BCR1'(w)$, $p_r < p(u) \leq (|r| + |w_2|)/2$, we conclude, by Lemma 2, that $BCR1'(w)$ contains at most $\frac{(|r|+|w_2|)/2-p_r}{|v|-p_r} + 1 \leq \frac{(|r|+|w_2|)/2}{|r|/3-p_r} + 1 \leq \frac{n}{2p_r(e_r/3-1)} + 1 \leq \frac{{e_r}^2}{2e_r/3-2} + 1 = O(e_r)$ m-repetitions.

Case 2.1.2.2 ($BCR1''(w)$): $BCR1''(w)$ consists of those m-repetitions of $BCR1(w)$ which have their prefix cyclic root completely inside r. First, show that for any m-repetition $r' \in BCR1''(w)$, its

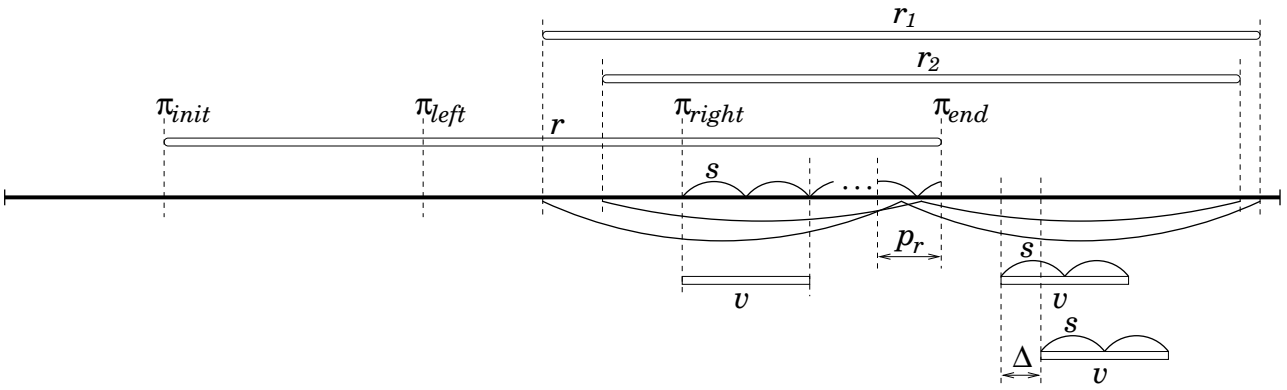

Figure 4: Illustration to Case 2.1.2.2

overlap with r is smaller than $p(r')+p_r$. Indeed, let $\pi_b = initpos(r')$, and consider $u = w[\pi_b+1..\pi_{end}]$ – the overlap of r' and r. Assume $|u| \geq p(r')+p_r$. Observe that $p(r') \neq p(r)$ as this would contradict Lemma 1(ii). Thus, $p(r') > p(r)$. By Fine and Wilf's theorem, u has a period $gcd(p(r'), p_r)$, and this implies that a root of r' of length $p(r')$ is not primitive. This contradicts the fact that $p(r')$ is the minimal period of r'. Thus, the end position π_{end} of r is less than p_r letters away from the end position of the prefix cyclic root of r'.

Consider the word $v = w[\pi_{right} + 1..\pi_{right} + 2p_r]$. By construction, $\pi_{end} - \pi_{right} \geq 3p_r$, and therefore by the above paragraph, v occurs inside the prefix cyclic root of both r_1 and r_2. Consider now $r_1, r_2 \in BCR1''(w)$. By Corollary 1, $p(r_1) \neq p(r_2)$, and assume that $p(r_1) > p(r_2)$. Again, v has two copies at positions $\pi_{right} + p(r_2)$ and $\pi_{right} + p(r_1)$ (see Figure 4). Note that $v = s^2$, where s is a cyclic root of r. Now if the two copies of v have an offset smaller than p_r (see Figure 4), then s occurs as a proper subword of $v = s^2$ which is impossible by Proposition 1 as s primitive. Therefore, $\Delta = p(r_1) - p(r_2) \geq p_r$.

As for any $u \in BCR1''(w)$, $|r|/3 - p_r \leq \pi_{end} - \pi_{right} - p_r + 1 \leq p(u) < |r|$, by Lemma 2, $BCR1''(w)$ contains at most $\frac{2|r|/3+p_r}{p_r} + 1 = \frac{2|r|}{3p_r} + 2 = O(e_r)$ m-repetitions.

In total, $BCR1(w)$ has $O(e_r)$ m-repetitions.

Case 2.1.3 ($BCR2(w)$): This case is symmetric to Case 2.1.2. Thus, $BCR2(w)$ contains $O(e_r)$ m-repetitions too.

