Sorting by translocations via reversals theory

Michal Ozery-Flato and Ron Shamir

School of Computer Science, Tel-Aviv University, Tel Aviv 69978, Israel
{ozery,rshamir}@post.tau.ac.il

Abstract. The understanding of genome rearrangements is an important endeavor in comparative genomics. A major computational problem in this field is finding a shortest sequence of genome rearrangements that "sorts" one genome into another. In this paper we focus on sorting a multi-chromosomal genome by translocations. We reveal new relationships between this problem and the well studied problem of sorting by reversals. Based on these relationships, we develop two new algorithms for sorting by translocations, which mimic known algorithms for sorting by reversals: a score-based method building on Bergeron's algorithm, and a recursive procedure similar to the Berman-Hannenhalli method. Though their proofs are more involved, our procedures for translocations match the complexities of the original ones for reversals.

1 Introduction

For over a decade, much effort has been put into large-scale genome sequencing projects. Analysis of biological sequence data that have accumulated so far showed that genome rearrangements play an important role in the evolution of species. A major computational problem in the research of genome rearrangements is finding a most parsimonious sequence of rearrangements that transforms one genome into the other. This is called the *genomic sorting problem*, and the corresponding number of rearrangements is called the *genomic distance* between the two genomes. Genomic sorting gives rise to a spectrum of fascinating combinatorial problems, each defined by the set of allowed rearrangement operations and by the representation of the genomes.

In this paper we focus on the problem of sorting by translocations. We reveal new similarities between sorting by translocations and the well studied problem of sorting by reversals. The study of the problem of sorting by translocations is essential for the full comprehension of any genomic sorting problem that permits translocations. Below we review the relevant previous results and summarize our results. Formal definitions are provided on the next section.

Following the pioneering work by Nadeau and Taylor [11], reversals and translocations are believed to be very common in the evolution of mammalian species. *Reversals* (or inversions) reverse the order and the direction of transcription of the genes in a segment inside a chromosome. *Translocations* exchange tails between two chromosomes. A translocation is *reciprocal* if none of the exchanged tails is empty. The genomic sorting problem restricted to reversals (respectively, reciprocal translocations) is referred to as SBR (respectively, SRT).

SBR and SRT were both proven to be polynomial. Hannenhalli and Pevzner [7] gave the first polynomial algorithm for SBR and since then other more efficient algorithms and simplifications for the analysis have been presented. Berman and Hannenhalli [4] presented a recursive algorithm for SBR. Kaplan, Shamir and Tarjan [8] simplified the analysis and gave an $O(n^2)$ algorithm for SBR. Using a linear time algorithm by Bader, Moret and Yan [1] for computing the reversal distance, the algorithm of Berman and Hannenhalli can be implemented in $O(n^2)$. A score-based algorithm for SBR was presented by Bergeron [2]. Tannier, Bergeron and Sagot [13] 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 [9].

SRT was first introduced by Kececioglue and Ravi [10] and was given a polynomial time algorithm by Hannenhalli [5]. Bergeron, Mixtacki and Stoye [3] pointed to an error in Hannenhalli's proof of the translocation distance formula and consequently in Hannenhalli's algorithm. They presented a new proof followed by an $O(n^3)$ algorithm for SRT. In a recent study [12] we proved that the algorithm of Tannier et al.[13] for SBR can be adapted to solve SRT while preserving the original time complexity (that is $O(n^{3/2}\sqrt{log(n)})$).

It is well known that a translocation on a multi-chromosomal genome can be simulated by a reversal on a concatenation of the chromosomes [6]. However, different translocations require different concatenations. In addition, not all of the reversals on a concatenation of the chromosomes have matching translocations. Thus, from a first glance the similarity between SRT and SBT ends here. In [12] we presented the "overlap graph with chromosomes" of two multi-chromosomal genomes, which is an extension of the "overlap graph" of two uni-chromosomal genomes. This auxiliary graph established a new framework for the analysis of SRT that enabled us to adapt the currently fastest algorithm for SBR to SRT [13, 12]. In this paper we reveal new relationships between SRT and SBR. Based on these relationships we develop two new algorithms for SRT, which mimic known algorithms for SBR: a score-based method building on Bergeron's algorithm [2], and a recursive procedure similar to the Berman-Hannenhalli method [4]. Though the proofs of the algorithms are more involved than those of their counterparts for SBR, our procedures for translocations match the complexities of the original ones for reversals: the score-based algorithms performs $O(n^2)$ operations on $O(n)$ bit vectors; the recursive algorithm runs in $O(n^2)$ time.

The paper is organized as follows. Section 2 gives the necessary preliminaries. Section 3 presents the score-based algorithm and Section 4 presents the recursive algorithm.

2 Preliminaries

This section provides a basic background for the analysis of SRT. It follows to a large extent the nomenclature and notation of [5, 8]. 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.

