Genome Rearrangement Phylogeny using the Single Cut or Join operation

Pedro C. Feijão *João Meidanis*

Technical Report - IC-09-030 - Relatório Técnico

September - 2009 - Setembro

Genome Rearrangement Phylogeny using the Single-Cut-or-Join operation

Pedro Cipriano Feijão[*] João Meidanis[†]

Abstract

The problem of parsimonious phylogeny reconstruction using genome rearrangement is called Multiple Genome Rearrangement Problem. There are two usual approaches: the small phylogeny problem, where a tree is given and we want to find the ancestral nodes that minimize total branch length of the tree; and the big phylogeny problem, finding the tree and ancestral nodes with minimum total branch length. In this paper we show that, under the Single-Cut-or-Join metric, the small phylogeny problem is solvable in polynomial time while the big phylogeny problem is NP-hard.

1 Introduction

For many years the field of phylogenetic reconstruction has been dominated by techniques based on alignments of sequences of one or more orthologous genes and proteins. Due to recent progress in genome sequencing, more data for phylogenetic reconstruction based on rearrangement distances between genomes is becoming available at a very fast rate. However, this phylogenetic reconstruction is a very challenging task. For the most simple distance measures (the breakpoint distance and the reversal distance), the problem is NP-hard even if one considers only three genomes.

Most of the research in parsimonious phylogenetic reconstruction based on rearrangement distances is based on the *multiple genome rearrangement problem* (MGRP), searching phylogenetic tree describing the most "plausible" rearrangement scenario for multiple genomes [11, 16]. Formally, given n genomes, find a tree T with the n genomes as leaf nodes and assign ancestral genomes to internal nodes of T such that the tree is optimal, i.e., the sum of rearrangement distances over all edges of the tree is minimal. This problem is also called the *big phylogeny problem* (BPP). If the tree is given, one needs to find only the internal nodes, and it becomes an instance of the *small phylogeny problem* (SPP). The special case for only three genomes is known as the *genome median problem* (GMP): given three genomes, find a fourth genome (to label the internal node connecting the three leaves) that minimizes the sum of its distances to the three given genomes. Even this smaller version is NP-hard for most rearrangement distances [19].

[*]Institute of Computing, University of Campinas, 13081-970 Campinas, São Paulo, Brazil.

[†]Scylla Bioinformatics, Campinas, São Paulo, Brazil and Institute of Computing, University of Campinas, 13081-970 Campinas, São Paulo, Brazil.

The first approach to solving the MGRP was proposed by Blanchette et al. [6] and was called *breakpoint phylogeny*, where they studied the SPP under the breakpoint distance (BP). They developed an iterative algorithm to solve the SPP where one iterates over each internal node of the given tree, solving a GMP on this node until convergence to a local minimum is achieved. This approach is used by most algorithms to solve the SPP to this day. On a follow-up paper, Sankoff and Blanchette developed a method for the BPP called BPAnalysis [15]. This method performs an extensive search over all possible tree topologies and solves the SPP on each one. Since the number of topologies is exponential on the number of genomes, this method was restricted to very small instances.

Later, Moret et al. developed an faster alternate method called GRAPPA [14], based on BPAnalysis, that improved the speed in several orders of magnitude. Also, with the availability of a linear algorithm for inversion distance [2], the breakpoint distance was replaced by the inversion distance [13]. Also, using the Disk-Covering method [12], their method was scaled to solve the BPP up to thousands of genomes [17].

Another approach to the MGRP using reversal was presented by Bourque and Pevzner in their program called MGR [7]. The main difference with GRAPPA is that the GMP is not solved exactly, but an faster heuristic is applied.

Since the MGR under the reversal distance can have multiple solutions, Bernt et al. [5] presented a software tool called amGRP which further significantly improves upon GRAPPA and MGR by storing not only a single optimal median, but a set of optimal medians after solving the GMP at each internal node.

Recently, other rearrangement distances were used to solve the MGRP. Adam and Sankoff developed an heuristic to the GMP for the Double Cut and Join (DCJ) distance [20] and used it in their algorithm for the SPP [1], using the iterative method of solving the GMP for each internal node until convergence is achieved. Bader et al. [3] presented an heuristic for the MGRP using weighted reversals and transpositions using different weight ratios.