Putting together cases 2.1.1, 2.1.2 and 2.1.3, we conclude that $BCR(w)$ contains $O(e_r)$ m-repetitions overall.

Case 2.2 ($SCR(w)$): We now turn to analyzing the set $SCR(w)$ consisting of those m-repetitions which intersect with $w[\pi_{left} + 1..\pi_{right}]$ and having a period smaller than p_r. We introduce $m = \pi_{right} - \pi_{left}$ (the length of $w[\pi_{left}+1..\pi_{right}]$), and $e_{mid} = m/p_r$ (the exponent of $w[\pi_{left}+1..\pi_{right}]$).

First observe that for any fixed position π in $w[\pi_{left} + 1..\pi_{right}]$, there are at most $2p_r$ m-repetitions from $SCR(w)$ crossing this position. This can be seen using our usual technique: If two m-repetitions from $SCR(w)$ have one period on the right (on the left) of π, they cannot have the same period length. Therefore, each of these two subsets cannot have more than p_r distinct m-repetitions and there are no more than $2p_r$ overall m-repetitions crossing π.

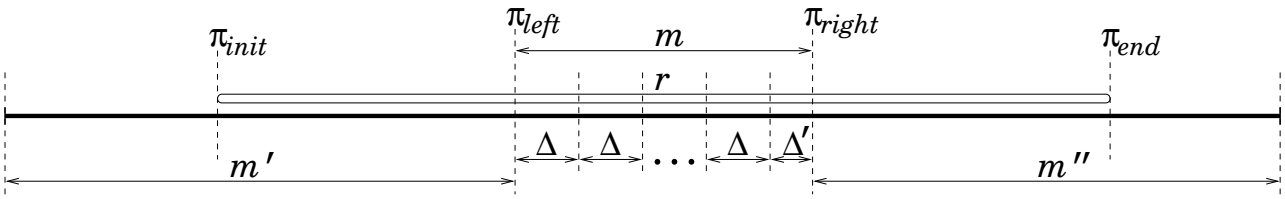

Figure 5: Illustration to Case 2.2

We now split $w[\pi_{left}+1..\pi_{right}]$ into consecutive blocks of length Δ (that will be defined later) with a possible remainder block of length $\Delta' < \Delta$ (see Figure 5). There are then $k = (m-\Delta')/\Delta$ blocks of length Δ. By induction hypothesis, there are at most $Rep(\Delta) \le C_1\Delta - C_2\sqrt{\Delta}\log\Delta$ m-repetitions inside each of them, and at most $Rep(\Delta') \le C_1\Delta' - C_2\sqrt{\Delta'}\log\Delta'$ m-repetitions inside the remainder block. In addition, there are at most $2p_r(k+2)$ m-repetitions crossing the borders delimiting the blocks. We then have the estimate

$$\begin{aligned}
\#SCR(w) &\le k(C_1\Delta - C_2\sqrt{\Delta}\log\Delta) + C_1\Delta' - C_2\sqrt{\Delta'}\log\Delta' + 2p_r(k+2)\\
&= C_1 m - C_2(k\sqrt{\Delta}\log\Delta + \sqrt{\Delta'}\log\Delta') + 2p_r(k+2)\\
&= C_1 m - C_2\frac{m}{\Delta}\sqrt{\Delta}\log\Delta + C_2\left(\frac{\Delta'\log\Delta}{\sqrt{\Delta}} - \sqrt{\Delta'}\log\Delta'\right) + 2p_r(k+2)
\end{aligned}$$

Considering the expression in parentheses as a real function of Δ' defined on the interval $[1..\Delta]$, we observe that this function is convex, and then reaches its maximum on one of the ends of the interval. We then conclude that $\frac{\Delta'\log\Delta}{\sqrt{\Delta}} - \sqrt{\Delta'}\log\Delta'$ reaches the maximum for $\Delta' = 1$ and is then bounded from above by $\frac{\log\Delta}{\sqrt{\Delta}} \le \frac{2\log e}{e} \le \frac{3}{2}$. We then get

$$\#SCR(w) \le C_1 m - C_2 m\frac{\log\Delta}{\sqrt{\Delta}} + \frac{3}{2}C_2 + 2p_r(k+2). \tag{4}$$

Putting everything together. We now count together all m-repetitions of classes $LR(w)$, $RR(w)$, $BCR(w)$, $SCR(w)$. Recall that our goal is to prove

$$\#LR(w) + \#RR(w) + \#BCR(w) + \#SCR(w) \le C_1 n - C_2\sqrt{n}\log n. \tag{5}$$