A *signed permutation* $\pi = (\pi_1, \ldots, \pi_n)$ is a permutation on the integers $\{1, \ldots, n\}$, where a sign of plus or minus is assigned to each number. If A is a genome with the set of genes $\{1, \ldots, n\}$ then any concatenation π_A of the chromosomes of A is a signed permutation of size n. In the following, we assume w.l.o.g. that there is a concatenation of the chromosomes in B, π_B, which is identical to the identity permutation. For example,

$$B = \{(1, 2, \ldots, 5), (6, 7, 8)\}.$$

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:

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

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

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

Note that we can mimic a prefix-prefix (respectively, prefix-suffix) translocation by a flip of one of the chromosomes followed by a prefix-suffix (respectively, prefix-prefix) translocation. As was demonstrated by Hannenhalli and Pevzner [6], a translocation on A can be simulated by a reversal on π_A in the following way:

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

The type of translocation depends on the relative orientation of X and Y in π_A (and not on their order): if the orientation is the same, then the translocation is prefix-suffix, otherwise it is prefix-prefix. The segment between X_2 and Y_1 may contain additional chromosomes that are flipped and thus unaffected.

For a chromosome $X = (x_1, \ldots, x_k)$ define $Tails(X) = \{x_1, -x_k\}$. Note that flipping X does not change $Tails(X)$. For a genome A_1 define $Tails(A_1) = \bigcup_{X \in A_1} Tails(X)$. For example:

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

Two genomes A_1 and A_2 are *co-tailed* if $Tails(A_1) = Tails(A_2)$. In particular, two co-tailed genomes have the same number of chromosomes. Note that if A_2 was obtained from A_1 by performing a reciprocal translocation then $Tails(A_2) = Tails(A_1)$. 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

Let N be the number of chromosomes in A (equivalently, B). We shall always assume that both A and B contain genes $\{1, \ldots, n\}$. The *cycle graph* of A and B, denoted $G(A, B)$, is defined as follows. The set of vertices is $\bigcup_{i=1}^{n} \{i^0, i^1\}$. For every pair of adjacent genes in B, i and $i+1$, add a grey edge $(i, i+1) \equiv (i^1, (i+1)^0)$. For every pair of adjacent genes in A, i and j, add a black edge $(i, j) \equiv (out(i), in(j))$, where $out(i) = i^1$ if i has a positive sign in A and otherwise $out(i) = i^0$, and $in(j) = j^0$ if j has a positive sign in A and otherwise $in(j) = j^1$. An example is given in Fig. 1. There are $n - N$ black edges and $n - N$ grey edges in $G(A, B)$. A grey edge $(i, i+1)$ is *external* if the genes i and $i+1$ belong to different chromosomes of A, otherwise it is *internal*.

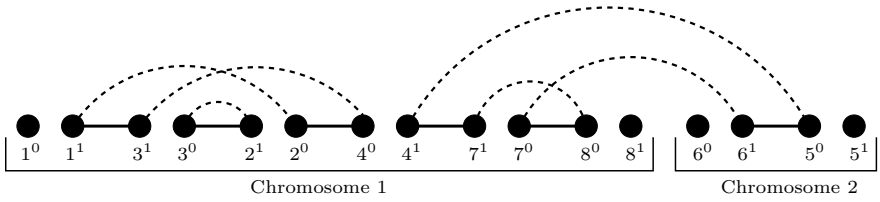

Fig. 1. The cycle graph $G(A_1, B_1)$, where $A_1 = \{(1, -3, -2, 4, -7, 8), (6, 5)\}$ and $B_1 = \{(1, \ldots, 5), (6, 7, 8)\}$. Dotted lines corresponds to grey edges.

Every vertex in $G(A, B)$ has degree 2 or 0, where vertices of degree 0 (isolated vertices) belong to $Tails(A)$ (equivalently, $Tails(B)$). Therefore, $G(A, B)$ is uniquely decomposed into cycles with alternating grey and black edges. Note that the cycle graph is uniquely decomposed into cycles iff A and B are co-tailed. An *adjacency* is a cycle with two edges.

2.2 The Overlap Graph with Chromosomes

Place the vertices of $G(A, B)$ along a straight line according to their order in π_A. Now, every grey edge can be associated with an interval of vertices of $G(A, B)$. Two intervals *overlap* if their intersection is not empty but none contains the other. The *overlap graph with chromosomes* of A and B w.r.t. π_A, denoted $\mathrm{OVCH}(A, B, \pi_A)$, is defined as follows. There are two types of vertices. The

first type corresponds to grey edges in $G(A, B)\}$. The second type corresponds to chromosomes of A. Two vertices are connected if their associated intervals of vertices overlap. For example see Fig. 2.

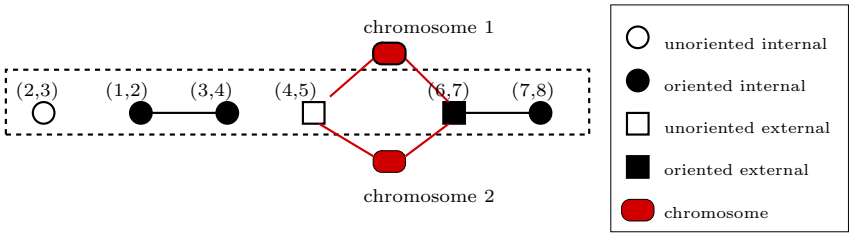

Fig. 2. The overlap graph with chromosomes $\mathrm{OVCH}(A_1, B_1, \pi_{A_1})$, where A_1, B_1 and π_{A_1} are as A_1 and B_1 are the genomes from Fig. 1 and $\pi_{A_1} = (1, -3, -2, 4, -7, 8, 6, 5)$. The graph induced by the vertices within the dashed rectangle is $\mathrm{OV}(A_1, B_1, \pi_{A_1})$.

