Journal of Discrete Algorithms 9 (2011) 344–357

Contents lists available at ScienceDirect

Journal of Discrete Algorithms

www.elsevier.com/locate/jda

An $O(n^{3/2}\sqrt{\log(n)})$ algorithm for sorting by reciprocal translocations

Michal Ozery-Flato, Ron Shamir *

The Blavatnik School of Computer Science, Tel Aviv University, Tel Aviv 69978, Israel

ARTICLE INFO

Article history:
Available online 8 April 2011

Keywords:
Translocations
Reversals
Genome rearrangements

ABSTRACT

We prove that sorting by reciprocal translocations can be done in $O(n^{3/2}\sqrt{\log(n)})$ for an n-gene genome. Our algorithm is an adaptation of the algorithm of Tannier, Bergeron and Sagot for sorting by reversals. This improves over the $O(n^3)$ algorithm for sorting by reciprocal translocations given by Bergeron, Mixtacki and Stoye (2006) [4].

1. Introduction

In this paper we study the problem of sorting by reciprocal translocations (abbreviated SRT). *Reciprocal translocations* exchange non-empty ends between two chromosomes. Given two multi-chromosomal genomes A and B, the problem of SRT is to find a shortest sequence of reciprocal translocations that transforms A into B. SRT was first introduced by Kececioglu and Ravi [11] and was given a polynomial time algorithm by Hannenhalli [6]. Bergeron, Mixtacki and Stoye [4] pointed to an error in Hannenhalli's proof of the reciprocal translocation distance formula and consequently in Hannenhalli's algorithm. They presented a new $O(n^3)$ algorithm, which to the best of our knowledge, is the only extant correct algorithm for SRT.[1]

Reversals (or inversions) reverse the order and the direction of transcription of the genes in a segment inside a chromosome. Given two uni-chromosomal genomes π_1 and π_2, the problem of sorting by reversals (abbreviated SBR) is to find a shortest sequence of reversals that transforms π_1 into π_2. This problem has been intensively studied [8,5,9,1,2,15]. Tannier, Bergeron and Sagot [15] presented an elegant algorithm for SBR that can be implemented in $O(n^{3/2}\sqrt{\log(n)})$ using a clever data structure by Kaplan and Verbin [10]. This is currently the fastest algorithm for SBR.

In this paper we prove that SRT can be solved in $O(n^{3/2}\sqrt{\log(n)})$ for an n-gene genome. Our algorithm for SRT is similar to the algorithm by Tannier, Bergeron and Sagot [15] for SBR. The key idea is to recast translocations as reversals, and then exploit the novel theoretical improvements in SBR theory to obtain faster SRT algorithms. (It should be noted that Hannenhalli and Pevzner have already established and exploited the basic connection between translocations and reversals, in the context of sorting a genome by reversals and translocations [7].) Our approach builds on generalizing the overlap graph. Most studies of SBR to date relied explicitly or implicitly on the combinatorial structure of the overlap graph for representing the relations between two permutations. Since translocations involve multiple chromosomes, we generalize the notion of (uni-chromosomal) overlap graph to include chromosomal information, and show that the same conceptual algorithmic framework developed for SBR applies to SRT, via this generalized overlap graph. While our final algorithm is very similar to that of Tannier et al., the proofs had to be completely redone. Another contribution of this study is in

* Corresponding author.
E-mail address: rshamir@tau.ac.il (R. Shamir).

[1] Li et al. [12] gave a linear time algorithm for computing the reciprocal translocation distance (without producing a shortest sequence). Wang et al. [16] presented an $O(n^2)$ algorithm for SRT. However, the algorithms in [12,16] rely on an erroneous theorem of Hannenhalli and hence provide incorrect results in certain cases.

1570-8667/$ – see front matter
doi:10.1016/j.jda.2011.04.003

showing that the general SRT problem can be reduced in linear time to a special case, and thus time complexity analysis can be done for such special cases only.

The paper is organized as follows. The necessary preliminaries are given in Section 2. In Section 3 we give a linear time reduction from SRT to a simpler restricted subproblem. In Section 4 we prove the main theorem and present the algorithm for the restricted subproblem. In Section 5 we describe an $O(n^{3/2}\sqrt{\log(n)})$ implementation of the algorithm. A preliminary version of this study was published in the proceedings of CPM 2006 [13].

