Journal of Computer and System Sciences 80 (2014) 1254–1265

Contents lists available at ScienceDirect

Journal of Computer and System Sciences

www.elsevier.com/locate/jcss

# Prefix–suffix duplication

Jesús García López [a], Florin Manea [b,c], Victor Mitrana [d,*]

[a] Department of Applied Mathematics, University School of Informatics, Polytechnic University of Madrid, Crta. de Valencia km. 7, 28031 Madrid, Spain
[b] Department of Computer Science, Christian-Albrechts University of Kiel, Christian-Albrechts-Platz 4, 24118 Kiel, Germany
[c] Faculty of Mathematics and Computer Science, University of Bucharest, Str. Academiei 14, RO-010014 Bucharest, Romania
[d] Department of Organization and Structure of Information, University School of Informatics, Polytechnic University of Madrid, Crta. de Valencia km. 7, 28031 Madrid, Spain

ARTICLE INFO

*Article history:*
Received 29 October 2012
Received in revised form 10 November 2013
Accepted 11 February 2014
Available online 19 February 2014

*Keywords:*
Bio-inspired operations
Prefix–suffix duplication
Language theoretical properties
Prefix–suffix duplication distance
Algorithms on words

ABSTRACT

We consider a bio-inspired formal operation on words called prefix–suffix duplication which consists in the duplication of a prefix or suffix of a given word. The class of languages defined by the iterated application of the prefix–suffix duplication to a word is considered. We show that such a language is context-free if and only if the initial word contains just one letter. Moreover, every language in this class is semilinear and belongs to **NL**. We propose a $\mathcal{O}(n^2 \log n)$ time and $\mathcal{O}(n^2)$ space recognition algorithm. Two algorithms are further proposed for computing the prefix–suffix duplication distance between two words, defined as the minimal number of prefix–suffix duplications applied to one of them in order to get the other one. The first algorithm runs in cubic time and uses quadratic space while the second one is more efficient, having $\mathcal{O}(n^2 \log n)$ time complexity, but needs $\mathcal{O}(n^2 \log n)$ space.



## 1. Introduction

One of the most frequent and less understood mutations among the genome rearrangements is the duplication of a segment of a chromosome [15]. In the process of duplication, a stretch of DNA is duplicated yielding two or more adjacent copies, called also tandem repeats. It is commonly asserted that approximately 5% of the genome is involved in duplications and the distribution of these tandem repeats varies widely along the chromosomes [20]. An interesting property of tandem repeats is to make possible a so-called "phylogenetic analysis" which might be useful in the investigation of the evolution of species by determining the most likely duplication history [22]. The detection of these tandem repeats and algorithms for tandem repeats reconstructing history have received a great deal of attention in bioinformatics [1,2,19].

However, a special type of duplications, known as telomeres, appear only at the ends of chromosomes. Generally, telomeres consist of tandem repeats of a small number of nucleotides, specified by the action of telomerase. They are considered to be protective DNA-protein complexes found at the end of eukaryotic chromosomes which stabilize the linear chromosomal DNA molecule [4,16]. The length of telomeric DNA is important for the chromosome stability: the loss of telomeric repeat sequences may result in chromosome fusion and lead to chromosome instability [12]. In [20] one states that it is a further challenge the sequencing of the 20% of the genome that is formed by repetitive heterochromatin which is implicated in the process of chromosome replication and maintenance.

* Corresponding author.
*E-mail addresses:* jglopez@eui.upm.es (J. García López), flm@informatik.uni-kiel.de (F. Manea), victor.mitrana@upm.es (V. Mitrana).

