Theoretical Computer Science 582 (2015) 74–82

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

Note

On hardness of several string indexing problems ☆

Kasper Green Larsen [a], J. Ian Munro [b], Jesper Sindahl Nielsen [a], Sharma V. Thankachan [c,*]

[a] MADALGO, Aarhus University, Denmark
[b] Cheriton School of CS, University of Waterloo, Canada
[c] School of CSE, Georgia Institute of Technology, USA

ARTICLE INFO

Article history:
Received 12 August 2014
Received in revised form 9 March 2015
Accepted 16 March 2015
Available online 21 March 2015
Communicated by R. Giancarlo

Keywords:
Document retrieval
Data structures
String searching
Lower bounds
Boolean matrix multiplication

ABSTRACT

Let $\mathcal{D} = \{d_1, d_2, \ldots, d_D\}$ be a collection of D string documents of n characters in total. The two-pattern matching problems ask to index $\mathcal{D}$ for answering the following queries efficiently.

- Report/count the unique documents containing P_1 *and* P_2.
- Report/count the unique documents containing P_1, *but not* P_2.

Here P_1 and P_2 represent input patterns of length p_1 and p_2 respectively. Linear space data structures with $O(p_1 + p_2 + \sqrt{nk}\log^{O(1)} n)$ query cost are already known for the reporting version, where k represents the output size. For the counting version (i.e., report the value k), a simple linear-space index with $O(p_1 + p_2 + \sqrt{n})$ query cost can be constructed in $O(n^{3/2})$ time. However, it is still not known if these are the best possible bounds for these problems. In this paper, we show a strong connection between these string indexing problems and the boolean matrix multiplication problem. Based on this, we argue that these results cannot be improved significantly using purely combinatorial techniques. We also provide an improved upper bound for a related problem known as *common colors query problem*.

Published by Elsevier B.V.

1. Introduction

Document listing is a fundamental problem in information retrieval, where the task is to index a collection of documents, such that whenever a pattern P comes as a query, we can efficiently find the unique documents containing P as a substring. This problem was introduced by Matias et al. and they provide a linear space and near-optimal time solution [2]. Later Muthukrishnan improved the result by providing a linear-space and optimal query time index [3]. The counting case asks to find the number of documents containing the query pattern. See [4] for an excellent survey on more results and extensions of document retrieval problems. In this paper, our focus is on the case where the query consists of two patterns (known as *two-pattern query problems*). The formal definitions of the problems under consideration are given below.

☆ Early parts of this work appeared in CPM 2014 [1]. Work supported in parts by NSERC of Canada, the Canada Research Chairs program and the Danish National Research Foundation grant DNRF84 through Center for Massive Data Algorithmics (MADALGO).

* Corresponding author.
E-mail addresses: larsen@cs.au.dk (K.G. Larsen), imunro@uwaterloo.ca (J.I. Munro), jasn@cs.au.dk (J.S. Nielsen), sharma.thankachan@gmail.com (S.V. Thankachan).

http://dx.doi.org/10.1016/j.tcs.2015.03.026
0304-3975/Published by Elsevier B.V.

Problem 1 Given a set of strings $\mathcal{D} = \{d_1, d_2, \ldots, d_D\}$ with $\sum_{i=1}^{D} |d_i| = n$, preprocess $\mathcal{D}$ to answer queries: given two strings P_1 and P_2 report all i's where both P_1 and P_2 occur in d_i.

Problem 2 Given a set of strings $\mathcal{D} = \{d_1, d_2, \ldots, d_D\}$ with $\sum_{i=1}^{D} |d_i| = n$, preprocess $\mathcal{D}$ to answer queries: given two strings P^+ and P^- report all i's where P^+ occurs in string d_i and P^- does not occur in string d_i.

Problem 3 Let $\mathcal{D} = \{(d_{1,1}, d_{1,2}), (d_{2,1}, d_{2,2}), \ldots, (d_{D,1}, d_{D,2})\}$ be a set of pairs of strings with $\sum_{i=1}^{D} |d_{i,1}| + |d_{i,2}| = n$. Preprocess $\mathcal{D}$ to answer queries: given two strings P_1 and P_2 report all i's where P_1 occurs in $d_{i,1}$ and P_2 occurs in $d_{i,2}$.