2. Preliminaries

This section provides a basic background for the analysis of SRT. It follows to a large extent the nomenclature and notation of [6,9,4]. In the model we consider, a *genome* is a set of chromosomes. A *chromosome* is a sequence of genes. A *gene* is identified by a positive integer. All genes in the genome are distinct. When it appears in a genome, a gene is assigned a sign of plus or minus. For example, the following genome consists of 8 genes in two chromosomes:

$$A_1 = \{(1, -3, -2, 4, -7, 8), (6, 5)\}$$

The *reverse* of a sequence of genes $I = (x_1, \ldots, x_l)$ is $-I = (-x_l, \ldots, -x_1)$. A *reversal* reverses a segment of genes inside a chromosome. Two chromosomes, X and Y, are *identical* if either $X = Y$ or $X = -Y$. Therefore, *flipping* chromosome X into $-X$ does not affect the chromosome it represents. For example, the following are two equivalent representations of the same genome

$$\{\underline{(1, -3, -2, 4, -7, 8)}, (6, 5)\} \equiv \{\underline{(-8, 7, -4, 2, 3, -1)}, (6, 5)\}$$

Let $X = (X_1, X_2)$ and $Y = (Y_1, Y_2)$ be two chromosomes, where X_1, X_2, Y_1, Y_2 are sequences of genes. A *translocation* cuts X into X_1 and X_2 and Y into Y_1 and Y_2 and exchanges segments between the chromosomes. It is called *reciprocal* if X_1, X_2, Y_1 and Y_2 are all non-empty. There are two ways to perform a translocation on X and Y. A *prefix–suffix* translocation switches X_1 with Y_2 resulting in:

$$(\underline{X_1}, X_2), (Y_1, \underline{Y_2}) \quad \Rightarrow \quad (\underline{-Y_2}, X_2), (Y_1, \underline{-X_1})$$

A *prefix–prefix* translocation switches X_1 with Y_1 resulting in:

$$(\underline{X_1}, X_2), (\underline{Y_1}, Y_2) \quad \Rightarrow \quad (\underline{Y_1}, X_2), (\underline{X_1}, Y_2)$$

The following is an example of prefix–prefix and prefix–suffix translocations that cut the genome in the same place:

$$\{(\underline{1, -3, -2, 4}, -7, 8), (\underline{6}, 5)\} \quad \Rightarrow \quad \{(\underline{6}, -7, 8), (\underline{1, -3, -2, 4}, 5)\}$$
$$\{(\underline{1, -3, -2, 4}, -7, 8), (6, \underline{5})\} \quad \Rightarrow \quad \{(\underline{-5}, -7, 8), (6, \underline{-4, 2, 3, -1})\}$$

Recall that chromosome flips do not affect the genome, but rather move between different representations of the same genome. Thus we can mimic one type of translocation by a flip of one of the chromosomes followed by a translocation of the other type.

For a chromosome $X = (x_1, \ldots, x_k)$ define $\mathit{Tails}(X) = \{x_1, -x_k\}$. Note that flipping X does not change $\mathit{Tails}(X)$. For a genome A define $\mathit{Tails}(A) = \bigcup_{X \in A} \mathit{Tails}(X)$. For example:

$$\mathit{Tails}(A_1) = \mathit{Tails}(\{(1, -3, -2, 4, -7, 8), (6, 5)\}) = \{1, -8, 6, -5\}$$

Two genomes A' and A'' are *co-tailed* if $\mathit{Tails}(A') = \mathit{Tails}(A'')$. In particular, two co-tailed genomes have the same number of chromosomes (recall that all genes in a genome are unique). Note that if A'' was obtained from A' by performing a reciprocal translocation then $\mathit{Tails}(A'') = \mathit{Tails}(A')$. Therefore, SRT is defined only for genomes that are co-tailed. For the rest of this paper the word "translocation" refers to a reciprocal translocation and we assume that the given genomes, A and B, are co-tailed.

2.1. The cycle graph

In this section we present the cycle graph of genomes A and B, which was first defined in [6]. Let N be the number of chromosomes in A (equivalently, B). We shall always assume that both A and B contain the genes $\{1, \ldots, n\}$. The *cycle graph* of A and B, denoted $G(A, B)$, is an undirected graph defined as follows. The set of vertices is $\bigcup_{i=1}^{n} \{i^0, i^1\}$. The vertices i^0 and i^1 are called the two *ends* of gene i (think of them as the ends of a small arrow directed from i^0 to i^1). For every pair of genes, i and j, where j immediately follows i in some chromosome of A (respectively, B) add a black (respectively, gray) (undirected) edge

$$(i, j) \equiv \big(\mathit{out}(i), \mathit{in}(j)\big)$$