In this article we study the MGRP problem under the Single-Cut-or-Join (SCJ) distance [8]. We will show that the SPP is polynomial, the first polynomial algorithm known for the SPP, and prove that the BPP is NP-hard. In Section 2 we introduce the genome formulation and basic definitions used throughout the paper. In Section 3 we solve the SPP polynomially and in Section 4 show that the BPP is NP-hard under the SCJ distance. Finally, in Section 5, we present our concluding remarks and directions for future investigations.

2 Genome Formulation and Definitions

We will use a standard genome formulation [4, 18]. A *gene* is an oriented sequence of DNA that starts with a tail and ends with a head, called the *extremities* of the gene. The tail of a gene a is denoted by a_t, and its head by a_h. Given a set of genes $\mathcal{G}$, the extremity set is $\mathcal{E} = \{a_t : a \in \mathcal{G}\} \cup \{a_h : a \in \mathcal{G}\}$. An *adjacency* is an unordered pair of two extremities that represents the linkage between two consecutive genes in a certain orientation on a chromosome, for instance $a_h b_t$. An extremity that is not adjacent to any other extremity is

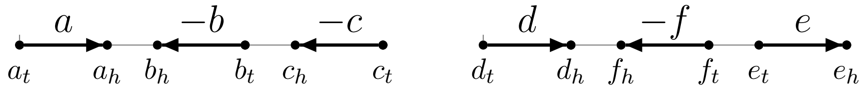

Figure 1: Graph G_π representing a genome with two linear chromosomes. Black directed edges represent genes, while grey edges link consecutive extremities.

called a *telomere*. A genome is represented by a set of adjacencies where the tail and head of each gene appear at most once. Telomeres will be omitted in our representation, since they are uniquely determined by the set of adjacencies and the extremity set $\mathcal{E}$. Two adjacencies are *conflicting* when they share a common extremity. Two conflicting adjacencies cannot belong to the same genome.

The *graph representation* of a genome π is a graph G_π whose vertices are the extremities of π and there is a grey edge connecting the extremities x and y when xy is an adjacency of π or a directed black edge if x and y are head and tail of the same gene. A connected component in G_π is a *chromosome* of π, and it is *linear* if it is a path, and *circular* if it is a cycle. A *circular genome* is a genome whose chromosomes are all circular, and a *linear genome* is a genome whose chromosomes are all linear. A *string representation* of a genome π, denoted by π_S, is a set of strings corresponding to the genes of π in the order they appear on each chromosome, with a bar over the gene if it is read from head to tail and no bar otherwise. Notice that the string representation is not unique: each chromosome can be replaced by its reverse complement.

For instance, given the set $\mathcal{G} = \{a, b, c, d, e, f\}$, and the genome $\pi = \{a_h b_h,\ b_t c_h, d_h f_h,$ $f_t e_t\}$, the graph G_π is given in Figure 1. Notice that telomeres a_t, c_t, d_t, and e_h are omitted from the set representation without any ambiguity. A string representation of this genome is $\pi_S = \left(a\ \bar{b}\ \bar{c}\ ,\ d\ \bar{f}\ e\right)$.

Since each genome is a set of adjacencies, standard set operations such as union, intersection and set difference can be applied to two (or more) genomes. In the case of intersection and set difference, the result is a set of adjacencies contained in at least one of the genomes, and therefore it is also a genome. On the other hand, the set resulting from a union operation might not represent a genome since the same extremity could be present in more than one adjacency. We will use these operations throughout this paper in our algorithms, and whenever union is used, we will prove that the resulting set represents a valid genome. Set difference between sets A and B will be denoted by $A - B$.

A *Single-Cut-or-Join* (SCJ) is a rearrangement operation defined by Feijão and Meidanis [8] that either breaks an adjacency in two telomeres (namely, its extremities) — a *cut* — or pairs two telomeres into an adjacency — the reverse operation, called *join*. The distance bewteen two genomes π and σ is easily calculated with the equation [8]:

$$d(\pi, \sigma) = |\pi - \sigma| + |\sigma - \pi| = |\pi| + |\sigma| - 2|\pi \cap \sigma| \tag{1}$$

3 Small Phylogeny Problem under SCJ

Given a group of n genomes $\pi_1, \ldots, \pi_n$ defined on the same set of genes $\mathcal{G}$ and given a tree T, where each leaf of T corresponds to a genome, the *small phylogeny problem* (SPP) consists in finding ancestral genomes $\mu_1, \dots, \mu_m$ corresponding to the m internal nodes of T such that the total branch length of T (the sum of the weight of each edge, defined as the distance between the genomes of its vertices) is minimized. Formally, we want to find

$$M = \min_{\mu_1,\dots,\mu_m} \sum_{e \in E(T)} d\big(v_1(e), v_2(e)\big) \tag{2}$$

where $E(T)$ is the set of edges of T and $v_1(e), v_2(e)$ are the vertices adjacent to edge e.