In order to prevent confusion, we will refer to vertices that correspond to chromosomes as "chromosomes" and reserve the word "vertex" for the vertices that correspond to edges. A vertex in $\mathrm{OVCH}(A, B, \pi_A)$ is *external* iff there is an edge connecting it to a chromosome, otherwise it is *internal*. Note that the internal/external state of a vertex in $\mathrm{OVCH}(A, B, \pi_A)$ does not depend on π_A (the partition of the chromosomes is known from A). A vertex in the overlap graph is *oriented* if its corresponding edge connects two genes with different signs in π_A, otherwise it is *unoriented*.

Let $\mathrm{OV}(A, B, \pi_A)$ be the subgraph of $\mathrm{OVCH}(A, B, \pi_A)$ induced by the set of vertices that correspond to grey edges (i.e. excluding the chromosomes' vertices). We shall use the word "component" for a connected component of $\mathrm{OV}(A, B, \pi_A)$. The set of components in $\mathrm{OVCH}(A, B, \pi_A)$ can be computed in linear time using an algorithm by Bader, Moret and Yan [1]. A component in $\mathrm{OVCH}(A, B, \pi_A)$ is *external* if at least one of the vertices in it is external, otherwise it is *internal*. A component is *trivial* if it is composed of one internal vertex. A trivial component corresponds to an adjacency. It is not hard to see that the set of internal components in $\mathrm{OVCH}(A, B, \pi_A)$ is independent of π_A. Denote by $\mathcal{IN}(A, B)$ the set of non-trivial internal components in $\mathrm{OVCH}(A, B, \pi_A)$.

2.3 The Reciprocal Translocation Distance

Let $c(A, B)$ denote the number of cycles in $G(A, B)$.

Theorem 1 *[3, 5] The reciprocal translocation distance between A and B is $d(A, B) = n - N - c(A, B) + F(A, B)$, where $F(A, B) \geq 0$ and $F(A, B) = 0$ iff $\mathcal{IN}(A, B) = \emptyset$.*

Let Δc denote the change in the number of cycles after performing a translocation on A. Then $\Delta c \in \{-1, 0, 1\}$ [5]. A translocation is *proper* if $\Delta c = 1$. translocation is *safe* if it does not create any new non-trivial internal component. A translocation ρ is *valid* if $d(A \cdot \rho, B) = d(A, B) - 1$. A It follows from Theorem 1 that if $\mathcal{IN}(A, B) = \emptyset$, then every safe proper translocation is necessarily valid.

In a previous paper [12] we presented a generic algorithm for SRT that uses a sub-procedure for solving SRT when $\mathcal{IN}(A, B) = \emptyset$. The generic algorithm focuses on the efficient elimination of the non-trivial internal components. We showed that the work performed by this generic algorithm, not including the sub-procedure calls, can be implemented in linear time. This led to the following theorem:

Theorem 2 *[12] SRT is linearly reducible to SRT with* $IN(A, B) = \emptyset$.

By the theorem above, it suffices to solve SRT assuming that $\mathcal{IN}(A, B) = \emptyset$. Both algorithms that we describe below will make this assumption.

2.4 The Effect of a Translocation on the Overlap Graph with Chromosomes

Let $H = \text{OVCH}(A, B, \pi_A)$ and let v be any vertex in H. Denote by $N(v) \equiv N(v, H)$ the set of vertices that are neighbors of v in H, including v itself (but not including chromosome neighbors). Denote by $\text{CH}(v) \equiv \text{CH}(v, H)$ the set of chromosomes that are neighbors of v in H. Hence if v is external then $|\text{CH}(v)| = 2$, otherwise $\text{CH}(v) = \emptyset$.

Every external grey edge e defines one proper translocation that cuts the black edges incident to e. (Out of the two possibilities of prefix-prefix or prefix-suffix translocations, exactly one would be proper.) For an external vertex v denote by $\rho(v)$ the proper translocation that the corresponding grey edge defines on A. Let $H \cdot \rho(v) = \text{OVCH}(A \cdot \rho(v), B, \pi_A)$. Given two sets S_1 and S_2 define $S_1 \bigoplus S_2 = (S_1 \bigcup S_2) \setminus (S_1 \bigcap S_2)$.

