Theoretical Computer Science ••• (••••) •••–•••

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

A space-efficient alphabet-independent Four-Russians' lookup table and a multithreaded Four-Russians' edit distance algorithm

Youngho Kim [a], Joong Chae Na [b], Heejin Park [c], Jeong Seop Sim [a,*]

[a] Department of Computer and Information Engineering, Inha University, Incheon 22212, South Korea
[b] Department of Computer Science and Engineering, Sejong University, Seoul 05006, South Korea
[c] Department of Computer Science and Engineering, Hanyang University, Seoul 04763, South Korea

A R T I C L E I N F O

Article history:
Received 7 August 2015
Received in revised form 16 April 2016
Accepted 25 April 2016
Available online xxxx

Keywords:
Approximate string matching
Edit distance
Four-Russians' algorithm
Parallelization

A B S T R A C T

Given two strings X ($|X| = m$) and Y ($|Y| = n$) over an alphabet Σ, the edit distance between X and Y can be computed in $O(mn/t)$ time with the help of the Four-Russians' lookup table whose block size is t. The Four-Russians' lookup table can be constructed in $O((3|\Sigma|)^{2t}t^2)$ time using $O((3|\Sigma|)^{2t}t)$ space. However, the construction time and space requirement of the lookup table grow very fast as the alphabet size increases and thus it has been used only when $|\Sigma|$ is very small. For example, when a string is a protein sequence, $|\Sigma| = 20$ and thus it is almost impossible to use the Four-Russians' lookup table on typical workstations. In this paper, we present an efficient alphabet-independent Four-Russians' lookup table. It requires $O(3^{2t}(2t)!t)$ space and can be constructed in $O(3^{2t}(2t)!t^2)$ time. Thus, the Four-Russians' lookup table can be constructed and used irrespective of the alphabet size. The time and space complexity were achieved by compacting the lookup table using a clever encoding of the preprocessed strings. Experimental results show that the space requirement of the lookup table is reduced to about 1/5,172,030 of its original size when $|\Sigma| = 26$ and $t = 4$. Furthermore, we present efficient multithreaded parallel algorithms for edit distance computation using the Four-Russians' lookup table. The parallel algorithm for lookup table construction runs in $O(t)$ time and the parallel algorithm for edit distance computation between X and Y runs in $O(m + n)$ time. Experiments performed on CUDA-supported GPU show that our algorithm runs about 942 times faster than the sequential version of the original Four-Russians' algorithm for 100 pairs of random strings of length approximately 1,000 when $|\Sigma| = 4$ and $t = 4$.

1. Introduction

Approximate string matching problems have been studied in diverse fields such as search engines [1,2], querying big data [3,4], computer security [5–7], bioinformatics [8–11] and so on. To measure the errors between strings in approximate string matching distance functions are used, such as the Hamming distance, the edit distance, and the weighted edit distance. Given two strings X and Y over an alphabet Σ, the Hamming distance between X and Y is the minimum number of

* Corresponding author.
E-mail addresses: yhkim8505@gmail.com (Y. Kim), jcna@sejong.ac.kr (J.C. Na), hjpark@hanyang.ac.kr (H. Park), jssim@inha.ac.kr (J.S. Sim).

http://dx.doi.org/10.1016/j.tcs.2016.04.028
0304-3975/© 2016 Elsevier B.V. All rights reserved.

		a	*b*	*c*	*a*	*c*	*d*	*c*	*a*	*c*
	0	1	2	3	4	5	6	7	8	9
a	1	0	1	2	3	4	5	6	7	8
b	2	1	0	1	2	3	4	5	6	7
a	3	2	1	1	1	2	3	4	5	6
b	4	3	2	2	2	2	3	4	5	6
c	5	4	3	2	3	2	3	3	4	5
a	6	5	4	3	2	3	3	4	3	4

Fig. 1. D table for $X = ababca$ and $Y = abcacdcac$.

change operations needed to convert X to Y when $|X| = |Y|$. The edit distance between X and Y is the minimum number of edit operations needed to convert X to Y. At this time, edit operations consist of insertions, deletions, and changes. The weighted edit distance between X and Y is the minimum cost needed to convert X to Y using a penalty matrix.

Recently, fast approximate string matching are likely to become even more severe due to the rapid growth in the size of these databases called big data. The edit distance between X ($|X| = m$) and Y ($|Y| = n$) can be computed using the well-known dynamic programming technique in $O(mn)$ time and space [12,13]. The edit distance also can be computed using the Four-Russians' algorithm. The Four-Russians' algorithm for computing edit distances consists of two steps, the preprocessing step and the computation step. The preprocessing step runs in $O((3|\Sigma|)^{2t}t^2)$ time to construct the lookup table of $O((3|\Sigma|)^{2t}t)$ size and the computation step runs in $O(mn/t)$ time to compute the edit distance between X and Y where t represents the length of the preprocessed strings [14,15]. However, the construction time and space requirement of the lookup table grow very fast as the alphabet size increases and thus it has been used only when $|\Sigma|$ is very small. For example, when a string is a protein sequence, $|\Sigma| = 20$ and thus it is almost impossible to use the Four-Russians' lookup table on typical workstations.

Our contributions are as follows.

- We present an alphabet-independent and space-efficient Four-Russians' lookup table. It requires $O(3^{2t}(2t)!t)$ space and can be constructed in $O(3^{2t}(2t)!t^2)$ time. Thus, the Four-Russians' lookup table can be used irrespective of the alphabet size. The time and space complexities were achieved by compacting the lookup table using a clever encoding of the preprocessed strings. Experimental results show that the space requirement of the lookup table is reduced to about 1/5,172,030 of its original size when $|\Sigma| = 26$ and $t = 4$.
- We present multithreaded parallel Four-Russians' algorithms. Our parallel algorithm for lookup table construction runs in $O(t)$ time using $3^{2t}(2t)!t$ threads and parallel edit distance computation runs in $O(m+n)$ time using m/t threads. By elaborated experiments, we show that our parallel algorithm computes edit distances between 100 pairs of two strings about 13 $\sim$ 942 times faster than the original Four-Russians' algorithm.

This paper is organized as follows. In Section 2, we describe the related works for the edit distance problem. In Section 3, we present our algorithm reducing sizes of lookup tables and a parallel version of the algorithm. Then we show experimental results in Section 4. We conclude in Section 5.

2. Related works

2.1. Dynamic programming for edit distances

Let X and Y be strings of lengths m and n over an alphabet Σ, respectively. We assume that $m \le n$ for convenience. We denote the i-th character of X by $X[i]$ and a substring $X[i]X[i+1]...X[j]$ by $X[i..j]$. It is well known that the edit distance between X and Y can be computed using a dynamic programming in $O(mn)$ time. Let D be a table of size $(m+1) \times (n+1)$ whose entry $D[i, j]$ $(0 \le i \le m, 0 \le j \le n)$ stores the edit distance between $X[1..i]$ and $Y[1..j]$. Initially, $D[i, 0] = i$ $(0 \le i \le m)$, $D[0, j] = j$ $(1 \le j \le n)$. Then the other entries can be computed by the following recurrence:

$$D[i, j] = min \begin{cases} D[i-1, j] + 1, \\ D[i, j-1] + 1, \\ D[i-1, j-1] + \delta(X[i], Y[j]) \end{cases} \tag{1}$$

where $\delta(X[i], Y[j]) = 0$ if $X[i] = Y[j]$, and $\delta(X[i], Y[j]) = 1$ otherwise. Fig. 1 shows the D table for $X = ababca$ and $Y = abcacdcac$.