On Sorting by Translocations

Anne Bergeron[1] Julia Mixtacki[2] Jens Stoye[3]

[1]LaCIM, Université du Québec à Montréal, Canada.
anne@lacim.uqam.ca
[2]International NRW Graduate School in Bioinformatics and Genome Research,
Universität Bielefeld, 33594 Bielefeld, Germany.
mixtacki@cebitec.uni-bielefeld.de
[3]Technische Fakultät, Universität Bielefeld, 33594 Bielefeld, Germany.
stoye@techfak.uni-bielefeld.de

Abstract

The study of genome rearrangements is an important tool in comparative genomics. This paper revisits the problem of sorting a multichromosomal genome by translocations, i.e. exchanges of chromosome ends. We give a formula for computing the translocation distance in linear time, and we give a new algorithm for sorting by translocations, correcting an error in a previous algorithm by Hannenhalli.

1 Introduction

We revisit the problem of sorting multichromosomal genomes by translocations that was introduced by Kececioglu and Ravi [6] and Hannenhalli [4]: Given two genomes A and B, the goal is to find a shortest sequence of exchanges of non-empty chromosome ends that transforms A into B.

Given their prevalence in eukaryotic genomes [3], a good understanding of translocations is necessary. However, the combinatorial theories that underly rearrangement algorithms are complex and prone to human errors [8, 9]. Using tools developed in the context of sorting two genomes by inversions, we establish on solid grounds Hannenhalli's equation for the translocation distance, and give a new algorithm for sorting by translocations.

The next section introduces the basic background. Section 3 presents a new translocation distance formula, and Section 4 discusses the algorithms. Finally, Section 5 concludes the paper.

2 Definitions and examples

2.1 Genes, chromosomes and genomes

We represent a *gene* by a signed integer where the sign represents its orientation. A *chromosome* is a sequence of genes and does not have an orientation. A *genome* is a set of chromosomes. We assume that each gene appears exactly once in a genome. If the k-th chromosome in a genome A of N chromosomes contains m_k genes, then the genes in A are represented by the integers $\{1, \ldots, n\}$ where $n = \sum_{k=1}^{N} m_k$. For example, the following genome consists of three chromosomes and nine genes:

$$A_1 = \{(4 \;\; 3), \quad (1 \quad 2 \quad -7 \quad 5), \quad (6 \quad -8 \quad 9)\}.$$

Since a chromosome does not have an orientation, we can *flip* the chromosome $X = (x_1, x_2, \dots, x_k)$ into $-X = (-x_k, \dots - x_2, -x_1)$ and still have the same chromosome.

A *translocation* transforms the chromosomes $X = (x_1, \dots, x_i, x_{i+1}, \dots, x_k)$ and $Y = (y_1, \dots, y_j, y_{j+1}, \dots, y_l)$ into the new chromosomes $(x_1, \dots, x_i, y_{j+1}, \dots, y_l)$ and $(y_1, \dots, y_j, x_{i+1}, \dots, x_k)$, where $1 \le i < k$ and $1 \le j < l$.

Given a chromosome $X = (x_1, x_2, \dots, x_k)$, the elements x_1 and $-x_k$ are called its *tails*. Since the exchanged chromosome ends are non-empty, genomes A and B must have the same set of tails. In the following, we assume that the elements of each chromosome of the target genome B are positive and in increasing order.

The *sorting by translocations problem* is to find a shortest sequence of translocations that transforms one given genome A into the target genome B. We call the length of such a shortest sequence the *translocation distance* of A, and denote this number by $d(A)$.

Translocations on a genome can be simulated by inversions of intervals of signed permutations, see [5, 8, 9]. For a genome A with N chromosomes, there are $2^N N!$ possible ways to chain the N chromosomes, each of these is called a *concatenation*. Given a concatenation, we extend it by adding a first element 0 and a last element $n+1$. This results in a signed permutation P_A on the set $\{0, \dots, n+1\}$. An *inversion* of an interval reverses the order of the interval while changing the sign of all its elements. See Fig. 1 for an example.

	P_{A_1} = (0 4 3 1 2 -7 5 6 -8 9 10)
A_1 = {(4 3), (1 2 -7 5), (6 -8 9)}	(0 4 3 -5 7 -2 -1 6 -8 9 10)
{(4 -7 5), (1 2 3), (-9 8 -6)}	(0 4 -7 5 -3 -2 -1 6 -8 9 10)
	(0 4 -7 -6 1 2 3 -5 -8 9 10)
{(4 -7 -6), (1 2 3), (-5 -8 9)}	(0 4 -7 -6 1 2 3 -9 8 5 10)
	(0 4 9 -3 -2 -1 6 7 8 5 10)
{(-9 -4), (1 2 3), (-5 -8 -7 -6)}	(0 -9 -4 -3 -2 -1 6 7 8 5 10)
	(0 -9 -8 -7 -6 1 2 3 4 5 10)
	(0 -5 -4 -3 -2 -1 6 7 8 9 10)
B_1 = {(1 2 3), (4 5), (6 7 8 9)}	Id = (0 1 2 3 4 5 6 7 8 9 10)

Figure 1: *Left:* An optimal sorting scenario for the translocation distance problem for the genomes A_1 and B_1; the exchanged chromosome ends are underlined. *Right:* Given an arbitrary concatenation, the problem can be modeled by transforming the signed permutation P_{A_1} into the identity permutation (Id) by inversions; solid lines denote inversions that represent translocations, dashed lines denote inversions that flip chromosomes.