The numbers $\#LR(w)$, $\#RR(w)$ will be estimated by induction. Denote $m' = \pi_{left}$ (the length of $w[1..\pi_{left}]$), $m'' = n - \pi_{right}$ (the length of $w[\pi_{right}+1..n]$). Thus, $m' + m + m'' = n$. By induction, we have

$$\#LR(w) \le C_1 m' - C_2\sqrt{m'}\log m' \tag{6}$$

$$\#RR(w) \le C_1 m'' - C_2\sqrt{m''}\log m'' \tag{7}$$

Substituting (6), (7) into (5), we have

$$C_1 m' - C_2\sqrt{m'}\log m' + C_1 m'' - C_2\sqrt{m''}\log m'' + \#BCR(w) + \#SCR(w) \le C_1 n - C_2\sqrt{n}\log n,$$

or, since $n - m' - m'' = m$,

$$C_2(\sqrt{n}\log n - \sqrt{m'}\log m' - \sqrt{m''}\log m'') + \#BCR(w) + \#SCR(w) \le C_1 m \tag{8}$$

Let us now estimate the expression in parentheses in the left-hand side. The following lemma follows from routine calculus considerations.

Lemma 4

$$\sqrt{n}\log n - \sqrt{m'}\log m' - \sqrt{m''}\log m'' \leq \frac{m(\log n + 1)}{\sqrt{n}}$$

Proof: The proof is given in Appendix A. □

From Lemma 4 it follows that to prove (8) it is sufficient to prove the following inequality.

$$C_2\frac{m(\log n + 1)}{\sqrt{n}} + \#BCR(w) + \#SCR(w) \leq C_1 m. \tag{9}$$

We have proved (Case 2.1) that $\#BCR(w) = O(e_r)$. Note that by construction, $e_r \leq 3e_{mid}$, and then $\#BCR(w) = O(e_{mid})$. We then rewrite (9) into

$$C_2\frac{m(\log n + 1)}{\sqrt{n}} + O(e_{mid}) + \#SCR(w) \leq C_1 m. \tag{10}$$

Completing the proof for $p_r \geq 8$. To finish the proof, we have to prove inequality (10) using estimate (4) for $\#SCR(w)$. For technical reasons we now assume that $p_r \geq 8$ (the case $p_r < 8$ will be considered separately), and we choose $\Delta = \lfloor \frac{{p_r}^2}{4} \rfloor$. With this choice of Δ, inequation (4) can be transformed as follows. First note that $\frac{\log x}{\sqrt{x}}$ is decreasing for $x \geq e^2$, and $\Delta = \lfloor \frac{{p_r}^2}{4} \rfloor \geq 16$ as $p_r \geq 8$. Therefore, $\frac{\log \Delta}{\sqrt{\Delta}} \geq \frac{\log \frac{p^2}{4}}{\sqrt{\frac{p^2}{4}}} = \frac{4\log p_r - 4}{p_r}$. Turn now to the term $2p_r(k+2) = 2p_r k + 4p_r$ in (4). Since $k = O(\frac{m}{p_r^2}) = O(\frac{e_{mid}}{p_r})$, then $p_r k = O(e_{mid})$, and we estimate $2p_r(k+2)$ as $O(e_{mid})$. We then rewrite (4) as

$$\#SCR(w) \leq C_1 m - 4C_2 e_{mid}(\log p_r - 1) + \frac{3}{2}C_2 + O(e_{mid}) \tag{11}$$

We further estimate $\frac{m(\log n+1)}{\sqrt{n}}$ in (10). The term $\frac{m\log n}{\sqrt{n}}$ is estimated as follows using the fact that $\frac{\log x}{\sqrt{x}}$ is decreasing for $x \geq e^2$. Since $p_r^2 \leq n$, we then have $\frac{m\log n}{\sqrt{n}} \leq m\frac{2\log p_r}{p_r} = 2e_{mid}\log p_r$. The term $\frac{m}{\sqrt{n}}$ can be simply estimated by $\frac{m}{\sqrt{n}} = e_{mid}\frac{p_r}{\sqrt{n}} \leq e_{mid}$. Therefore,

$$\frac{m(\log n + 1)}{\sqrt{n}} \leq e_{mid}(2\log p_r + 1),$$

and to prove (10), it then suffices to prove

$$C_2 e_{mid}(2\log p_r + 1) + O(e_{mid}) + \#SCR(w) \leq C_1 m. \tag{12}$$

Substituting (11) into (12), we have the inequality

$$C_2 e_{mid}(2\log p_r + 1) + C_1 m - 4C_2 e_{mid}(\log p_r - 1) + \frac{3}{2}C_2 + O(e_{mid}) \leq C_1 m,$$

or

$$O(e_{mid}) \leq C_2 e_{mid}(2\log p_r - 5) - \frac{3}{2}C_2. \tag{13}$$

Recalling that $\log p_r \geq 3$ and $e_{mid} \geq 3$, inequation (13) can be satisfied by choosing a sufficiently large constant C_2.

