Journal of Computer and System Sciences 82 (2016) 477–487

Contents lists available at ScienceDirect

Journal of Computer and System Sciences

www.elsevier.com/locate/jcss

Computing maximal-exponent factors in an overlap-free word ☆

Golnaz Badkobeh [a], Maxime Crochemore [a,b,*]

[a] King's College London, UK
[b] Université Paris-Est, France

ARTICLE INFO

Article history:
Received 6 November 2013
Received in revised form 26 August 2014
Accepted 27 August 2014
Available online 2 December 2015

Keywords:
Word
String
Repetition
Power
Repeat
Periodicity
Word exponent
Return word
Algorithm
Automaton

ABSTRACT

The exponent of a word is the quotient of its length over its smallest period. The exponent and the period of a word can be computed in time proportional to the word length. We design an algorithm to compute the maximal exponent of all factors of an overlap-free word. Our algorithm runs in linear-time on a fixed-size alphabet, while a naive solution of the question would run in cubic time. The solution for non-overlap-free words derives from algorithms to compute all maximal repetitions, also called runs, occurring in the word.
We also show there is a linear number of occurrences of maximal-exponent factors in an overlap-free word. Their maximal number lies between $0.66n$ and $2.25n$ in a word of length n. The algorithm can additionally locate all of them in linear time.

1. Introduction

We consider the question of computing the maximal exponent of factors (substrings) of a given word (string). The exponent of a word is the quotient of the word length over the word smallest period. For example `alfalfa` has period 3 and exponent 7/3, and `restore` has period 5 and exponent 7/5. A word with exponent e is also called an e-power. The exponent indicates better than the period the degree of repetitiveness of factors occurring in a word.

In this article we focus on factors whose exponent is at most 2. Such factors can uniquely be written as uvu where u is the longest border of uvu, that is, the longest proper prefix that is also a suffix of the factor. Note that the exponent is 1 if and only if u is the empty word, while it is 2 if and only if v is the empty word. Consistently with the existing literature a word whose exponent is 1, the minimal possible exponent, admits only the empty word as a border and is called border-free. A word is called a square when its exponent is a positive even integer. In this article, a factor whose exponent is smaller than 2 is called a repeat, while a factor whose exponent is at least 2 is called a repetition or a periodic factor. In other words, in the former case the factor u repeats at two distant positions.

☆ An extended abstract of a preliminary version of the article was presented at SPIRE'2012 [1].

* Corresponding author at: King's College London, UK.
E-mail addresses: golnaz.badkobeh@googlemail.com (G. Badkobeh), maxime.crochemore@kcl.ac.uk (M. Crochemore).
URLs: http://www.inf.kcl.ac.uk/pg/badkobeh/ (G. Badkobeh), http://www.inf.kcl.ac.uk/staff/mac/ (M. Crochemore).

http://dx.doi.org/10.1016/j.jcss.2015.11.007
0022-0000/

The study of repeats in a word is relevant to long-distance interactions between separated occurrences of the same segment (the u part) in the word. Although occurrences may be far away from each other, they may interact when, for example, the word is folded as it is the case for genomic sequences. A very close problem to considering those repeats is that of computing maximal pairs (positions of the two occurrences of u) with gaps constraints as described by Gusfield [2] and later improved by Brodal et al. [3].

From a combinatorial point of view, the question is related to return words: z is a return word associated with u if u is a prefix of zu and u has no internal occurrence in zu. For instance, if u has only two occurrences in uvu (as a prefix and a suffix) then uv is a return word for u. The word then links two consecutive occurrences of u. The analysis of return words provide characterisations for word morphisms and infinite words. For example, a binary infinite Sturmian word, generalisation of Fibonacci word, is characterised by the fact that every factor (occurring infinitely many times) admits exactly two return words (see [4] and references therein).