2.2 Elementary intervals and cycles

Let A be a genome on the set $\{1, \dots, n\}$. We consider the extended signed permutation P_A defined by an arbitrary concatenation of the chromosomes.

Definition 1 *A pair $p \cdot q$ of consecutive elements in a signed permutation is called a* point. *A point is called an* adjacency *if it is a point of the form $i \cdot i+1$ or $-(i+1) \cdot -i$, $0 \le i \le n$, otherwise it is called a* breakpoint.

For example, the signed permutation P_{A_1} has ten points. The points $1 \cdot 2$, $5 \cdot 6$ and $9 \cdot 10$ are adjacencies, and all the other points are breakpoints.

$$P_{A_1} = (0 \underset{\circ}{\;} 4 \underset{\bullet}{\;} 3 \underset{\circ}{\;} 1 \underset{\bullet}{\;} 2 \underset{\bullet}{\;} \text{-}7 \underset{\bullet}{\;} 5 \underset{\circ}{\;} 6 \underset{\bullet}{\;} \text{-}8 \underset{\bullet}{\;} 9 \underset{\circ}{\;} 10)$$

The points inside chromosomes are called *black* points, otherwise *white*. When sorting, eventually all black points must become adjacencies. The *right*, or *left*, point of an element of P_A is the point immediately to its right, or left.

Definition 2 *For each pair of unsigned elements* $(k, k+1)$, $0 \leq k < n+1$, *define the* elementary interval I_k *associated to the pair* $k \cdot k+1$ *of unsigned elements to be the interval whose endpoints are:*

1. *the right point of* k, *if* k *is positive, otherwise its left point;*
2. *the left point of* $k+1$, *if* $k+1$ *is positive, otherwise its right point.*

Since we assume that genomes have the same set of tails and that the elements of the target genome are positive and in sorted order, the two endpoints of an elementary interval will always be either both black or both white. From Defintion 2 it follows that exactly two elementary intervals of the same color meet at each breakpoint.

Definition 3 *A black (or white)* cycle *is a sequence of breakpoints that are linked by black (respectively white) elementary intervals. Adjacencies define* trivial *cycles.*

The elementary intervals and cycles of the permutation P_{A_1} are shown in Fig. 2.

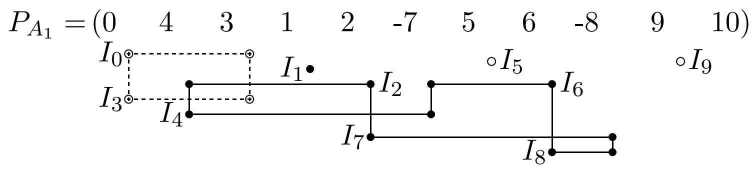

Figure 2: The signed permutation P_{A_1} has five cycles; two of them are black and all the others are white cycles.

We are interested in how a translocation changes the number of breakpoints, as well as the number of cycles.

Lemma 1 (Kececioglu and Ravi [6]) *A translocation in genome A modifies the number of black cycles of* P_A *by* 1, 0 *or* -1.

Following the terminology of Hannenhalli [4], a translocation is called *proper* if it increases the number of black cycles by 1, *improper* if it leaves the number of black cycles unchanged and *bad* if it decreases the number of black cycles by 1. An elementary interval whose endpoints belong to different chromosomes is called *interchromosomal*, otherwise it is called *intrachromosomal*.

Lemma 2 (Bergeron *et al.* [2]) *For each interchromosomal elementary interval in* P_A, *there exists a proper translocation in the genome A.*

2.3 Intrachromosomal components

Elementary intervals and cycles can be grouped into higher structures:

Definition 4 *A* component *of a signed permutation is an interval from element i to element $i+j$ or from $-(i+j)$ to $-i$, where $j > 0$, whose set of unsigned elements is $\{i, \ldots, i+j\}$, and that is not the union of smaller such intervals.*

We refer to a component by giving its first and last elements such as $[i \ldots j]$. When the elements of a component belong to the same chromosome, then the component is said to be *intrachromosomal.* An intrachromosomal component is called *minimal* if it does not contain any other intrachromosomal component. An intrachromosomal component that is an adjacency is called *trivial*, otherwise *non-trivial.* For example, consider the genome

$$A_2 = \{(\,1 \quad -2 \quad 3 \quad 8 \quad 4 \quad -5 \quad 6),\ (\,7 \quad 9 \quad -10 \quad 11 \quad -12 \quad 13 \quad 14 \quad -15 \quad 16)\}.$$

The signed permutation P_{A_2} has six intrachromosomal components; all of them are minimal and all except $[13 \ldots 14]$ are non-trivial. They can be represented by a boxed diagram such as in Fig. 3. Note that $[3 \ldots 9]$ is a component that is not intrachromosomal.

