Algorithmic Approaches for Genome Rearrangement: A Review

Zimao Li, Lusheng Wang, *Member, IEEE*, and Kaizhong Zhang, *Member, IEEE*

***Abstract*—Genome rearrangement is a new and important research area that studies the gene orders and the evolution of gene families. With the development of fast sequencing techniques, large-scale DNA molecules are investigated with respect to the relative order of genes in them. Contrary to the traditional alignment approach, genome rearrangements are based on comparison of gene orders. Recently, it became a topic capturing wide attention. In this paper, we cover many kinds of rearrangement events such as *reversal*, *transposition*, *translocation*, *fussion*, *fission*, and so on. Different types of distances between genomes or chromosomes are discussed. A variety of mathematic models are included.**

***Index Terms*—Genome rearrangement, reversal, syntenic distance, translocation, transposition.**

I. Introduction

GENOME rearrangement was pioneered by Dobzhansky and Sturtevant [38], 60 years ago, who published a milestone paper with an evolutionary tree presenting a rearrangement scenario with 17 reversals for the species *Drosophila pseudoobscura* and *Miranda*. Many subsequent studies show that genome rearrangement is a common mode of molecular evolution in plants, mammals, viral, and bacteria [6], [55], [56], [58]–[60], [71], [96], [108], [110], [112].

In the late 1980s, Palmer *et al*. compared the mitochondrial genomes of *Brassica oleracea* (cabbage) and *Brassica campestris* (turnip) and found that they are very closely related (many genes are 99%–99.9% identical) [90]. Another example [88] shows that the only major difference between the two bacteria *Escherichia coli* and *Salmonella typhimurium* is the order of genes in their chromosomes. Rearrangement event *reversal* plays an important role in the diversity of plants and bacteria. In a study of *herps* viruses, researchers [54] faced the problem of analyzing an entire spectrum of genome rearrangements, in particular, *transpositions*. In 1984, when attempting to analyze genome rearrangements in mammalian genomes, Nadeau and Taylor [87] estimated that just 178 ± 39 rearrangement events happened since the separation of lineages leading to humans and mice 80 million years ago. This estimation was validated by Copeland *et al*. [32] based on a man–mouse genetic linkage map. The most common rearrangement events in mammalian evolution are *translocations* and reversals.

Both reversals and transpositions rearrange the order of genes in the same chromosome. A reversal reverses the order of a segment of genes in the chromosome, whereas a transposition removes a segment of genes from the chromosome and inserts it into the other place of the same chromosome. A translocation acts on two chromosomes. Suppose two chromosomes X and Y are cleavaged as the prefix–suffix form (X_1, X_2) and (Y_1, Y_2), where none of the four segments X_1, X_2, Y_1, and Y_2 are empty. A translocation swaps the prefix of one chromosome with the prefix or suffix of the other chromosome, resulting in two new chromosomes (X_1, Y_2) and (Y_1, X_2) or (Y_1, X_1^R) and (Y_2^R, X_2), where X^R denote the reverse of X.

Other rearrangement events such as *fusions* and *fissions* are common in mammalian evolution. For example, the only difference in the overall genome organization of humans and chimpanzees is the fusion of chimpanzee chromosomes 12 and 13 into human chromosome 2. A fusion concatenates two chromosomes into one, and a fission does the opposite work. Fusions and fissions are often considered together with some of the three basic operations, such as reversal, translocation, and transposition.

Classical alignment algorithms for sequence comparison can only handle local mutations instead of global rearrangements [68], [94], [124]. The first serious strike to the computation of rearrangement distance started in 1993 by Kececioglu and Sankoff [73]. Since then, more than 100 research papers on algorithmic results for genome rearrangement have been published. Recently, it has been developed into a wide research area with many mathematical models and unsolved computational problems. For details, see the new book [112] and a book chapter [108].

In this paper, we will review most of the algorithmic results for genome rearrangement. The rest of the paper is organized as follows. Section II defines the most common rearrangement operations. Section III covers results for reversals. Transpositions and translocations are discussed in Sections IV and V, respectively. Section VI is about duplication genes. Section VII presents results on syntenic distance. The median problem and phylogenetic tree reconstruction are reviewed in Section VIII. Conclusions can be found in Section IX.

II. Rearrangement Operations

A *gene* is represented by an *identity* that can be either a positive integer or a signed integer. Let $\mathcal{E}$ be a set of identities (genes). A *chromosome* is a permutation of identities over $\mathcal{E}$.

Manuscript received May 7, 2004; revised January 7, 2005. This work was supported in part by the Shandong Province Excellent Middle-aged and Young Scientists, Encouragement Fund (03BS004), in part by the Ministry of Education Study Abroad Returnees Research Start-up Fund, NSFC Project (60073042, 60273032), and in part by the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. CityU 1047/01E). This paper was recommended by Associate Editor M. Last.