Lemma 1 *[12] Let v be an oriented external vertex in H. Then $H \cdot \rho(v)$ is obtained from H by the following operations. (i) Complement the subgraph induced by $N(v)$ and flip the orientation of every vertex in $N(v)$. (ii) For every vertex $u \in N(v)$ such that the endpoints of u and v share at least one common chromosome, complement the edges between u and $\text{CH}(u) \bigcup \text{CH}(v)$ (In other words $\text{CH}(u, H \cdot \rho(v)) = \text{CH}(u, H) \bigoplus \text{CH}(v, H)$).*

Two overlap graphs with chromosomes are *equivalent* if one can be obtained from the other by a sequence of chromosome flips. For a chromosome X let $\rho(X)$ denote a flip of chromosome X in π_A. Let $H \cdot \rho(X) = \text{OVCH}(A, B, \pi_A \cdot \rho(X))$.

Lemma 2 *[12] $H \cdot \rho(X)$ is obtained from H by complementing the subgraph induced by the set $\{u : X \in \text{CH}(u)\}$ and flipping the orientation of every vertex in it.*

It follows that an unoriented external vertex v in H becomes an oriented (external) vertex in $H \cdot \rho(X)$, where $X \in \mathrm{CH}(v)$.

3 A score-based algorithm

In this section we present a score-based algorithm for SRT when $\mathcal{IN}(A, B) = \emptyset$. This algorithm is similar to an algorithm by Bergeron for SBR [2].

Denote by $N_{\mathrm{IN}}(v)$ and $N_{\mathrm{EXT}}(v)$ the neighbors of v that are respectively internal and external. It follows that $N_{\mathrm{IN}}(v) \bigcup N_{\mathrm{EXT}}(v) \bigcup \{v\} = N(v)$.

Lemma 3 *Let v be an oriented external vertex in H and let w be a neighbor of v. w has no external neighbors in $H \cdot \rho(v)$ iff $N_{\mathrm{EXT}}(w) \subseteq N_{\mathrm{EXT}}(v)$ and $N_{\mathrm{IN}}(v) \subseteq N_{\mathrm{IN}}(w)$.*

For each vertex v in H we define the *score* $|N_{\mathrm{EXT}}(v)| - |N_{\mathrm{IN}}(v)|$. Define $\Delta\mathrm{IN}(H, v)$ as the set of vertices that belong to external components in H but are in non-trivial internal components in $H \cdot \rho(v)$.

Lemma 4 *Let O be a set of oriented external vertices. Let $v \in O$ be a vertex with maximal score in O. Then $O \bigcap \Delta\mathrm{IN}(H, v) = \emptyset$.*

Proof. Assume that $u \in O \bigcap \Delta\mathrm{IN}(H, v)$. Then $u \in N(v, H)$ and by Lemma 3 $N_{\mathrm{EXT}}(u) \subseteq N_{\mathrm{EXT}}(v)$ and $N_{\mathrm{IN}}(v) \subseteq N_{\mathrm{IN}}(u)$. However, since v has the maximal score in O, we get $N_{\mathrm{EXT}}(u) = N_{\mathrm{EXT}}(v)$ and $N_{\mathrm{IN}}(v) = N_{\mathrm{IN}}(u)$. Therefore, u is an isolated internal vertex in $H \cdot \rho(v)$, a contradiction for $u \in \Delta\mathrm{IN}(H, v)$. □

Theorem 3 *Let O be a non-empty set of* all *the oriented external vertices in H that overlap the same pair of chromosomes (i.e. $\mathrm{CH}(u) = \mathrm{CH}(v)$ for every $u, v \in O$). Let $v \in O$ be a vertex that has the maximal score in O. Let S be the set of all the vertices w that satisfy the following conditions in H:*

1. *w is a neighbor of v,*
2. *w is an unoriented external vertex and $\mathrm{CH}(w) = \mathrm{CH}(v)$,*
3. *$N_{\mathrm{EXT}}(w) \subseteq N_{\mathrm{EXT}}(v)$,*
4. *$N_{\mathrm{IN}}(v) \subseteq N_{\mathrm{IN}}(w)$, and*
5. *$O \bigcap N_{\mathrm{EXT}}(v) \subseteq N_{\mathrm{EXT}}(w)$.*

If $S = \emptyset$ then $\rho(v)$ is safe. Otherwise, let $w \in S$ be a vertex that has a maximal score in $H \cdot \rho(X)$, where $X \in \mathrm{CH}(v)$. Then $\rho(w)$ is safe.

Proof. Suppose $S = \emptyset$ and assume that v is not safe. Let $w \in \Delta\mathrm{IN}(H, v)$ be a neighbor of v in H. It follows from Lemma 1 that $\mathrm{CH}(w) = \mathrm{CH}(v)$. It follows from Lemmas 3 and 4 that $w \in S$, a contradiction.