When two intrachromosomal components overlap on one element, we say that they are *linked.* Successive linked intrachromosomal components form a *chain.* A chain that cannot be extended to the left or right is called *maximal.*

Definition 5 *Given a chromosome X and its intrachromosomal components, we define a forest F_X by the following construction:*

1. *Each non-trivial intrachromosomal component is represented by a round node.*
2. *Each maximal chain that contains non-trivial components is represented by a square node whose (ordered) children are the round nodes that represent the non-trivial intrachromosomal components of this chain.*
3. *A square node is the child of the smallest intrachromosomal component that contains this chain.*

We extend the above definition to a forest of a genome by combining the forests of all chromosomes:

Definition 6 *Given a genome A consisting of chromosomes $\{X_1, X_2, \ldots, X_N\}$. The forest F_A is the set of forests $\{F_{X_1}, F_{X_2}, \ldots, F_{X_N}\}$.*

Note that the forest F_A can consist of more than one tree in contrast to the unichromosomal case [1]. Figure 3 shows the forest F_{A_2} that consists of three trees.

When sorting a genome, eventually all its non-trivial intrachromosomal components, and hence all its trees, are *destroyed.* We say that a translocation destroys an intrachromosomal component C if C is not an intrachromosomal component in the resulting genome. At most two minimal intrachromosomal components on two different chromosomes, plus all intrachromosomal components containing these two components, can be destroyed by a single translocation.

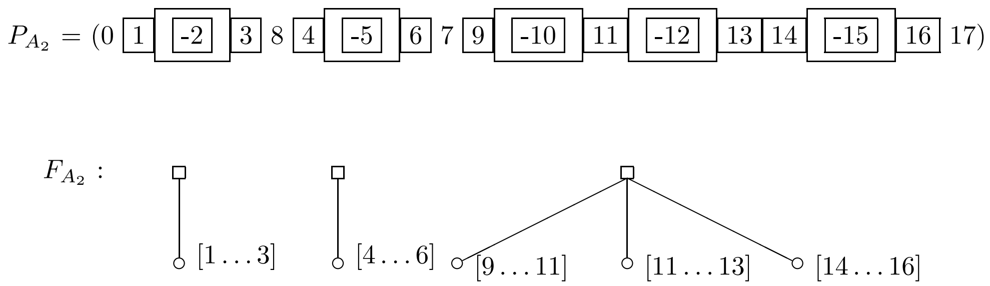

Figure 3: The intrachromosomal components of the signed permutation P_{A_2} of the genome $A_2 = \{(1 -2\ \ 3\ \ 8\ \ 4 -5\ \ 6), (7\ \ 9 -10\ \ 11 -12\ \ 13\ \ 14 -15\ \ 16)\}$ and the forest F_{A_2}.

3 Computing the translocation distance

The following lemma is a central observation in proving a new distance formula and establishing an invariant for the sorting algorithm.

Lemma 3 (Bergeron *et al.* [2]) *Let A be a genome whose forest has L leaves and T trees. If L is even, and $T > 1$, then there always exists a sequence of proper translocations, followed by a bad translocation, such that the resulting genome A' has $L' = L - 2$ leaves and $T' \neq 1$ trees.*

Lemma 3 implies that when the number of leaves is even, and $T > 1$, we can always destroy the forest optimally: we can use proper translocations to separate the forest, and then remove two leaves with a bad translocation. Eventually, all trees are destroyed, i.e. $T = 0$. The basic idea of the following theorem is to reduce all other cases to the simple case of Lemma 3.

Theorem 1 (Bergeron *et al.* [2]) *Let A be a genome with c black cycles and F_A be the forest associated to A. Then*

$$d(A) = n - N - c + t$$

where

$$t = \begin{cases} L + 2 & \text{if } L \text{ is even and } T = 1 \\ L + 1 & \text{if } L \text{ is odd} \\ L & \text{if } L \text{ is even and } T \neq 1. \end{cases}$$

For example, the genome A_2 of Section 2.3 consists of two chromosomes and 16 elements. The signed permutation P_{A_2} has seven black cycles. The forest F_{A_2} has three trees and five leaves (see Fig. 3). Therefore, we have

$$d(A_2) = n - N - c + t = 16 - 2 - 7 + 6 = 13.$$

4 Algorithms

Using the formula of Theorem 1, the translocation distance between two genomes can be computed in linear time, a result previously given by Li *et al.* [7]. The algorithm to compute the translocation distance is similar to the one to compute the reversal distance presented in [1].