Z. Li is with the College of Computer Science, South-Central University for Nationalities, 430074 Wuhan, China (e-mail: lizm@sdu.edu.cn).

L. Wang is with the Department of Computer Scicence, City University of Hong Kong, Kowloon, Hong Kong (e-mail: lwang@cs.cityu.edu.hk).

K. Zhang is with Department of Computer Science, University of Western Ontario, London, ON N6A 5B7, Canada (e-mail: kzhang@csd.uwo.ca).

Digital Object Identifier 10.1109/TSMCC.2005.855522

Unless clearly stated, each gene appears exactly once in a chromosome. If a gene can appear more than once in a chromosome, we use a *string* on $\mathcal{E}$ to represent the chromosome. A *genome* is a collection of chromosomes.

When we compare two chromosomes, we assume that one chromosome is the *identity permutation* $\iota = (1, 2, \ldots, n)$ and the other is a permutation $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$. If the orientation of each gene is provided, the chromosome is represented by a *signed* permutation, where each identity is a signed integer; i.e., a sign "+" or "−" is associated.

Now, we define some important operations related to genome rearrangement.

Definition 1: A *reversal* operation $\rho(i, j)$ of an interval $[i, j]$ on a permutation $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$ is the transformation

$$\rho\colon \pi_1, \ldots, \pi_{i-1}, \underline{\pi_i, \pi_{i+1}, \ldots, \pi_{j-1}, \pi_j}, \pi_{j+1}, \ldots, \pi_n)$$
$$\rightarrow (\pi_1, \ldots, \pi_{i-1}, \underline{\pi_j, \pi_{j-1}, \ldots, \pi_{i+1}, \pi_i}, \pi_{j+1}, \ldots, \pi_n).$$

If π is a signed permutation, in addition to reversing the order, a reversal $\pi \cdot \rho(i, j)$ also changes the signs of the genes $\pi_i, \pi_{i+1}, \ldots, \pi_j$.

Given permutations $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$ and $\sigma = (\sigma_1, \sigma_2, \ldots, \sigma_n)$, the *reversal distance* between π and σ is the minimum number of reversals required to transform π into σ.

Definition 2: A *transposition* $\rho(i, j, k)$ on $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$ is the transformation

$$\rho\colon (\pi_1, \ldots, \pi_{i-1}, \underbrace{\pi_i, \ldots, \pi_{j-1}}, \underbrace{\pi_j, \ldots, \pi_{k-1}}, \pi_k, \ldots, \pi_n)$$
$$\rightarrow (\pi_1, \ldots, \pi_{i-1}, \underbrace{\pi_j, \ldots, \pi_{k-1}}, \underbrace{\pi_i, \ldots, \pi_{j-1}}, \pi_k, \ldots, \pi_n).$$

Here, $1 \leq i < j \leq n+1, 1 \leq k \leq n+1$, and $k \notin [i, j]$.

It is easy to see that $\pi \cdot \rho(i, j, k)$ has the effect of moving the gene block $\pi_i, \pi_{i+1}, \ldots, \pi_{j-1}$ to a new location in π. Another view is that for $i < j < k$, $\rho(i, j, k)$ exchanges gene blocks $\pi_i, \pi_{i+1}, \ldots, \pi_{j-1}$ and $\pi_j, \pi_{j+1}, \ldots, \pi_{k-1}$.

Given permutations $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$ and $\sigma = (\sigma_1, \sigma_2, \ldots, \sigma_n)$, the *transposition distance* between π and σ is the minimum number of transpositions required to transform π into σ.

Definition 3: An *inverted transposition* $\rho(i, j, k)$ on $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$ can be viewed as a transposition operation plus a reversal operation having the following effect:

$$\rho\colon (\pi_1, \ldots, \pi_{i-1}, \underbrace{\pi_i, \ldots, \pi_{j-1}}, \underbrace{\pi_j, \ldots, \pi_{k-1}}, \pi_k, \ldots, \pi_n)$$
$$\rightarrow (\pi_1, \ldots, \pi_{i-1}, \underbrace{\pi_j, \ldots, \pi_{k-1}}, \underbrace{\pi_{j-1}, \ldots, \pi_i}, \pi_k, \ldots, \pi_n).$$

If π is a signed permutation, in addition to reversing the order, an inverted transposition $\pi \cdot \rho(i, j, k)$ also changes the signs of the genes $\pi_{j-1}, \pi_{j-2}, \ldots, \pi_i$.

Definition 4: A *block-interchange* $\rho(i, j, k, l)$ on $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$ is the transformation

$$\rho : (\pi_1, \ldots, \pi_{i-1}, \underbrace{\pi_i, \ldots, \pi_j}, \pi_{j+1} \ldots, \pi_{k-1},$$
$$\underbrace{\pi_k, \ldots, \pi_l}, \pi_{l+1}, \ldots, \pi_n)$$
$$\rightarrow (\pi_1, \ldots, \pi_{i-1}, \underbrace{\pi_k, \ldots, \pi_l}, \pi_{j+1} \ldots, \pi_{k-1},$$
$$\underbrace{\pi_i, \ldots, \pi_j}, \pi_{l+1}, \ldots, \pi_n).$$

Here, $1 \leq i \leq j < k \leq l \leq n$.