Suppose $S \neq \emptyset$. Let $O_1 = O \bigcap N_{\mathrm{EXT}}(v)$. Then there are all possible edges between S and O_1 in H (last condition). Let $H' = H \cdot \rho(X)$, where $X \in \mathrm{CH}(v)$. In H' all the vertices in S are oriented. Moreover, there are no edges between S and $O_1 \bigcup \{v\}$ in H'. It follows that $O_1 \bigcup \{v\}$ remain external after performing a

translocation on any vertex in S. Let $w \in S$ be a vertex with maximal score in S and assume $\Delta\mathrm{IN}(H', w) \neq \emptyset$. Let $u \in \Delta\mathrm{IN}(H', w)$ be a neighbor of w in H'. Then u satisfies: *(i)* $\mathrm{CH}(u) = \mathrm{CH}(w)$ and *(ii)* there are no edges between u and $O_1 \bigcup \{v\}$ in H'. Moreover, u is oriented in H' since otherwise $u \in O_1$ and thus could not be a neighbor of w. It follows that u satisfies conditions 1, 2 and 5 in H. However, by Lemma 4 it follows that $u \notin S$. Hence there are two possible cases:

Case 1: $N_{\mathrm{EXT}}(u) \not\subseteq N_{\mathrm{EXT}}(v)$ in H (i.e. condition 3 is not satisfied). Suppose $z \in N_{\mathrm{EXT}}(u)$ and $z \notin N_{\mathrm{EXT}}(v)$ in H. Then $z \notin N_{\mathrm{EXT}}(w)$ in H (condition 3).

Case 1.*a*: $X \notin \mathrm{CH}(z)$. Then in H': $z \in N_{\mathrm{EXT}}(u)$ and $z \notin N_{\mathrm{EXT}}(w)$. Then according to Lemma 3, z has an external neighbor in $H' \cdot \rho(w)$, a contradiction.

Case 1.*b*: $X \in \mathrm{CH}(z)$. Then in H': $z \notin N(u)$, $z \in N(v)$, $z \in N(w)$. Therefore, in $H' \cdot \rho(w)$ the path u, z, v exists, a contradiction (since v is external in $H' \cdot \rho(w)$).

Case 2: $N_{\mathrm{IN}}(v) \not\subseteq N_{\mathrm{IN}}(u)$ in H (i.e. condition 4 is not satisfied). Then there exists $x \in N_{\mathrm{IN}}(v)$, $x \notin N_{\mathrm{IN}}(u)$ in H'. It follows from condition 4 that $x \in N_{\mathrm{IN}}(w)$ in H'. Since x is internal, all its edges exist in H'. It follows from Lemma 3 that u has an external neighbor (x) in $H' \cdot \rho(w)$, a contradiction. □

Theorem 3 immediately implies an $O(n^3)$ algorithm that can be implemented using operations on bit vectors, in a similar manner to the implementation of the algorithm of Bergeron [2] for SBR. The algorithm is presented in Fig. 3 and uses the following notations. $\boldsymbol{v}$ denotes a bit vector of size $n - N$ corresponding to a vertex v, where $\boldsymbol{v}[u] = 1$ iff u is a neighbor of v. $\boldsymbol{X}$ denotes a bit vector of size $n - N$ corresponding to chromosome X where $\boldsymbol{X}[v] = 1$ iff $X \in \mathrm{CH}(v)$. $\boldsymbol{ext}$ and $\boldsymbol{o}$ are two bit vectors of size $n - N$. $\boldsymbol{ext}[u] = 1$ iff u is external. $\boldsymbol{o}[u] = 1$ iff u is oriented. The score of each vertex is stored in an integer vector $\boldsymbol{score}$. $\bigwedge$, $\bigvee$, $\bigoplus$ and $\neg$ respectively denote the bitwise-AND, bitwise-OR, bitwise-XOR and bitwise-NOT operators.

One of the major differences between this algorithm and the original algorithm [2] is that in some cases our algorithm performs two passes of maximum score search while Bergeron's algorithm performs only one pass.

4 A recursive algorithm

In this section we present a recursive algorithm for SRT when $\mathcal{IN}(A, B) = \emptyset$. This algorithm is similar to an algorithm by Berman and Hannenhali for SBR [4].

Theorem 4 *Let v be an oriented external vertex in H. Let w be a neighbor of v in H. If $w \in \Delta\mathrm{IN}(H, v)$ then $\Delta\mathrm{IN}(H, w) \subset \Delta\mathrm{IN}(H, v)$.*

Proof. Suppose $w \in \Delta\mathrm{IN}(H, v)$. Obviously $\mathrm{CH}(v) = \mathrm{CH}(w)$. Let x be a vertex in H such that $x \notin \Delta\mathrm{IN}(H, v)$. We shall prove that $x \notin \Delta\mathrm{IN}(H, w)$. Let $x = x_0, \ldots, x_k = y$ be a shortest path from x to an external vertex in $H \cdot \rho(v)$. Then in H: x_j is neighbor of v iff x_j is a neighbor of w, for $j = 1..k$.