http://dx.doi.org/10.1016/j.jcss.2014.02.011
0022-0000/

Treating chromosomes and genomes as languages raises the possibility that the structural information contained in biological sequences can be generalized and investigated by formal language theory methods [18]. Thus, the interpretation of duplication as a formal operation on words has inspired several works in the area of Formal Languages opened by [6,21] and continued in a series of papers, see, e.g., [11] and the references therein. In the first part of this paper we follow a similar approach to that from [6,21,10]. We consider only duplications that may appear in the ends of the words only, called prefix–suffix duplications, similar to the case of telomeric DNA. In this context, we investigate the class of languages that can be defined by the iteratively application of the prefix–suffix duplication to a word and try to compare it to other well studied classes of languages. To this end, we show that the languages of this class have a rather complicated structure even if the initial word is rather simple; more precisely, they are already non-context-free as soon as the initial word contains at least two different letters. Consequently, one can derive a trivial algorithm deciding in linear time whether the prefix–suffix duplication language defined by a given word is context-free (or equivalently, in the case of unary languages, regular) just by counting the different letters that occur in the given word. Naturally, we also investigate how complicated these languages actually are, or, formally, try to derive upper bounds for the class of prefix–suffix duplication languages. We show that all the languages of this class have a linear Parikh image and belong to **NL**, hence are polynomially recognizable. Starting from this result, we focus on the computational complexity of solving problems related to such languages. First, we are interested in finding an efficient algorithm solving the membership problem for such languages. To this aim, in the second part of the paper, we propose a $\mathcal{O}(n^2 \log n)$ time and $\mathcal{O}(n^2)$ space recognition algorithm. Then, we consider the prefix–suffix duplication distance between two given words, defined as the minimal number of prefix–suffix duplications applied to one word in order to get the other, and develop efficient algorithmic solutions to compute it. We propose two algorithms: a cubic time one which uses a quadratic memory and a more efficient one, namely $\mathcal{O}(n^2 \log n)$ time complexity, but with some extra memory consumption, that is $\mathcal{O}(n^2 \log n)$ space complexity. It is worth mentioning that the efficiency of the algorithms we present follows from the application of a series of non-trivial combinatorics on words remarks as well as the usage of several data-structures, specific to stringology.

One should note that the investigation we pursue here is not aimed to tackle real biological solutions. In fact, its aim is to provide a better understanding of the structural properties of strings obtained by prefix–suffix duplication as well as specific tools for the manipulation of such strings. On the long run, such tools could provide the foundations on which applications working with real data are built.

## 2. Preliminaries

We assume the reader to be familiar with fundamental concepts from Formal Language Theory, such as the classes of the Chomsky hierarchy, finite automaton, generalized sequential machine (gsm), which can be found in many textbooks, e.g., the handbook [17]. The same assumption concerns fundamental concepts from Complexity Theory such as Turing machine, random access machine (RAM) with logarithmic word size and standard unit-cost operations, time and space complexity classes; see, for instance, [14].

We start by summarizing the notions used throughout this work. An *alphabet* is a finite and nonempty set of symbols. The cardinality of a finite set $A$ is written $card(A)$. Any finite sequence of symbols from an alphabet $V$ is called *word* over $V$. The set of all words over $V$ is denoted by $V^*$ and the empty word is denoted by $\varepsilon$; we further let $V^+ = V^* \setminus \{\varepsilon\}$. Given a word $w$ over an alphabet $V$, we denote by $|w|$ its length, while $|w|_a$ denotes the number of occurrences of the letter $a$ in $w$. Furthermore, $alph(w)$ denotes the minimal alphabet $W$ such that $w \in W^*$, i.e. $alph(w) = \{a \in V \mid |w|_a \neq 0\}$. If $w = xyz$ for some $x, y, z \in V^*$, then $x$, $y$, $z$ are called prefix, subword, suffix, respectively, of $w$. For a word $w$, $w[i..j]$ denotes the subword of $w$ starting at position $i$ and ending at position $j$, $1 \leqslant i \leqslant j \leqslant |w|$; by convention, $w[i..j] = \varepsilon$ if $i > j$. If $i = j$, then $w[i..j]$ is the $i$-th letter of $w$ which is simply denoted by $w[i]$. A *period* of a word $w$ over $V$ is a positive integer $p$ such that $w[i] = w[j]$ for all $i$ and $j$ with $i \equiv j \pmod p$. By $per(w)$ (called *the period of* $w$) we denote the smallest period of $w$.

We say that the pair ${}_w(i, p)$ is a *duplication* (*repetition*) in $w$ starting at position $i$ in $w$ if $w[i..i+p-1] = w[i+p..i+2p-1]$. Analogously, the pair $(i, p)_w$ is a duplication in $w$ ending at position $i$ in $w$ if $w[i-2p+1..i-p] = w[i-p+1..i]$. In both cases, $p$ is called the length of the duplication.

Given a word $x$ over an alphabet $V$, we consider the following duplication operations:

- *Prefix duplication*, namely $PD(x) = \{ux \mid x = uy \text{ for some } u \in V^+\}$. The *suffix duplication* is defined analogously, that is $SD(x) = \{xu \mid x = yu \text{ for some } u \in V^+\}$.
- *Prefix–suffix duplication*, namely $PSD(x) = PD(x) \cup SD(x)$.

The prefix–suffix duplication is naturally extended to languages $L$ by $PSD(L) = \bigcup_{x \in L} PSD(x)$. We further define:

$$PSD^0(x) = \{x\},$$
$$PSD^{k+1}(x) = PSD^k(x) \cup PSD\big(PSD^k(x)\big), \quad \text{for any } k \geqslant 0,$$
$$PSD^*(x) = \bigcup_{k \geqslant 0} PSD^k(x).$$