Theoretical Computer Science ●●● (●●●●) ●●●–●●●

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

Prefix–suffix square reduction

Paolo Bottoni [a], Anna Labella [a], Victor Mitrana [b,*,1]

[a] Department of Computer Science, "Sapienza" University of Rome, Via Salaria 113, 00198 Rome, Italy
[b] Faculty of Mathematics and Computer Science, University of Bucharest, Str. Academiei 14, 010014, Bucharest, Romania

ARTICLE INFO

Article history:
Received 19 July 2016
Received in revised form 28 October 2016
Accepted 2 December 2016
Available online xxxx

Keywords:
Duplication
Prefix–suffix square reduction
Primitive square reduction root
Formal languages

ABSTRACT

In this work we introduce the operation of prefix–suffix square reduction as the inverse of the prefix–suffix duplication studied in the literature. This operation reduces all possible squares that are prefixes or suffixes of a word to one half of these squares. Two variants are considered, depending on the unbounded and bounded length of the removed prefix or suffix. We investigate the complexity of the (non-uniform) membership problem for the prefix–suffix square reduction of a given language and the closure properties of some classes of languages under these operations as well as under their iterated variants. Afterwards, we define the primitive prefix–suffix square root of a word w as a word x that can be obtained from w by iterated prefix–suffix square reductions and it is irreducible in turn, i.e., no further prefix or suffix square reduction can be applied. We prove that the language of primitive prefix–suffix square roots of all words over an alphabet is never regular for alphabets with at least two symbols in the unbounded case, and always regular in the bounded case. The paper ends with a brief discussion on some open problems and some algorithmic aspects.

1. Introduction

One of the most frequent and vividly investigated phenomena in different contexts is that of repetitive structures. In genetics, for instance, one of the less understood mutations among the genome rearrangements is the duplication of a segment of a chromosome [18]. In the process of duplication, a stretch of DNA is duplicated, yielding two or more adjacent copies, also called 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 [23]. An interesting property of tandem repeats is to make a so-called "phylogenetic analysis" possible, which might be useful in the investigation of the evolution of species by determining the most likely duplication history [25]. The detection of these tandem repeats, as well as algorithms for tandem repeats reconstructing history, have received a great deal of attention in bioinformatics [2,3,22]. Thus, duplicating factors and reducing squares to one of their halves is an interesting algorithmic problem with some motivation from bioinformatics. Reductions of squares seems to be of importance in data compression, where the compressed words have to contain some information that allows the reconstruction of the original word [10,11].

Treating chromosomes and genomes as languages opens the possibility that the structural information contained in biological sequences can be generalized and investigated by formal language theory methods [21]. Thus, the interpretation

[*] Corresponding author.
E-mail addresses: bottoni@di.uniroma1.it (P. Bottoni), labella@di.uniroma1.it (A. Labella), mitrana@fmi.unibuc.ro (V. Mitrana).
[1] Research supported by the Visiting Professor Program - "Sapienza" University of Rome.

http://dx.doi.org/10.1016/j.tcs.2016.12.005
0304-3975/© 2016 Elsevier B.V. All rights reserved.

of duplication as a formal operation on words has inspired several works in the area of Formal Languages, opened by [5,24] and continued in a series of papers, see, e.g., [12,13] and the references therein. A special type of duplications inspired by the tandem repeats, known as telomeres, which appear only at the end of chromosomes, has been considered in [9]. They are considered to be protective DNA-protein complexes found at the end of eukaryotic chromosomes which stabilize the linear chromosomal DNA molecule [4,19]. 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 [16]. Thus, in [9], one considers duplications that may only appear at beginning and end of the words only, called prefix–suffix duplications. In this context, the aforementioned work investigates the class of languages that can be defined by the iterative application of the prefix–suffix duplication to a word and tries to compare it to other well studied classes of languages. Starting from the biochemical reality that inspired the definition of this operation in [9], namely that the telomeres cannot be arbitrarily long, a restricted variant of duplication, called bounded duplication, was introduced in [7]. In this variant, the length of the prefix or suffix that is duplicated is bounded by a predefined constant. In both papers algorithms for computing different measures on these operations for words and languages are presented.