Completing the proof for $p_r < 8$. The above analysis used the assumption $p_r \geq 8$, and to complete the proof, we are left with analyzing $\#SCR(w)$ for $p_r < 8$.

If $4 \leq p_r < 8$, we split $w[\pi_{left}+1..\pi_{right}]$ into blocks of length $\Delta = p_r$ (instead of $\Delta = \lfloor \frac{{p_r}^2}{4} \rfloor$ as in the case $p_r \geq 8$). The terms in (4) are now estimated as follows. $\frac{\log \Delta}{\sqrt{\Delta}} = \frac{\log p_r}{\sqrt{p_r}} \geq 1$ as $\log x \geq \sqrt{x}$ for $4 \leq x < 8$. Since now $k = \lfloor \frac{m}{\Delta} \rfloor \leq \frac{m}{p_r} = e_{mid}$, we have $2p_r(k+2) \leq 2p_r(e_{mid}+2) \leq 4p_r e_{mid} \leq 28 e_{mid} = O(e_{mid})$. We now rewrite (4) into

$$\#SCR(w) \leq C_1 m - C_2 m + \frac{3}{2}C_2 + O(e_{mid}). \tag{14}$$

Substituting (14) into (10), we get

$$C_2 \frac{m(\log n + 1)}{\sqrt{n}} + O(e_{mid}) \leq C_2 m - \frac{3}{2}C_2 \tag{15}$$

We also need to estimate $\frac{m(\log n+1)}{\sqrt{n}}$ differently. We simply note that $\frac{\log n+1}{\sqrt{n}} \leq \frac{5}{6}$ for $n \geq 81$, and then $\frac{m(\log n+1)}{\sqrt{n}} \leq \frac{5}{6}m$. We then have to prove

$$\frac{5}{6}C_2 m + O(e_{mid}) \leq C_2 m - \frac{3}{2}C_2,$$

or, as $m = p_r e_{mid} \geq 4 e_{mid}$,

$$O(e_{mid}) \leq \frac{2}{3}C_2 e_{mid} - \frac{3}{2}C_2.$$

Since $e_{mid} \geq 3$, this inequality can be made true by choosing a sufficiently large constant C_2.

The last case to analyze is $p_r < 4$. A routine check shows that if $p_r = 1, 2$, then $SRC(w) = \emptyset$. If $p_r = 3$, the only case when $SRC(w) \neq \emptyset$ is when r has the form $\ldots aabaab \ldots$. In this case, $\#SCR(w) \leq e_{mid} + 1 = O(e_{mid})$. We have then to prove

$$C_2 \frac{m(\log n + 1)}{\sqrt{n}} + O(e_{mid}) \leq C_1 m.$$

As above, we use $\frac{\log n+1}{\sqrt{n}} \leq \frac{5}{6}$, and we have

$$\frac{5}{6}C_2 m + O(e_{mid}) \leq C_1 m.$$

Since we assumed $C_1 \geq 2C_2$ (see the beginning of the proof), an appropriate choice of C_2 makes hold the above inequality.

The case analysis is exhausted. Choosing a sufficiently large constant C_2 which satisfies all the cases above allows to satisfy inequality (5). The proof of Theorem 4 is completed. Theorem 3 follows.

5 Finding all m-repetitions in a word

In this section we show that Theorem 3 allows to obtain a linear-time algorithm for finding all m-repetitions in a word together with their periods. The algorithm is a modification of Main's algorithm [Mai89] for finding all *leftmost* occurrences of distinct m-repetitions, which is in turn based on the Crochemore's idea of s-factorization [Cro81]. We first describe the Main's algorithm.

Definition 1 ([Cro81, Mai89]) *Let w be an arbitrary word. The* s-factorization *of w is the factorization $w = u_1u_2 \dots u_k$, where u_i's are defined inductively as follows:*

- *If letter a occurring in w immediately after $u_1u_2 \dots u_{i-1}$ does not occur in $u_1u_2 \dots u_{i-1}$, then $u_i = a$.*
- *Otherwise, u_i is the longest word such that $u_1u_2 \dots u_{i-1}u_i$ is a prefix of w and u_i has at least two (possibly overlapping) occurrences in $u_1u_2 \dots u_{i-1}u_i$.*

If $w = u_1u_2 \dots u_k$ is the s-factorization, we call u_i's *s-factors*.

Example 1 *The s-factorization of the word* 1011010110110 *is*

1 0 1 101 01101 10

The usefulness of s-factorization is explained by the following theorem, which is a slight reformulation of Theorem 3.4 from [Mai89].