The usual way to solve this problem is the approach proposed by Blanchette et al. [6], where one iterates over each internal node of T, solving a *genome median problem* (GMP) — finding the genome that minimizes the sum of the distances to its three neighbours — until convergence to a local minimum is achieved. This approach has been used with some rearrangement distances, such as BP [6, 7], reversal [13] and SCJ [1] distances. One difficulty of using this approach is that the GMP is NP-hard on most rearrangement distances, except for the SCJ distance [8] and the BP distance on some specific cases [19]. Since the GMP is easy under the SCJ distance, we could use the same approach, but we will show the stronger result that the SPP has a polynomial solution under the SCJ distance, the first polynomial result for this problem under any proposed rearrangement distance.

In the SCJ model we can think of each adjacency as a character, and we have unitary cost of including or removing an adjacency — changing a character. Using this analogy, we can solve the SPP running Fitch's algorithm for small parsimony [9] for every adjacency present on at least one of the genomes considered, deciding which ancestral genomes contain the adjacency. When each character is independent, to infer the characters present in each ancestral genome is only a matter of running Fitch's algorithm for each charater and joining the results. In our case, however, the characters are not independent, as conflicting adjacencies cannot belong simultaneously to the same genome. Nevertheless, this very strategy generates valid ancestral genomes in polynomial time and is indeed optimal, as we will see in Theorem 1.

Some additional definitions related to Fitch's algorithm must be made before we proceed. In the first part of the algorithm, every internal node is assigned a set depending on the sets of its children, in a bottom-up way. We will call these sets *bottom-up sets*. Furthermore, given an adjacency d and a rooted tree T whose leaves correspond to the genomes being analysed, consider the result of running Fitch's algorithm on tree T, using as binary characteristic the presence of adjacency of d, and initializing the root with zero (absence) if its bottom-up set is $\{0, 1\}$. We denote by $B(d, n)$ the bottom-up set generated for node n of T in the first (bottom-up) pass of the algorithm, and by $F(d, n)$ the final character assignment to node n.

Before we prove the main theorem we will need the results of two preliminary lemmas.

Lemma 1. *Given two conflicting adjacencies d and e, we have the following result: for each node v of T, if $B(d, v) = \{1\}$, then $B(e, v) = \{0\}$; and, if $B(d, v) = \{0, 1\}$, then $B(e, v) \neq \{1\}$.*

Proof. We will prove both results simultaneously by strong induction on h, the height of an internal node v, defined as the maximum length of a path from v to any of its descendant leaves.

When $h = 0$, the node is a leaf, and both $B(d, v)$ and $B(e, v)$ are singletons. If $B(d, v) = \{1\}$, the corresponding genome contains adjacency d, and therefore does not contain adjacency e, since they are conflicting. Therefore $B(e, v) = \{0\}$. Notice that $B(d, v) \neq \{0, 1\}$ because v is a leaf. The property is therefore valid for nodes with $h = 0$.

For an internal node v with height $k \geq 1$, assume by induction that any node with height $h < k$ satisfies the properties. By definition of height, both children of v have heights strictly smaller than k, and therefore satisfy the properties. To prove the properties for v, observe than there are two ways for $B(d, v)$ to be $\{1\}$, shown in Figure 2. In case (a), both children have sets $\{1\}$ with respect to d. Therefore, by the induction hypothesis, both children have sets $\{0\}$ with respect to e, implying that $B(e, v) = \{0\}$. In case (b), one child has set $\{1\}$ and the other has set $\{0, 1\}$ with respect to d. Then, with respect to e, one child has set $\{0\}$ and the other may have set $\{0\}$ or $\{0, 1\}$, again implying that $B(e, v) = \{0\}$.