On the other hand, the inverse operation, namely reducing repetitions, is very natural. Returning to our source of inspiration, for computing a phylogenetic network (a set of evolutionary relationships between different genes, chromosomes, genomes, or even species) it is necessary to detect all squares and compute all possible direct predecessors. But this is just one step; if we want to compute other possible predecessors, not necessarily direct ones, this process must be iterated. We close these considerations by stressing that the investigation we pursue here is not aimed to tackle real biological solutions. The biological phenomenon is just a source of inspiration for our approach; our approach uses the biological concepts at a simplified level and only from the point of view of theoretical computer science. Its aim is actually to provide a better understanding of the structural properties of strings obtained by prefix–suffix square reductions. On the long run, such tools might provide the foundations on which applications working with real data are built.

In this paper, we follow the line opened in [14,15] for the operations considered in [9] and [7]. We define the unbounded prefix–suffix square reduction, which is the inverse of the duplication defined in [9] and the bounded prefix–suffix square reduction, which is the inverse of the duplication defined in [6,7]. Our operation can be informally defined as the process of reducing a square (tandem repeat) to one of its halves, provided that the square is either a prefix or a suffix of the current word. We consider both variants: bounded and unbounded as well as, for each of them, the non-iterated and the iterated cases.

The paper is organized as follows. In Section 2, we recall all the concepts and notations we need and give the formal definitions for the unbounded and bounded prefix–suffix square reductions. Then, in Section 3, we investigate the non-iterated version of these operations. We show that, in general, the membership for the unbounded prefix–suffix square reduction of a language L can be decided in $\mathcal{O}(nf(n))$ time, provided that the membership for L can be decided in $\mathcal{O}(f(n))$ time. The factor n is just the bound in the case of bounded prefix–suffix square reduction. This factor is not necessary for regular languages. Then we show that the space complexity of the membership problem for both unbounded and bounded prefix–suffix square reduction of a language remains the same to that of the given language. As far as closure properties are concerned, the class of regular languages is closed under unbounded and bounded prefix–suffix square reduction, while there are linear languages such that their unbounded prefix–suffix square reduction is not even context-free. The iterated versions of the bounded and unbounded prefix–suffix square reductions are then considered in Section 4. We show that the class of regular languages is still closed under iterated bounded prefix–suffix square reduction, but the unbounded case remains open. We also show that there are linear languages such that their iterated unbounded prefix–suffix square reduction is not context-free, while the closure of the classes of linear and context-free languages under iterated bounded prefix–suffix square reduction remains open. Afterwards, we define the primitive prefix–suffix square root of a word w as a word x that can be obtained from w by iterated prefix–suffix square reductions and is irreducible, i.e., no further prefix or suffix square reduction can be applied. The primitive prefix–suffix square root of a language contains all the primitive prefix–suffix square roots of its words. The language of primitive prefix–suffix square roots over a given alphabet V is the primitive prefix–suffix square root of V^*. We prove that this language is never regular for alphabets with at least two symbols in the unbounded case, and always regular in the bounded case. The papers ends with Section 5, containing a brief discussion on some open problems and some algorithmic aspects.

2. Preliminaries

We assume the reader to be familiar with fundamental concepts of formal language theory and complexity theory (see, e.g., [20] and [17], respectively). 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 a *word* over V. The set of all words over V is denoted by V^* and the empty word is denoted by ε; 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 symbol $a \in V$ in w. Furthermore, $alph(w)$ denotes the minimal alphabet $W \subseteq V$ such that $w \in W^*$, i.e. $alph(w) = \{a \in V \mid |w|_a \neq 0\}$. Obviously, $alph(L) = \bigcup_{x \in L} alph(x)$. 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