Theorem 5 *Let $w = u_1u_2 \dots u_k$ be the s-factorization of w, and let r be an m-repetition in w such that $initpos(r) \leq initpos(u_i)$ and $initpos(u_i) \leq endpos(r) < endpos(u_i)$. Then $initpos(u_i) - initpos(r) \leq |u_i| + 2|u_{i-1}|$.*

Theorem 5 suggests a partition of all m-repetitions of w into two classes:

1. m-repetitions r such that $initpos(r) \leq initpos(u_i)$ and $initpos(u_i) \leq endpos(r) < endpos(u_i)$ for some s-factor u_i,
2. m-repetitions r such that $initpos(u_i) < initpos(r) < endpos(r) < endpos(u_i)$ for some s-factor u_i.

The above classification does not cover the m-repetitions r which are suffixes of w (i.e. those for which $endpos(r) = endpos(w)$), but we make this set empty by appending a new symbol \$ at the end of w. This also ensures that the last s-factor u_k consists of one letter. m-repetitions verifying conditions 1 and 2 will be called m-repetitions of type 1 and 2 respectively.

As follows from the definition of s-factorization, an m-repetition of type 2 has another occurrence on the left. Therefore, finding all m-repetitions of type 1 guarantees finding all *distinct* m-repetitions, and in particular all *leftmost* occurrences of distinct m-repetitions.

Let us describe now how m-repetitions of type 1 are found by Main's algorithm. Assume we are given the s-factorization $w = u_1 \dots u_k$. By Theorem 5, we have to find, for each $2 \leq i \leq k$, the m-repetitions in the word t_iu_i, which start in t_i and end in u_i, where t_i is the suffix of $u_1 \dots u_{k-1}$ of length $|u_{i-1}| + 2|u_i|$ (t is the whole word $u_1 \dots u_{i-1}$ in case its length is smaller than $2|u_{i-1}| + |u_i|$). Let us show how to find, in general, all m-repetition in a word tu that start in t and end in u.

Assume that $t = t[1..m]$, $u = u[1..n]$, and we want to find all m-repetitions r in the word $v = tu = v[1..m+n]$ such that $initpos(r) \leq m$ and $endpos(r) \geq m$. Every such repetition belongs (non-exclusively) to one of the two classes: the repetitions which have a period in u and those which have a period in t. Note that by Corollary 1, for every $1 \leq j \leq n$, there is at most one m-repetition of period j starting in t, ending in u, and having a period in u. This shows, in particular, that the number of such m-repetitions is linear in $|u|$. Similarly, the number of such m-repetitions having a period in t is linear in $|t|$, and thus, the number of m-repetitions in $v = tu$ which start in $|t|$ and end in u is linear in $|v|$.

Let us focus on m-repetitions r which have a period in u. The m-repetitions which have a period in t are found symmetrically. We need two auxiliary functions:

- $LP(i),\ 2 \leq i \leq n+1$ defined by $LP(i) = \max\{j | u[1..j] = u[i..i+j-1]\}$ for $2 \leq i \leq n$, and $LP(n+1) = 0$,
- $LS(i),\ 1 \leq i \leq n$ defined by $LS(i) = \max\{j | t[m-j+1..m] = v[m+i-j+1..m+i]\}$.

Informally, $LP(i)$ is the length of the longest prefix of u which is also a prefix of $u[i..n]$, and $LS(i)$ is the length of the longest suffix of t which is also a suffix of $tu[1..i]$. The following theorem holds.

Theorem 6 ([Mai89]) *For $1 \leq j \leq n$, there exists an m-repetition of period j in $v = tu$ which starts in t, ends in u and has a period in u iff $LS(j) + LP(j+1) \geq j$. If the inequality holds, this m-repetition is $v[m - LS(j) + 1..m + j + LP(j+1)]$.*

Function LP can be computed in time linear in $|u|$ and LS in time linear in $|v|$ using the Knuth-Morris-Pratt algorithm (see [Mai89, CR94]). Therefore, all m-repetitions in $v = tu$ which start in t and end in u can be computed in $O(|v|)$ time.

To find all m-repetitions of type 1 in a word w, the Main's algorithm proceeds as follows. First compute the s-factorization $w = u_1 u_2 \ldots u_k$. This computation can be done in time $O(|w|)$ using suffix tree construction [McC76, Ukk95]. Then for each i from 2 to k compute, using the above method, the m-repetitions in word tu, where u is u_i and t is the suffix of $u_1 \ldots u_{i-1}$ of length $2|u_{i-1}| + |u_i|$. Each such computation takes time $O(|u_{i-1}| + |u_i|)$, and therefore finding all m-repetitions of type 1 takes $O(|w|)$ time.