Using a similar reasoning, we see that when $B(d, v) = \{0, 1\}$, there are two possible cases, shown on Figure 3, and in both cases $B(e, v)$ cannot be $\{1\}$. The lemma is proved. □

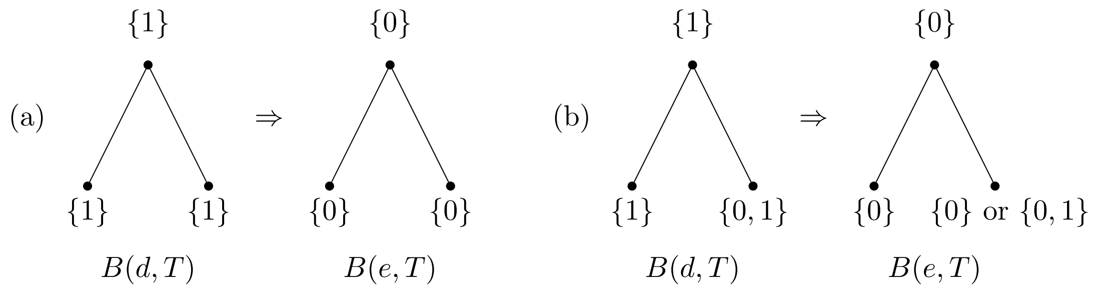

Figure 2: Possible cases for an internal node with bottom-up set $\{1\}$, where d and e are conflicting adjacencies.

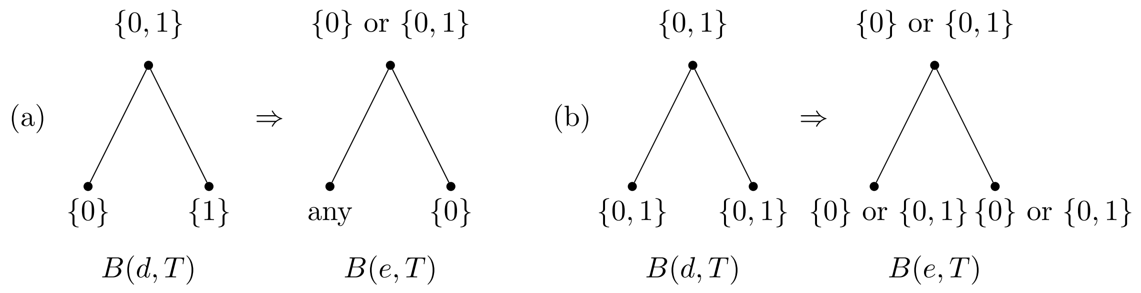

Figure 3: Possible cases for an internal node with bottom-up set $\{0, 1\}$, where d and e are conflicting adjacencies.

Lemma 2. *Given two conflicting adjacencies d and e, for every node v of T, if $F(d, v) = 1$ then $F(e, v) = 0$.*

Proof. The proof is by contradiction. Of course the result is true when v is a leaf. Suppose then that there are internal nodes with value 1 with respect to both d and e. Choose such a node with minimum depth (distance from root to node) and call it n. Since $F(d,v)=1$, we have that $B(d,v)$ is either $\{1\}$ or $\{0,1\}$. It cannot be $\{1\}$, otherwise Lemma 1 would imply that $B(e,v)=\{0\}$, contradicting the fact that $F(e,v)=1$. Therefore, $B(d,v)=\{0,1\}$. The same reasoning applies exchanging d and e, implying that $B(e,v)=\{0,1\}$ as well.

Notice now that v cannot be the root of the tree, otherwise $F(d,v)=F(e,v)=0$ by the way we set up Fitch's algorithm to decide ties at the root. Since v is not the root, its parent node p must have $F(d,p)=1$ and $F(e,p)=1$, contradicting the minimality of v's depth, and concluding the proof.

