Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

Order preserving pattern matching revisited ☆

Md. Mahbubul Hasan [a,1], A.S.M. Shohidull Islam [a,b], Mohammad Saifur Rahman [a], M. Sohel Rahman [a,*]

[a] AℓEDA Group, Department of CSE, BUET, Dhaka 1000, Bangladesh
[b] Department of Computational Engineering and Science, McMaster University, Hamilton, Ontario, Canada

ARTICLE INFO

Article history:
Received 18 March 2014
Available online 11 December 2014

Keywords:
Algorithms
Pattern matching
Document processing
Sequence analysis

ABSTRACT

In this paper, we study the order preserving pattern matching (OPPM) problem, which is a very recent variant of the classic pattern matching problem. We revisit this variant, present a new interesting pattern matching algorithm and for the first time consider string regularities from this new perspective.

© 2014 Published by Elsevier B.V.

1. Introduction

Given a string (or text) T and pattern P under an alphabet Σ, the classic string/pattern matching problem asks whether P occurs in T and if yes, then it further reports the occurrences of P in T. This problem has extensive applications in different branches of science and engineering. Due to different types of requirements in different application scenarios, a plethora of variants of the classic string matching problem have been introduced and studied in the literature. The focus of this paper is a very recent variant which is called the order preserving pattern matching (OPPM). In OPPM, like the classic variant, we have a text T and a pattern P as input. However, the underlying alphabet here is an integer alphabet. And, instead of looking for a substring of the text which is identical to the given pattern, we are interested in locating a fragment which is order-isomorphic with the pattern. Two sequences over an integer alphabet are order-isomorphic if the relative order between any two elements at the same positions in both the sequences is the same.

To the best of our knowledge, OPPM was first studied independently by Kim et al. [6] and Kubica et al. [8].[2] Since then, within quite a short period of time, a number of works on OPPM in different directions have been reported in the literature. For example, order preserving suffix trees and index data structures have been devised and different applications thereof have been discussed by Crochemore et al. [2,3]. On the other hand, Cho et al. [1] have presented practically fast algorithms for OPPM. Gawrychowski and Uznanski [4] have considered the approximate version of the problem where k mismatches are allowed.

In this paper, we revisit the order preserving pattern matching problem. Our main contribution in this paper is the study of string regularities from an order preserving point of view. To the best of our knowledge, this is the first attempt to capture the concept regularities form this perspective. In what follows we will conveniently refer to this as *order preserving regularities*. String regularities have been the focus of attention of stringology researchers since long. The following has been articulated by Smyth [10] in a very recent survey on string regularities:

> … In the intervening century, certainly thousands of research papers have been written by mathematicians and (over the last half century) also computer scientists that relate in some way to periodicity, or its variants, in strings. A word that has recently been brought into service to describe these variants is “regularities” …

Apart from periodicity, the most notable and studied regularities of a string are borders and covers. Hence, in this paper we consider periods, borders and covers of strings from the order preserving point of view (Section 4). We also discuss yet another order preserving pattern matching algorithm (Section 4). Our algorithms are based on the so called Z-algorithm discussed by Gusfield in Chapter 2 of his famous book [5]. In particular, we propose modifications to the Z-algorithm or Z-function of Gusfield to make it useful in the order preserving framework (Section 3). The modified Z-algorithm for a string presented in this paper could be of independent interest.

☆ This paper has been recommended for acceptance by A. Koleshnikov.

* Corresponding author at: Commonwealth Academic Fellow funded by the UK Government. Currently on a sabbatical leave from BUET. Tel.: +8801552389480; fax: +448712475276.

E-mail address: msrahman@cse.buet.ac.bd (M. Sohel Rahman).

[1] Currently working at Google Zurich.

[2] The problem was also presented at the Theo Murphy International Scientific Meeting of the Royal Society on Storage and Indexing of Massive Data.

http://dx.doi.org/10.1016/j.patrec.2014.11.013
0167-8655/© 2014 Published by Elsevier B.V.

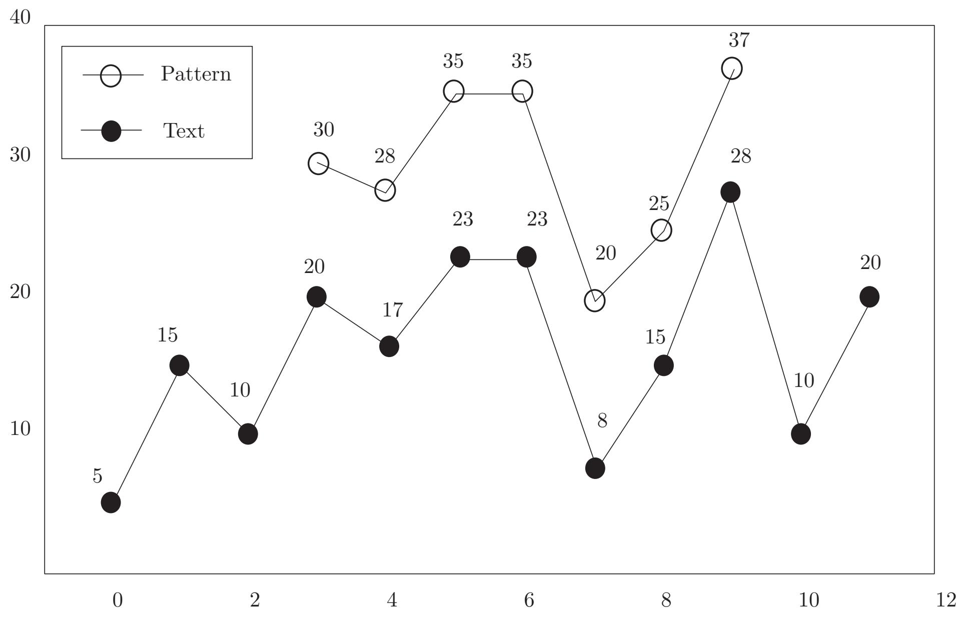

Fig. 1. An example instance of OPPM.