Note that according to the definition of type 1, at each step we need only those repetitions which end strictly before the end of u_i. The reason for this requirement is that if a repetition is a suffix of $u_1 \ldots u_i$, it may not be an m-repetition, as it may extend in w to the right to a longer repetition. On the other hand, if it is an m-repetition, it will be found at the next step of the algorithm, and thus will not be missed. Note also that the algorithm may still output the same m-repetition many times (even unboundedly many times). However, the essential feature is that the algorithm is linear-time and finds all m-repetitions of type 1.

To find *all* m-repetitions, we have to find, in addition, all m-repetitions of type 2. We now show how it can be done. The task is greatly simplified by the fact that every m-repetition of type 2 occurs entirely inside some s-factor u_i of the s-factorization, and each u_i has an earlier occurrence in w.

During the computation of the s-factorization we store, for each s-factor u_i, a pointer to an earlier occurrence of u_i in w. This can be easily done using the suffix tree construction, so that the computation of s-factorization remains linear-time. Let v_i be this earlier occurrence of u_i, and let $\Delta_i = initpos(u_i) - initpos(v_i)$. Obviously, each m-repetition of type 2 occurring inside u_i is a copy of an m-repetition occurring inside v_i shifted by Δ_i to the right.

We now proceed as follows. First, we compute all m-repetitions of type 1 with the Main's algorithm. Then we sort them, using basket sort, into n lists, such that list j contains the m-repetitions with end position j. (Note that during the sort we can eliminate the duplicates.) Then we process all the lists in the increasing order and sort the m-repetitions again, using basket sort, into n lists according to their initial position. It should be clear that after this double sort, the m-repetitions with the same initial position j are sorted inside the list j in the increasing order of their end positions. As there is a linear number of m-repetitions of type 1, both sort procedures take a linear time.

Now we find the m-repetitions of type 2. We will store them in the same data structure. For each u_i, $i = 1..k$, and for each internal position j inside u_i, we have to find the m-repetitions of w starting at this position and ending inside u_i. We then have to find the m-repetitions starting at

position $j - \Delta_i$ in v_i which end inside v_i, and then shift them by Δ_i to the right. Note that these repetitions may be either of type 1, or previously found repetitions of type 2. We look through the list $j - \Delta_i$ and retrieve its prefix consisting of those m-repetitions which end inside v_i. Then we shift each of these m-repetitions by Δ_i and append a modified copy of this prefix to the head of the list j. Note that the data structure is preserved, as all appended m-repetitions have their end position inside u_i, and those which have been previously stored in the list j are of type 1 and then have their end position outside u_i. Since we process u_i's from left to right, no m-repetition can be missed. Thus, we recover all m-repetitions of type 2 and after all u_i's have been processed, the data structure contains all m-repetitions of both types.

Note that when we retrieve a prefix of the list corresponding to some position in v_i, each repetition in this prefix results in a new m-repetition of type 2 in u_i. This shows that the time spent to processing the lists is proportional to the number of newly found m-repetitions. Theorem 3 from the previous section states that the number of all m-repetitions is linear in the length of the word. This proves that the whole algorithm takes linear time.

To conclude, we note that the algorithm finds the m-repetitions together with their periods, which allows to easily derive other repetitions, such as primitively- or non-primitively-rooted squares, cubes, etc. For example, as a side result, we obtain algorithms for finding all squares, either primitively- or non-primitively-rooted, in time $O(n + T)$, where T is the number of corresponding squares (see Introduction for the comparison of this bound with other works).

6 Concluding remarks

The proof of Theorem 3 in Section 4 is quite technical. Its main drawback is that it does not allow, in practice, to extract a constant factor of the linear bound because of the technical assumption $n \geq 81$. It remains an open question if a simpler proof can be found which would imply a constant factor. We conjecture that for the binary alphabet this constant factor is equal to 1. Table 1 gives the maximal number of m-repetitions for the binary alphabet for low values of n, along with an example of a word realizing this number.

Concerning counting results (Section 3), we note that Fibonacci words don't realize the maximal number of m-repetitions among the binary words. For example, for length 21 this number is 15 (see Table 1), while in the Fibonacci word f_7 of length 21 the number of m-repetitions is 13.

While the number of m-repetitions in Fibonacci words is one less than the number of distinct squares, the maximal number of m-repetitions in binary words of length n is apparently slightly bigger than the maximal number of distinct squares. In Table 2, these numbers are compared for a few small values n (the number of distinct squares is borrowed from [FS98b]).