Problem 1 was introduced by Muthukrishnan [3]. He presented a data structure using $O(n^{1.5}\log^{O(1)} n)$-space (in words) with $O(p_1 + p_2 + \sqrt{n} + k)$ time for query processing, where $p_1 = |P_1|$ and $p_2 = |P_2|$ and k is the output size.[1] Later Cohen and Porat [5] presented a space efficient structure of $O(n \log n)$-space, but with a higher query time of $O(p_1 + p_2 + \sqrt{nk \log n} \log^2 n)$. The space and the query time of was improved by Hon et al. [6] to $O(n)$ words and $O(p_1 + p_2 + \sqrt{nk \log n} \log n)$ time. See [7] for a succinct space solution for this problem as well as an improved linear space structure with query time $O(p_1 + p_2 + \sqrt{nk \log n} \log\log n)$. Problem 2 is known as the forbidden (or excluded) pattern query problem. This was introduced by Fischer et al. [8], where they presented an $O(n^{3/2})$-bit solution with query time $O(p_1 + p_2 + \sqrt{n} + k)$. Immediately, Hon et al. [9] improved its space occupancy to $O(n)$ words, but with a higher query time of $O(p_1 + p_2 + \sqrt{nk \log n} \log^2 n)$. They presented an $O(n)$-space and $O(p_1 + p_2 + \sqrt{n} \log\log n)$ query time structure for the counting version of Problem 2 (i.e., just report the value k). We remark that the same framework can be adapted to handle the counting version of Problem 1 as well. Also the $O(\log\log n)$ term in the query time can be removed by replacing predecessor search queries in their algorithm by range emptiness queries. In summary, we have $O(n)$-space and $\tilde{\Omega}(\sqrt{n})$ query time solutions for the reporting/counting versions of these problems. However, the question whether these are the best possible bounds remains unanswered.

Problem 3 is known as the *two-dimensional substring indexing* problem, and was introduced by Ferragina et al. [10]. They reduced it to another problem known as the *common colors query* problem, where the task is to preprocess an array of colors and maintain a data structure, such that whenever two ranges comes as a query, we can output the unique colors which are common to both ranges. Based on their solution for this new problem, they presented an $O(n^{2-\varepsilon})$ space and $O(n^{\varepsilon} + k)$ query time solution for Problem 3, where ε is any constant in $(0, 1]$. Later Cohen and Porat [5] presented a space efficient solution for the common colors query problem of space $O(n \log n)$ words and query time $O(\sqrt{nk \log n} \log^2 n)$. Therefore, the current best data structure for *two-dimensional substring indexing* problem occupies $O(n \log n)$ space and processes a query in $O(p_1 + p_2 + \sqrt{nk \log n} \log^2 n)$ time.

Problems 1 and 3 have been independently studied but we note that they are actually equivalent up to constant factors. Suppose we have a solution for Problem 3, and we are given the input for Problem 1, i.e. a set of strings $\mathcal{D} = \{d_1, d_2, \ldots, d_D\}$. Now build the data structure for Problem 3 with the input $\mathcal{D}' = \{(d_1, d_1), (d_2, d_2), \ldots, (d_D, d_D)\}$. The queries remain the same. This reduction has an overhead factor of 2.

Similarly, suppose we have a solution for Problem 1 and we are given the input for Problem 3, i.e. $\mathcal{D} = \{(d_{1,1}, d_{1,2}), (d_{2,1}, d_{2,2}), \ldots, (d_{D,1}, d_{D,2})\}$. We make a new alphabet Σ' such that $|\Sigma'| = 2|\Sigma|$. Now create the set of strings $\mathcal{D}' = \{d_1, d_2, \ldots, d_D\}$ where $d_i = d_{i,1} d'_{i,2}$ and $d'_{i,2}$ is $d_{i,2}$ where each character σ is changed to $\sigma + |\Sigma|$. A query is changed in the same manner: (P_1, P_2) is changed to (P_1, P'_2) where P'_2 is P_2 with each character σ replaced by $\sigma + |\Sigma|$. This reduction increases the input length by a factor of at most 2 (one extra bit per character).

1.1. Our results

In this paper, we use the Word-RAM model of computation with word size $w = \Omega(\log n)$. The following summarizes our main results.

- We present a strong connection between the counting versions of the string indexing problems (Problem 1, 2, and 3) and the boolean matrix multiplication problem. Specifically, we show that multiplying two $\sqrt{n} \times \sqrt{n}$ boolean matrices can be reduced to the problem of indexing $\mathcal{D}$ (in Problem 1, Problem 2, or Problem 3) and answering n counting queries. However, matrix multiplication is a well known hard problem and this connection gives us a hardness result for the pattern matching problems under considerations.
- We present an improved upper bound for the common colors query problem, where the space and query time are $O(n)$ and $O(\sqrt{nk} \log^{1/2+\varepsilon} n)$ respectively, where $\varepsilon > 0$ is any constant. Therefore, we now have a linear-space and $O(p_1 + p_2 + \sqrt{nk} \log n)$ query time index for the two-dimensional substring indexing problem (Problem 3).

2. Hardness results

The hardness results are reductions from *boolean matrix multiplication*. Through this section we use similar techniques to [11–13]. In the boolean matrix multiplication problem we are given two $n \times n$ matrices A and B with $\{0, 1\}$ entries. The

[1] Specifically k is the maximum of 1 and the output size.