Table 1
An example of Z-function.

S	=	A	C	A	G	G	T	A	C	A	G	T	T	C	C	C	T	C	G	A	C	A	C	C	T	A	C	T	A	C	C	T	A	A	G
$Z(S)$	=	34	0	1	0	0	0	4	0	1	0	0	0	0	0	0	0	0	0	3	0	2	0	0	0	2	0	0	2	0	0	0	1	1	0

Besides the inherent combinatorial and algorithmic beauty of the problem itself, the motivation also comes from a number of interesting practical applications. For example, it can be applied to a time series analysis like share prices on stock markets to recognize specific patterns and to musical melody matching of two musical scores [1,6]. An example is given in Fig. 1. The problem also has some interesting relation to the combinatorial study of patterns in permutations, in particular to the study of pattern avoidance [8].

2. Preliminaries

We use Σ to denote the set of numbers such that the comparison of two numbers can be done in constant time and Σ^* denotes the set of strings over the alphabet Σ. The length of a string $S = (S[1], S[2], \ldots, S[n])$ is denoted by $|S| = n$. For $n = 0$, we say that $S = \varepsilon$, the empty string. If $S = UVW$, then U is said to be a prefix, V a substring (also called a factor) and W a suffix of S. If $VW \neq \varepsilon$ $(UV \neq \varepsilon)$ then U (W) is a proper prefix (proper suffix) of S. Similarly, if $UW \neq \varepsilon$, then V is a proper substring. A substring $(S[i], S[i+1], \ldots, S[j])$ of S is denoted by $S[i, j]$, the ith prefix $S[1, i]$ by prefix$_i(S)$ and the ith suffix $S[i, |S|]$ by suffix$_i(S)$.

The rank of a number c in a string S is defined as follows:

$$\mathrm{rank}_S(c) = 1 + |\{i \mid S[i] < c, 1 \leq i \leq |S|\}|.$$

To be consistent with the previous works [6], we assume that all the numbers in a string are distinct. The order preserving representation (OPR) $\sigma(S)$ of a string S can be defined as follows:

$$\sigma(S) = \mathrm{rank}_S(S[1]), \mathrm{rank}_S(S[2]), \ldots, \mathrm{rank}_S(S[|S|]).$$

If two strings S_1 and S_2 have identical OPR, i.e., $\sigma(S_1) = \sigma(S_2)$, then we say that the two strings are OPR-equal.

Since we are interested in string regularities, here we discuss some related notions and definitions. Part of the following description is conveniently adopted from a recent survey of Smyth [10]. If $S = S[1 \ldots n]$ has a proper (though possibly empty) prefix U that is also a suffix of S, then U is said to be a border of S. For example, for string $S = (1, 2, 1, 2, 1)$, one border would be $(1, 2, 1)$. The entire string can be considered as a special *border*. If for some $p \in 1 \ldots n$, $S[i] = S[p + i]$ for every $i \in 1 \ldots n - p$, then S is said to have period p. Thus S always has the empty border ε and the trivial period n. It is well-known, and easy to prove, that S has period p if, and only if, it has a border of length $n - p$. The border array β_S of a string S is an array of length n such that $\beta_S[i]$ equals the length of the longest border of $S[1 \ldots i]$ for every $i \in 1 \ldots n$. Since $\beta_S[i] = b > 0$ implies that $\beta_S[b]$ is the next largest border of $S[1 \ldots i]$, it follows that β_S specifies all the borders, hence all the periods, of every prefix of S. A simple $\Theta(n)$-time algorithm can compute the border array of a string having length n [9].

A string S has quasiperiod $q < n$ if and only if there exists a string $U = U[1 \ldots q]$, called a cover of S, such that every position of S lies within an occurrence of U. Thus a cover must also be a border of S. For example, $U = (1, 2, 1)$ is a cover of $S = (1, 2, 1, 2, 1, 1, 2, 1)$. A cover $U \neq S$ is called a *proper cover*.

Since we will be heavily using the Z-function of Gusfield [5], here we briefly review the concept. The Z-function, $Z_i(S)$ is the length of the longest substring of S that starts at position i and matches a prefix of S. We give an example of the Z-function for a string S on the DNA alphabet (i.e., $\{A, C, G, T\}$) in Table 1.

3. Modified *Z*-function

3.1. Definition

In this section, we propose a modification of the Z-function to make it useful from the order preserving point of view. We start with the following formal definition. For the sake of notational ease, we do not introduce an extended notation to denote Z-function from order preserving point of view; so, in the rest of this paper, unless otherwise specified, we will continue to use the term Z-function to consider it from the order preserving point of view.

Definition 1. Given a string S, the Z-function $Z_i(S)$, $1 \leq i \leq |S|$ is the length of the longest prefix P of suffix$_i(S)$ such that $\sigma(P) = \sigma(S[1, |P|])$.