Turning to the sorting by translocations problem, we show an algorithm that sorts a genome optimally in Algorithm 1.

Initially, one or two translocations are applied in order to arrive at the preconditions of Lemma 3. If the forest consists of a single tree with an even number of leaves (line 2), any one leaf is destroyed. In the resulting genome, if the number of leaves is odd and in a single tree, its middle leaf is destroyed, i.e. $T > 1$. In the case where the number of leaves is odd and there is more than one tree, a translocation that destroys one leaf of the greatest tree is applied. In all cases, a genome A' is obtained with $T' = 0$, or $T' > 1$ and L' even.

Then, as long as there exist intrachromosomal components (i.e. $T > 1$ and L is even), the forest can be optimally destroyed as described after Lemma 3. Once all intrachromosomal components are destroyed (i.e. $T = 0$), the genome can be sorted using proper translocations that do not create new non-trivial intrachromosomal components.

Theorem 2 *Algorithm 1 solves the sorting by translocations problem in $O(n^3)$ time.*

Initially, the forest F_A associated to a genome A is constructed. This can be done in $\mathcal{O}(n)$ time as discussed above. The algorithm requires at most $\mathcal{O}(n)$ iterations. The bad translocations of line 9 can be found in constant time. Since there are $\mathcal{O}(n)$ proper translocations and each translocation requires the construction of the forest to verify the condition $T' = T$ and $L' = L$, the search for a proper translocation in line 11 takes $\mathcal{O}(n^2)$ time. Hence, the total time complexity of Algorithm 1 is $\mathcal{O}(n^3)$.

Algorithm 1 (Sorting by translocations algorithm)

1: L is the number of leaves, and T the number of trees in the forest F_A associated to the genome A
2: **if** L is even **and** $T = 1$ **then**
3:    destroy one leaf such that $L' = L - 1$
4: **end if**
5: **if** L is odd **then**
6:    perform a bad translocation such that $T' = 0$, or $T' > 1$ and $L' = L - 1$
7: **end if**
8: **while** A is not sorted **do**
9:    **if** there exist intrachromosomal components on different chromosomes **then**
10:       perform a bad translocation such that $T' = 0$, or $T' > 1$ and L' is even
11:    **else**
12:       perform a proper translocation such that T and L remain unchanged
13:    **end if**
14: **end while**

5 Conclusion

The real challenge in developing genome rearrangement algorithms is to propose algorithms whose validity can be checked, both mathematically and biologically. The most useful set of rearrangement operations is currently including translocations, fusions, fissions and inversions [9]. Unfortunately, we think that few people are able to assess the mathematical validity of the current algorithms. The work we have done in this paper opens the way to simpler description and implementation of such algorithms.

References

[1] A. Bergeron, J. Mixtacki, and J. Stoye. Reversal distance without hurdles and fortresses. In *CPM 2004 Proceedings*, volume 3109 of *LNCS*, pages 388–399. Springer Verlag, 2004.

[2] A. Bergeron, J. Mixtacki, and J. Stoye. On Sorting by Translocations. In *RECOMB 2005 Proceedings*, accepted for publication.

[3] Mouse Genome Sequencing Consortium. Initial sequencing and comparative analysis of the mouse genome. *Nature*, 420:520–562, 2002.

[4] S. Hannenhalli. Polynomial-time algorithm for computing translocation distance between genomes. *Discrete Appl. Math.*, 71(1-3):137–151, 1996.

[5] S. Hannenhalli and P. A. Pevzner. Transforming men into mice (polynomial algorithm for genomic distance problem). In *FOCS 1995 Proceedings*, pages 581–592. IEEE Press, 1995.

[6] J. D. Kececioglu and R. Ravi. Of mice and men: Algorithms for evolutionary distances between genomes with translocation. In *Proceedings of the sixth ACM-SIAM Symposium on Discrete Algorithm*, pages 604–613. Society of Industrial and Applied Mathematics, 1995.

[7] G. Li, X Qi, X. Wang, and B. Zhu. A linear-time algorithm for computing translocation distance between signed genomes. In *CPM 2004 Proceedings*, volume 3109 of *LNCS*, pages 323–332. Springer Verlag, 2004.

[8] M. Ozery-Flato and R. Shamir. Two notes on genome rearrangements. *J. Bioinf. Comput. Biol.*, 1(1):71–94, 2003.

[9] G. Tesler. Efficient algorithms for multichromosomal genome rearrangements. *J. Comput. Syst. Sci.*, 65(3):587–609, 2002.