In spite of this closeness between the number of m-repetitions and that of distinct squares, there is no apparent connection between them. It is possible to conceive words with a big number of m-repetitions and small number of distinct squares. For example, the result of [FS95] implies that there exist words with only three distinct squares but with unbounded number of m-repetitions. Still, we are wondering if the fact that the number of m-repetitions in Fibonacci words is one less than the number of distinct squares is a simple coincidence or it has some combinatorial explanation.

Another interesting phenomenon is that in words that realize the maximal number of m-repetitions, there are no repetitions of big exponent. According to computer experiments, m-repetitions present in these words are of exponent at most 3. Here again, Fibonacci words reflect a typical situation, since all repetitions they contain are of exponent smaller than 4.

The latter remark is somewhat related to the hypothesis suggested in [SG98] about the linearity of the maximal number of "branching tandem repeats" in a word. Branching tandem repeats are

n	max # of m-repetitions	example
5	2	00011
6	3	001001
7	4	0010011
8	5	00110011
9	5	000110011
10	6	0010011001
11	7	00100110011
12	8	001001100100
13	8	0001001100100
14	10	00100110010011
15	10	000100110010011
16	11	0010011001001100
17	12	00100101101001011
18	13	001001100100110011
19	14	0010011001001100100
20	15	00101001011010010100
21	15	00010100101101001010 0
22	16	0010010100101101001011
23	17	00100101001011010010100
24	18	001001100100110110011011
25	19	0010011001000100110010011
26	20	00101001011010010100101101
27	21	001001010010110100101001011
28	22	0010100101101001010010110100
29	23	00101001011010010100101101011
30	24	001011010010110101101001011010
31	25	0010100101101001010010110100101

Table 1: Maximal number of m-repetitions for small n over binary alphabet

squares uu (not necessarily primitively-rooted) which are not followed by the first letter of u. To relate this to m-repetitions, branching tandem repeats are suffixes of the m-repetitions of length $2kp(r)$, where r is the corresponding m-repetition and $k \geq 1$. The linearity of the maximal number of branching tandem repeats would imply our Theorem 3, as there are at least as many branching tandem repeats as m-repetitions (each branching tandem repeat corresponds to an m-repetition but one m-repetition may contain several branching tandem repeats). A general result which would imply both our Theorem 3 and the conjecture of [SG98] would be to show that the sum of the exponents of all m-repetitions in a word is linearly bounded.

References

[AP83] A. Apostolico and F.P. Preparata. Optimal off-line detection of repetitions in a string. *Theoretical Computer Science*, 22(3):297–315, 1983.

n	max nb of max repetitions	max nb of distinct squares
3	1	1
4	2	2
5	2	2
6	3	3
7	4	3
8	5	4
9	5	5
10	6	6
11	7	7
12	8	7
13	8	8

Table 2: Comparison of maximal number of m-repetitions and maximal number of distinct squares

[CR94] M. Crochemore and W. Rytter. *Text algorithms.* Oxford University Press, 1994.

[CR95] M. Crochemore and W. Rytter. Squares, cubes, and time-space efficient string searching. *Algorithmica*, 13:405–425, 1995.

[Cro81] Maxime Crochemore. An optimal algorithm for computing the repetitions in a word. *Information Processing Letters*, 12:244–250, 1981.

[Cro83] M. Crochemore. Recherche linéaire d'un carré dans un mot. *Comptes Rendus Acad. Sci. Paris Sér. I Math.*, 296:781–784, 1983.

[CS96] J.D. Currie and R.O. Shelton. Cantor sets and Dejean's conjecture. *Journal of Automata, Languages and Combinatorics*, 1(2):113–128, 1996.

[FS95] A.S. Fraenkel and J. Simpson. How many squares must a binary sequence contain? *Electronic Journal of Combinatorics*, 2(R2):9pp, 1995. `http://www.combinatorics.org/Journal/journalhome.html`.

[FS98a] A.S. Fraenkel and J. Simpson. The exact number of squares in Fibonacci words. *Submitted for publication*, May 1998.

[FS98b] A.S. Fraenkel and J. Simpson. How many squares can a string contain? *J. Combinatorial Theory (Ser. A)*, 82:112–120, 1998.

[IMS97] C.S. Iliopoulos, D. Moore, and W.F. Smyth. A characterization of the squares in a Fibonacci string. *Theoretical Computer Scince*, 172:281–291, 1997.

[Kos94] S. R. Kosaraju. Computation of squares in string. In M. Crochemore and D. Gusfield, editors, *Proceedings of the 5th Annual Symposium on Combinatorial Pattern Matching*, number 807 in Lecture Notes in Computer Science, pages 146–150. Springer Verlag, 1994.