□

Theorem 1. *Consider a rooted tree T whose leafs correspond to genomes over the same set of genes $\mathcal{G}$. Then the sets $B(v)=\{d : F(d,v)=1\}$, where v is an internal node v of T, are valid genomes and assigning $B(v)$ to node v minimizes the total SCJ branch length of the tree T.*

Proof. From the definition of SCJ distance, we derive a distance equation where the contribution of each adjacency is independent:

$$d(\pi_1,\pi_2)=|\pi_1|+|\pi_1|-2|\pi_1\cap\pi_2|=\sum_{d\in D}\delta_d(\pi_1)+\delta_d(\pi_2)-2\delta_d(\pi_1\cap\pi_2) \tag{3}$$

where D is the set of all adjacencies and $\delta_d(\pi)$ is defined as

$$\delta_d(\pi)=\begin{cases}1, & d\in\pi\\ 0, & d\notin\pi\end{cases}$$

We want to minimize the total branch lenght of the tree,

$$\begin{aligned}
M &= \min_{\mu_1,\dots,\mu_n}\sum_{e\in E(T)} d\big(v_1(e),v_2(e)\big)\\
&= \min_{\mu_1,\dots,\mu_n}\sum_{e\in E(T)}\sum_{d\in D}\delta_d(v_1(e))+\delta_d(v_2(e))-2\delta_d(v_1(e)\cap v_2(e))\\
&= \min_{\mu_1,\dots,\mu_n}\sum_{d\in D}\sum_{e\in E(T)}\delta_d(v_1(e))+\delta_d(v_2(e))-2\delta_d(v_1(e)\cap v_2(e))\\
&= \sum_{d\in D}\left(\min_{\mu_1,\dots,\mu_n}\sum_{e\in E(T)}\delta_d(v_1(e))+\delta_d(v_2(e))-2\delta_d(v_1(e)\cap v_2(e))\right)
\end{aligned}$$

We know that given an adjacency d, the minimum

$$\min_{\mu_1,\dots,\mu_n}\sum_{e\in E(T)}\delta_d(v_1(e))+\delta_d(v_2(e))-2\delta_d(v_1(e)\cap v_2(e))$$

is achieved including the adjacency d in every node v where $F(d,v)=1$. Repeating this procedure for each adjacency we build the genomes $B(v)$, which are valid because no pair of conflicting adjacencies can belong to the same genome, as a result from Lemma 2. Therefore, setting the genomes $B(v)$ to each node v minimizes the sum of all branch lengths.

□

4 Big Phylogeny Problem under SCJ

The big phylogeny problem under SCJ can be stated as follows. Given genomes n genomes $\pi_1, \ldots, \pi_n$ defined on the same set of genes $\mathcal{G}$, find a tree T whose leafs are in one-to-one correspondence with the genomes $\pi_1, \ldots, \pi_n$, and find ancestral genomes $\mu_1, \ldots, \mu_m$ corresponding to the m internal nodes of T so that the total branch length of T (the sum of the weight of each edge, defined as the distance between the genomes of its vertices) is minimized.

Again, we can think of each adjacency as a character, and we have unitary cost of including or removing an adjacency — changing a character. Using this analogy, if the characters were independent, we would have an instance of the Steiner tree problem in $\{0,1\}^N$, which is NP-hard [10]. In our case, the characters are not necessarily independent, as conflicting adjacencies cannot belong simultaneously to the same genome. However, given an instance of the Steiner problem in $\{0,1\}^N$, it is possible to code it as an SCJ problem where the adjacencies behave as independent characters for our purposes, thus effectively reducing this NP-hard problem to big phylogeny under SCJ. As a result, SCJ big phylogeny turns out to be NP-hard as well.

The coding is simple. Given n $\{0,1\}$ vectors of size N, consider the set of genes $\mathcal{G} = \{g^1, g^2, ..., g^N\}$ and code a vector $v = (v_1, v_2, \ldots, v_N)$ as a genome having adjacencies:

$$C(v) = \{g_h^i g_t^{i+1} : v_i = 1\},$$

where $C(v)$ denotes the genome coding vector v, and the sum $i+1$ “wraps around”, that is, when it becomes $N+1$ we assume the value is in fact 1. Notice that, besides adjacencies being nonconflicting within a genome, they remain pairwise nonconflicting across all genomes considered.

Now think about the big phylogeny under SCJ instance with these genomes. A solution to this problem will contain only adjacencies present in at least one of the input genomes, since any other adjacency can be safely removed throughout without increasing the total distance. Therefore, the solution can be translated back to $\{0,1\}^N$ vectors, that is, for each genome π at an internal node there will be a vector v such that $C(v) = \pi$. This yields a solution to the original Steiner problem, because the coding C preserves distances (Hamming distance in the origin, SCJ distance in the destination).

5 Conclusion and Future Directions

The MGRP under the SCJ distance is a much easier problem than in any other rearrangement distance. In fact, it is the only distance that the SPP has a polynomial distance.