The notion of maximal exponent is central to questions related to the avoidability of powers in infinite words. An infinite word is said to avoid e-powers (resp. e^+-powers) if the exponents of its finite factors are smaller than e (resp. no more than e). Dejean [5] introduced the repetitive threshold $\mathrm{RT}(a)$ of an a-letter alphabet: the smallest rational number for which there exists an infinite word on a letters whose finite factors have exponent at most $\mathrm{RT}(a)$. In other words, the maximal exponent of factors of such a word is $\mathrm{RT}(a)$, the minimum possible. The word is also said to be $\mathrm{RT}(a)^+$-power free. It is known from Thue [6] that $r(2) = 2$, Dejean [5] proved that $r(3) = 7/4$ and stated the exact values of $\mathrm{RT}(a)$ for every alphabet size $a > 3$. Dejean's conjecture was eventually proved in 2009 after partial proofs given by several authors (see [7,8] and references therein).

The exponent of a word can be calculated in linear time using basic string matching that computes the smallest period associated with the longest border of the word (see [9]). A straightforward consequence provides a $O(n^3)$-time solution to compute the maximal exponent of all factors of a word of length n since there are potentially of the order of n^2 factors. However, a quadratic time solution is also a simple application of basic string matching. By contrast, our solution runs in linear time on a fixed-size alphabet.

When a word contains runs, that is, maximal periodicities of exponent at least 2, computing their maximal exponent can be achieved in linear time by adapting the algorithm of Kolpakov and Kucherov [10] that computes all the runs occurring in the word. Their result relies on the fact that there exists a linear number of runs in a word [10] (see [11,12] for precise bounds). Nevertheless, this does not apply to square-free words, which we are considering here.

Our solution works indeed on overlap-free words, not only on square-free words, that is, on words whose maximal exponent of factors is at most 2. Thus, we are looking for factors w of the form uvu, where u is the longest border of w. In order to accomplish this goal, we exploit two main tools: the Suffix Automaton of some factors and a specific factorisation of the whole word.

The Suffix Automaton (see [9]) is used to search for maximal-exponent factors in a product of two words due to its ability to locate occurrences of all factors of a pattern. Here, we enhance the automaton to report the right-most occurrences of those factors. Exploiting only the Suffix Automaton in a balanced divide-and-conquer manner produces a $O(n \log n)$-time algorithm.

In order to eliminate the log factor we additionally exploit a word factorisation, namely the f-factorisation (see [9]), a type of LZ77 factorisation (see [13]) fit for word algorithms. It has now become common to use this factorisation to derive efficient or even optimal algorithms. The f-factorisation allows one to skip larger and larger parts of the words during an online processing. For our purpose, it is composed of factors occurring before their current position with no overlap. The factorisation can be computed in $O(n \log a)$-time (where a is the alphabet size) using a Suffix Tree or a Suffix Automaton, and in linear time on an integer alphabet using a Suffix Array [14].

The running time of the proposed algorithm depends additionally on the repetitive threshold of the underlying alphabet size of the word. The threshold restricts the context of the search for a second occurrence of u associated with a factor uvu.

We show a very surprising property of factors whose exponent is maximal in an overlap-free word: there are no more than a linear number of occurrences of them, although the number of occurrences of maximal (i.e. non-extensible) factors can be quadratic.

We show a lower bound of $0.66n$ and an upper bound of $2.25n$ on their maximal number for a word of length n. They improve on the bounds given in a preliminary version [1] of the article. The lower bound is based on a result of Pansiot [15] on the repetitive threshold of four-letter alphabets.

As a consequence, the algorithm can be modified to output all occurrences of maximal-exponent factors of an overlap-free word in linear time.

The question would have a simple solution by computing MinGap on each internal node of the Suffix Tree of the input word, as is discussed in the conclusion. MinGap of a node is the smallest difference between the positions assigned to leaves of the subtree rooted at the node. Unfortunately, the best algorithms for MinGap computation, equivalent to MaxGap computation, run in time $O(n \log n)$ (see [16–18] and the discussion in [19]).

A remaining question to the present study is to unify the algorithmic approaches for locating runs in non-overlap-free words and maximal-exponent factors in overlap-free words.

The plan of the article is as follows. After defining the problem in the next section we present the general scheme of the algorithm that relies on the f-factorisation of the input word in Section 3. The sub-function operating a Suffix Automaton is