1. Choose v with maximal score such that $\boldsymbol{ext}[v] = \boldsymbol{o}[v] = 1$.
2. Choose X, Y such that $\boldsymbol{X}[v] = \boldsymbol{Y}[v] = 1$.
3. $\boldsymbol{S1} \leftarrow \boldsymbol{X} \bigwedge \boldsymbol{Y} \bigwedge \boldsymbol{v} \bigwedge \neg \boldsymbol{o}$
4. Build the vector $\boldsymbol{S}$ as follows.
   $\boldsymbol{S}[w] \leftarrow 1$ if the following conditions hold:
   - $\boldsymbol{S1}[w] = 1$ *(conditions 1 and 2)*
   - $(\boldsymbol{w} \bigwedge \boldsymbol{ext}) \bigvee \boldsymbol{v} = \boldsymbol{v}$ *(condition 3)*
   - $(\boldsymbol{v} \bigwedge \neg \boldsymbol{ext}) \bigvee \boldsymbol{w} = \boldsymbol{w}$ *(condition 4)*
   - $(\boldsymbol{v} \bigwedge \boldsymbol{ext} \bigwedge \boldsymbol{o}) \bigvee \boldsymbol{w} = \boldsymbol{w}$ *(condition 5)*
5. If $\boldsymbol{S} \neq 0$ then flip X:
   a. For every u such that $\boldsymbol{X}[u] = 1$:
      i. $\boldsymbol{score} \leftarrow \boldsymbol{score} + \boldsymbol{u}$
      ii. $\boldsymbol{u} \leftarrow \boldsymbol{u} \bigoplus \boldsymbol{X}$
      iii. $\boldsymbol{score} \leftarrow \boldsymbol{score} - \boldsymbol{u}$
   b. Choose v such that $\boldsymbol{S}[v] = 1$ and $\boldsymbol{score}[v]$ is maximal.

(Perform $\rho(v)$ where v is an oriented external vertex)

6. $\boldsymbol{score} \leftarrow \boldsymbol{score} + \boldsymbol{v}$
7. $\boldsymbol{v}[v] = 1$
8. *For every u such that* $\boldsymbol{v}[u] = 1$
   a. *If* $\boldsymbol{ext}[u] = 1$*: then* $\boldsymbol{score} \leftarrow \boldsymbol{score} + \boldsymbol{u}$
      else: $\boldsymbol{score} \leftarrow \boldsymbol{score} - \boldsymbol{u}$
   b. $\boldsymbol{u}[u] = 1$, $\boldsymbol{u} \leftarrow \boldsymbol{u} \bigoplus \boldsymbol{v}$
   c. *If* $\boldsymbol{ext}[u] = 0$*:* $\boldsymbol{X}[u] = 1$, $\boldsymbol{Y}[u] = 1$, $\boldsymbol{ext}[u] = 1$
      Else if $\boldsymbol{X}[u] + \boldsymbol{Y}[v] = 2$*:* $\boldsymbol{X}[u] = 0$, $\boldsymbol{Y}[u] = 0$,$\boldsymbol{ext}[u] = 0$
      Else if $\boldsymbol{X}[u] = 1$*:* $\boldsymbol{X}[u] = 0$, $\boldsymbol{Y}[u] = 1$.
      Else if $\boldsymbol{Y}[u] = 1$*:* $\boldsymbol{Y}[u] = 0$, $\boldsymbol{X}[u] = 1$.
   d. *If* $\boldsymbol{ext}[u] = 1$*:* $\boldsymbol{score} \leftarrow \boldsymbol{score} - \boldsymbol{u}$
      Else: $\boldsymbol{score} \leftarrow \boldsymbol{score} + \boldsymbol{u}$

Fig. 3. A score-based algorithm for performing a safe translocation.

Case 1: w is oriented in H. Then the subgraphs induced by the vertices $\{x_0, \ldots, x_k\}$ in $H \cdot \rho(w)$ $H \cdot \rho(v)$ are the same. Hence in $H \cdot \rho(w)$: y is external and the path in $x = x_0, \ldots, x_k = y$ exists.

Case 2: w is unoriented in H. In $H \cdot \rho(v)$ the vertices in $\{x_0, \ldots, x_{k-1}\}$ are internal and $x_k (= y)$ is external. Therefore $x_j \in \{x_0, \ldots, x_{k-1}\}$ satisfies in H: *(i)* x_j is a neighbor of v iff x_j is external and $\mathrm{CH}(x_j) = \mathrm{CH}(w)$, and *(ii)* x_j is not a neighbor of v iff x_j is internal. Denote by H' the graph obtained from H after flipping one of the chromosomes in $\mathrm{CH}(w)$.

Case 2.a: at least one vertex in $\{x_0, ..., x_{k-1}\}$ is a neighbor of v in H. Choose $x_j \in \{x_0, \ldots, x_{k-1}\}$ a neighbor of v in H such that $\{x_0, \ldots, x_{j-1}\}$ are not neighbors of v in H. Then in H the following conditions are satisfied: *(i)* $x_0, \ldots, x_j$ is a path, *(ii)* all the vertices in $\{x_0, \ldots, x_{j-1}\}$ are internal and *(iii)* x_j is external satisfying $\mathrm{CH}(x_j) = \mathrm{CH}(v)$. Therefore in H' the path $x_0, \ldots, x_j$ still exists and none of the vertices in the path is a neighbor of v (equivalently, w). Hence, the path remains intact in $H' \cdot \rho(w)$.