The next direction will be the implementation of algorithms for the MGRP and testing on known datasets such as Campanulaceae Chloroplast DNA, Metazoan MtDNA and Mammals, and also on simulated data.

References

[1] Adam, Z., and Sankoff, D. The ABCs of MGR with DCJ. *Evol Bioinform Online 4* (2008), 69–74.

[2] Bader, D. A., Moret, B. M., and Yan, M. A linear-time algorithm for computing inversion distance between signed permutations with an experimental study. *J Comput Biol 8*, 5 (2001), 483–491.

[3] Bader, M., Abouelhoda, M. I., and Ohlebusch, E. A fast algorithm for the multiple genome rearrangement problem with weighted reversals and transpositions. *BMC Bioinformatics 9* (2008), 516.

[4] Bergeron, A., Mixtacki, J., and Stoye, J. A unified view of genome rearrangements. In *Proc. WABI 06* (2006), vol. 4175 of *LNCS*, pp. 163–173.

[5] Bernt, M., Merkle, D., and Middendorf, M. Using median sets for inferring phylogenetic trees. *Bioinformatics 23*, 2 (Jan 2007), e129–e135.

[6] Blanchette, M., Bourque, G., and Sankoff, D. Breakpoint phylogenies. *Genome Inform Ser Workshop Genome Inform 8* (1997), 25–34.

[7] Bourque, G., and Pevzner, P. A. Genome-scale evolution: reconstructing gene orders in the ancestral species. *Genome Res 12*, 1 (Jan 2002), 26–36.

[8] Feijão, P., and Meidanis, J. SCJ:a variant of breakpoint distance for which sorting, genome median and genome halving problems are easy. To appear in 9th Workshop on Algorithms in Bioinformatics (WABI 2009), 2009.

[9] Fitch, W. M. Toward defining the course of evolution: Minimum change for a specific tree topology. *Systematic Zoology 20* (1971), 406–416.

[10] Foulds, L. R., and Graham, R. L. The Steiner problem in phylogeny is NP-complete. *Adv. Applied Math. 3* (1982), 43–49.

[11] Hannenhalli, S., Chappey, C., Koonin, E. V., and Pevzner, P. A. Genome sequence comparison and scenarios for gene rearrangements: A test case. *Genomics 30*, 2 (1995), 299–311.

[12] Huson, D. H., Nettles, S. M., and Warnow, T. J. Disk-covering, a fast-converging method for phylogenetic tree reconstruction. *J Comput Biol 6*, 3-4 (1999), 369–386.

[13] Moret, B. M., Siepel, A. C., Tang, J., and Liu1, T. Inversion medians outperform breakpoint medians in phylogeny reconstruction from gene-order data. In *Proc. WABI 2002* (2002), vol. 2452 of *LNCS*, pp. 521–536.

[14] Moret, B. M., Wang, L. S., Warnow, T., and Wyman, S. K. New approaches for reconstructing phylogenies from gene order data. *Bioinformatics 17 Suppl 1* (2001), S165–S173.

[15] Sankoff, D., and Blanchette, M. Multiple genome rearrangement and breakpoint phylogeny. *J Comput Biol 5*, 3 (1998), 555–570.

[16] Sankoff, D., Sundaram, G., and Kececioglu, J. D. Steiner points in the space of genome rearrangements. *International Journal of Foundations of Computer Science 7*, 1 (1996), 1–9.

[17] Tang, J., and Moret, B. M. E. Scaling up accurate phylogenetic reconstruction from gene-order data. *Bioinformatics 19 Suppl 1* (2003), i305–i312.

[18] Tannier, E., Zheng, C., and Sankoff, D. Multichromosomal genome median and halving problems. In *WABI '08: Proceedings of the 8th international workshop on Algorithms in Bioinformatics* (Berlin, Heidelberg, 2008), Springer-Verlag, pp. 1–13.

[19] Tannier, E., Zheng, C., and Sankoff, D. Multichromosomal median and halving problems under different genomic distances. *BMC Bioinformatics 10*, 1 (Apr 2009), 120.

[20] Yancopoulos, S., Attie, O., and Friedberg, R. Efficient sorting of genomic permutations by translocation, inversion and block interchange. *Bioinformatics 21*, 16 (Aug 2005), 3340–3346.