[Lot83] M. Lothaire. *Combinatorics on Words*, volume 17 of *Encyclopedia of Mathematics and Its Applications.* Addison Wesley, 1983.

[Mai89] M. G. Main. Detecting leftmost maximal periodicities. *Discrete Applied Mathematics*, 25:145–153, 1989.

[McC76] E. M. McCreight. A space-economical suffix tree construction algorithm. *Journal of the ACM*, 23(2):262–272, 1976.

[ML84] M.G. Main and R.J. Lorentz. An $O(n \log n)$ algorithm for finding all repetitions in a string. *Journal of Algorithms*, 5(3):422–432, 1984.

[ML85] M.G. Main and R.J. Lorentz. Linear time recognition of square free strings. In A. Apostolico and Z. Galil, editors, *Combinatorial Algorithms on Words*, volume 12 of *NATO Advanced Science Institutes, Series F*, pages 272–278. Springer Verlag, 1985.

[MP92] F. Mignosi and G. Pirillo. Repetitions in the Fibonacci infinite word. *RAIRO Theoretical Informatics and Applications*, 26(3):199–204, 1992.

[MRS95] F. Mignosi, A. Restivo, and S. Salemi. A periodicity theorem on words and applications. In *Proceedings of the 20th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, volume 969 of *Lecture Notes in Computer Science*, pages 337–348. Springer Verlag, 1995.

[Séé85] P. Séébold. Propriétés combinatoires des mots infinis engendrés par certains morphismes. Rapport 85-16, LITP, Paris, 1985.

[SG98] J. Stoye and D. Gusfield. Simple and flexible detection of contiguous repates using a suffix tree. In M. Farach-Colton, editor, *Proceedings of the 9th Annual Symposium on Combinatorial Pattern Matching*, number 1448 in Lecture Notes in Computer Science, pages 140–152. Springer Verlag, 1998.

[Sli83] A.O. Slisenko. Detection of periodicities and string matching in real time. *Journal of Soviet Mathematics*, 22:1316–1386, 1983.

[Ukk95] E. Ukkonen. On-line construction of suffix trees. *Algorithmica*, 14(3):249–260, 1995.

Appendix A

Here we prove Lemma 4 asserting that for $n \geq 81$ and $m, m', m'' \geq 1$ such that $m + m' + m'' = n$, we have

$$\sqrt{n} \log n - \sqrt{m'} \log m' - \sqrt{m''} \log m'' \leq \frac{m(\log n + 1)}{\sqrt{n}}. \tag{16}$$

Without loss of generality assume that $m' \geq m''$. Consider the function $f(x) = \sqrt{x} \log x$. $f(x)$ is positive ($f(x) \geq 0$) increasing ($f'(x) > 0$) and concave ($f''(x) < 0$) for $x \geq 1$. Then $f(m'') \geq f(n) - f(n - m'')$ and therefore $f(n) - f(m') - f(m'') \leq f(n - m'') - f(m')$. This difference is maximal when m' is minimal, that is $m' = m'' = \frac{n-m}{2}$. We than have

$$f(n) - f(m') - f(m'') \leq f(\frac{n}{2} + \frac{m}{2}) - f(\frac{n}{2} - \frac{m}{2}) \tag{17}$$

The right-hand side of (17) is equal to

$$\frac{f^2(\frac{n}{2} + \frac{m}{2}) - f^2(\frac{n}{2} - \frac{m}{2})}{f(\frac{n}{2} + \frac{m}{2}) + f(\frac{n}{2} - \frac{m}{2})}. \tag{18}$$

Since $f(x)$ is concave for $x \geq 1$, this implies that for sufficiently large x, y (actually, for $x, y \geq e^2$) we have $f(x) + f(y) \geq f(x + y)$. Then, the denominator of (18) can be estimated from below as $f(\frac{n}{2} + \frac{m}{2}) + f(\frac{n}{2} - \frac{m}{2}) \geq f(n) = \sqrt{n} \log n$. The numerator of (18) is equal to

$$\int_{\frac{n}{2}-\frac{m}{2}}^{\frac{n}{2}+\frac{m}{2}} g(x)\, dx, \tag{19}$$

where $g(x) = (f^2(x))' = \log^2(x) + 2 \log e \log x$. It is easy to check that $g(x)$ is also positive, increasing and concave for $x \geq 1$. Then the estimation $\int_a^b g(x)\, dx \leq (b-a) g(\frac{b+a}{2})$ holds, and above integral can be estimated by $mg(\frac{n}{2}) = m(\log^2(\frac{n}{2}) + 2 \log e \log \frac{n}{2}) \leq m(\log^2 n + \log n)$. Substituting into (18) the estimates for the numerator and denominator, inequation (16) follows.