Case 2.b: none of the vertices in $\{x_0, \ldots, x_{k-1}\}$ is a neighbor of v in H. Then the path $x_0, \ldots, x_k$ exists in H'. v is not a neighbor of w in H' hence v remains external in $H' \cdot \rho(w)$. If x_k is a neighbor of v and w in H' then the path $x_0, \ldots, x_k, v$ exists in $H' \cdot \rho(w)$ and hence $x = x_0 \notin \Delta\mathrm{IN}(H, w)$. If x_k is not a neighbor of v and w in H' then x_k is necessarily external in H' (equivalently, H). Thus none of the subgraphs induced by $\{x_0, \ldots, x_k\}$ in H' and $H' \cdot \rho(w)$ are identical. Hence $x = x_0 \notin \Delta\mathrm{IN}(H, w)$. □

Theorem 5 *If H contains an external vertex then there exists an external vertex v such that $\Delta\mathrm{IN}(H, v) \leq \frac{n-N}{2}$.*

Proof. Let v be an external vertex and assume $\mathrm{CH}(v) = \{X, Y\}$. Let $V_{XY} = \{u : \mathrm{CH}(u) = \{X, Y\}\}$. Let $O_{XY} \subseteq V_{XY}$ be the set of oriented vertices in V_{XY}. We can assume w.l.o.g. that $|O_{XY}| \geq \frac{|V_{XY}|}{2}$ (otherwise we flip X).

Case 1: there are two vertices, $v_1, v_2 \in O_{XY}$, which are not neighbors. Let $M_1 = \Delta\mathrm{IN}(H, v_1)$ and $M_2 = \Delta\mathrm{IN}(H, v_2)$. We shall prove that $M_1 \bigcap M_2 = \emptyset$, and hence $\min\{|M_1|, |M_2|\} \leq \frac{n-N}{2}$. Assume $u \in M_1$ and let $u = u_0, \ldots, u_k = v_1$ be the shortest path from u to v_1 in H. Since v_2 remains intact in $H \cdot \rho(v_1)$ there is no edge from v_2 to any edge in that path. Therefore this path exists in $H \cdot \rho(v_2)$ and hence $u \notin M_2$.

Case 2: the vertices in O_{XY} form a click. Let v be a vertex with maximal score $(|N_{\mathrm{IN}}(v) - N_{\mathrm{EXT}}(v)|)$. Then by Lemma 4, $O_{XY} \bigcap \Delta\mathrm{IN}(H, v) = \emptyset$ and hence $|\Delta\mathrm{IN}(H, v)| \leq |V_{XY} \setminus O_{XY}| \leq \frac{|V_{XY}|}{2} \leq \frac{n-N}{2}$. □

Algorithm *Find_Safe_Translocation_Recursive*

1. $\pi_A \leftarrow$ a concatenation of the chromosomes in A
2. Choose v from $H = H(A, B, \pi_A)$ according to Theorem 5.
3. If $\Delta\mathrm{IN}(H, v) \neq \emptyset$:
   a. $M \leftarrow \Delta\mathrm{IN}(H, v)$
   b. $\mathrm{Genes}(M) \leftarrow \{i : (i, i+1) \in M\}$
   c. Let A_M (respectively, B_M) be the genome accepted from A (respectively, B) after deleting all the genes that do not appear in $\mathrm{Genes}(M)$. Remove common adjacencies of A_M and B_M by deleting one of the genes in each adjacency from both A_M and B_M. Relabel the genes in A_M and B_M such that there is a concatenation of the chromosomes in B_M that is identical to the identity permutation.
   d. $v \leftarrow$ *Find_Safe_Translocation_Recursive*(A_M, B_M)
4. Return v

Fig. 4. A recursive algorithm for locating a safe translocation.

Figure 4 presents a recursive algorithm for SRT that follows from Theorems 4 and 5. Note that in step 3.*d* the two genomes A_M and B_M must be co-tailed

since their cycle graph contains only cycles. We prove below that each call of the algorithm can be implemented in linear time, hence the algorithm is $O(n^2)$.

Computing $\Delta\text{IN}(H, v)$: We use a linear time algorithm by Bader, Moret and Yan [1] for computing the components of an overlap graph. The input for the algorithm is the permutation $\pi_A \cdot \rho(v)$. The *span* of a component M is an interval of genes $I(M) = [i, j] \subset \pi_A$, where $i = arg\min\{\pi_A^{-1}(i_1), \pi_A^{-1}(i_2) \mid (i_1, i_2) \in M\}$ and $j = arg\max\{\pi_A^{-1}(j_1), \pi_A^{-1}(j_2) \mid (j_1, j_2) \in M\}$. Clearly we can compute the spans of all the components in linear time. A component is internal iff the two endpoints of its span belong to the same chromosome of A.

Implementation of step 2: Align the vertices of $G(A, B)$ according to π_A. For v, a vertex in H, denote by $Left(v)$ and $Right(v)$ the left and right endpoints respectively of its corresponding grey edge. Find two chromosomes X and Y such that there exists an external vertex that overlaps both of them. Suppose X is found to the left of Y in π_A. Flip if necessary chromosome Y in π_A to achieve $|O_{XY}| \geq \frac{|V_{XY}|}{2}$. Suppose $O_{XY} = \{v_1, \ldots, v_k\}$, where $Left(v_j) < Left(v_{j+1})$ for $j = 1..k-1$.

If there exist two subsequent vertices v_j and v_{j+1} such that $Right(v_j) > Right(v_{j+1})$, then we found two edges that do not overlap. The computation of $\Delta\text{IN}(H, v_j)$ and $\Delta\text{IN}(H, v_{j+1})$ is as described above. Otherwise, the vertices in O_{XY} form a click. We calculate the score for all the vertices in O_{XY} in linear time in the following way. Let $\{I_1, \ldots, I_k\}$ be a set of intervals forming a click. Let $U = \{J_1, \ldots, J_l\}$ another set of intervals. Let $U(j)$ denote the number of intervals in U which overlap with I_j. There is an algorithm by Kaplan, Shamir and Tarjan [8] that computes $U(j)$, $j = 1..k$ in $O(k + l)$. We use this algorithm twice to compute $|N_{\text{EXT}}(v_j)|$ and $|N_{\text{IN}}(v_j)|$, for $j = 1..k$.

5 Summary

In spite of the fundamental observation of Hannenhalli and Pevzner that translocations can be mimicked by reversals [6], until recently the analyses of SRT and SBR had little in common. Here and in [12] we tighten the connection between the two problems, by presenting a new framework for the study of SRT that builds directly on ideas and theory developed for SBR. Using this framework we show here how to transform two central algorithms for SBR, Bergeron's score-based algorithm and the Berman-Hannenhalli's recursive algorithm, into algorithms for SRT. These new algorithms for SRT maintain the time complexity of the original algorithms for SBR. These results strengthen our understanding of the connection between the two problems. Still, deeper investigation into the relation between SRT and SBR is needed. In particular, providing a reduction from SRT to SBR or vice versa is an open interesting problem.

References

1. D.A. Bader, B. M.E. Moret, and M. Yan. A linear-time algorithm for computing inversion distance between signed permutations with an experimental study. *Journal of Computational Biology*, 8(5):483–491, 2001.

2. A. Bergeron. A very elementary presentation of the Hannenhalli-Pevzner theory. *Discrete Applied Mathematics*, 146(2):134–145, 2005.
3. A. Bergeron, J. Mixtacki, and J. Stoye. On sorting by translocations. *Journal of Computational Biology*, 13(2):567–578, 2006.
4. P. Berman and S. Hannenhalli. Fast sorting by reversal. In Daniel S. Hirschberg and Eugene W. Myers, editors, *Combinatorial Pattern Matching, 7th Annual Symposium*, volume 1075 of *Lecture Notes in Computer Science*, pages 168–185, Laguna Beach, California, 10-12 June 1996. Springer.
5. S. Hannenhalli. Polynomial algorithm for computing translocation distance between genomes. *Discrete Applied Mathematics*, 71:137–151, 1996.
6. S. Hannenhalli and P. Pevzner. Transforming men into mice (polynomial algorithm for genomic distance problems). In *36th Annual Symposium on Foundations of Computer Science (FOCS'95)*, pages 581–592, Los Alamitos, 1995. IEEE Computer Society Press.
7. S. Hannenhalli and P. Pevzner. Transforming cabbage into turnip: Polynomial algorithm for sorting signed permutations by reversals. *Journal of the ACM*, 46:1–27, 1999. (Preliminary version in Proceedings of the Twenty-Seventh Annual ACM Symposium on Theory of Computing 1995 (STOC 95), pages 178–189).
8. H. Kaplan, R. Shamir, and R. E. Tarjan. Faster and simpler algorithm for sorting signed permutations by reversals. *SIAM Journal of Computing*, 29(3):880–892, 2000. (Preliminary version in Proceedings of the eighth annual ACM-SIAM Symposium on Discrete Algorithms 1997 (SODA 97), ACM Press, pages 344–351).
9. H. Kaplan and E. Verbin. Sorting signed permutations by reversals, revisited. *Journal of Computer and System Sciences*, 70(3):321–341, 2005. A preliminary version appeared in *Proc. CPM03*, Springer, 2003, pages 170–185.
10. J. D. Kececioglu and R. Ravi. Of mice and men: Algorithms for evolutionary distances between genomes with translocation. In *Proceedings of the 6th Annual Symposium on Discrete Algorithms*, pages 604–613, New York, NY, USA, January 1995. ACM Press.
11. J. H. Nadeau and B. A. Taylor. Lengths of chromosomal segments conserved since divergence of man and mouse. *Proc Natl Acad Sci U S A*, 81(3):814–818, 1984.
12. M. Ozery-Flato and R. Shamir. An $O(n^{3/2}\sqrt{\log(n)})$ algorithm for sorting by reciprocal translocations. Accepted to CPM 2006. Available at http://www.cs.tau.ac.il/∼ozery/srt_cpm06.pdf.
13. E. Tannier, A. Bergeron, and M. Sagot. Advances on sorting by reversals. *to appear in Discrete Applied Mathematics*.