It is easy to see that a block-interchange swaps two nonintersecting gene blocks in a permutation, whereas a transpositoin swaps two adjacent gene blocks.

Given permutations $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$ and $\sigma = (\sigma_1, \sigma_2, \ldots, \sigma_n)$, we want to find a shortest series of block-interchanges $\rho_1, \rho_2, \ldots, \rho_t$ such that $\pi \cdot \rho_1 \cdot \rho_2 \cdots \rho_t = \sigma$. We call t the *block-interchange distance* between π and σ.

Note that reversals, transpositions, and block-interchanges act on one permutation. Another operation acts on two permutations.

Definition 5: Given two permutations $\pi = (\pi_1, \pi_2, \ldots, \pi_m)$ and $\sigma = (\sigma_1, \sigma_2, \ldots, \sigma_n)$, a *prefix-prefix translocation* $\rho_{\mathrm{pp}}(\pi, \sigma, i, j)$ produces two permutations $(\pi_1, \pi_2, \ldots \pi_{i-1}, \sigma_j, \sigma_{j+1}, \ldots, \sigma_n)$ and $(\sigma_1, \sigma_2, \ldots, \sigma_{j-1}, \pi_i, \pi_{i+1}, \ldots, \pi_m)$, and a *prefix-suffix translocation* $\rho_{\mathrm{ps}}(\pi, \sigma, i, j)$ produces two permutations $(\sigma_n, \sigma_{n-1}, \ldots \sigma_j, \pi_i, \ldots, \pi_m)$ and $(\sigma_1, \ldots, \sigma_{j-1}, \pi_{i-1}, \pi_{i-2}, \ldots, \pi_1)$. Here $1 < i \leq m$ and $1 < j \leq n$.

If π and σ are signed permutations, a prefix–suffix translocation $\rho_{\mathrm{pp}}(\pi, \sigma, i, j)$ results in two signed permutations $(-\sigma_n, -\sigma_{n-1}, \ldots - \sigma_j, \pi_i, \ldots, \pi_m)$ and $(\sigma_1, \ldots, \sigma_{j-1}, -\pi_{i-1}, -\pi_{i-2}, \ldots, -\pi_1)$.

The *translocation distance* between genomes G_1 and G_2 is the minimum number of translocations required to convert G_1 into G_2.

Definition 6: A *fusion* $\rho(\pi, \sigma)$ acting on two permutations $\pi = (\pi_1, \pi_2, \ldots, \pi_m)$ and $\sigma = (\sigma_1, \sigma_2, \ldots, \sigma_n)$ produces one chromosome $(\pi_1, \pi_2, \ldots, \pi_m, \sigma_1, \sigma_2, \ldots, \sigma_n)$.

Definition 7: A fission $\rho(\pi, i)$ acting on one permutation $\pi = (\pi_1, \pi_2, \ldots, \pi_m)$ produces two permutations $(\pi_1, \pi_2, \ldots, \pi_{i-1})$ and $(\pi_i, \pi_{i+1}, \ldots, \pi_m)$, where $1 < i \leq m$.

Most of the exact and approximate algorithms for genome rearrangements are based on the notion of *breakpoint*, which is defined as follows.

Definition 8: Given a permutation $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$ of $\iota = (1, 2, \ldots, n)$, we extend both permutations to $(0, \pi_1, \pi_2, \ldots, \pi_n, n+1)$ and $(0, 1, 2, \ldots, n, n+1)$. A *breakpoint* on π is a pair (π_i, π_{i+1}) with $|\pi_{i+1} - \pi_i| \neq 1$ for $0 \leq i \leq n$.

A *strip* of a permutation π is a segment $\pi_i \pi_{i+1} \ldots \pi_j$ where (π_{i-1}, π_i) and (π_j, π_{j+1}) are breakpoints and there is no breakpoint in this segment. The size of a strip is the number of genes in the strip.

Generally, given two permutations $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$ and $\sigma = (\sigma_1, \sigma_2, \ldots, \sigma_n)$, we say π_i and π_j are adjacent in π if $|i - j| = 1$. A pair (g, h) in π is a breakpoint if g and h are adjacent in π but not in σ. Define $\Phi(\pi, \sigma)$ to be the number of breakpoints in π. Obviously, $\Phi(\sigma, \pi) = \Phi(\pi, \sigma)$. The number of breakpoints between two genomes is the most general measure of genomic distance requiring no assumptions about the mechanisms of genome evolution and is very easy to calculate. Given

multiple genomes, Sankoff and Blanchette [15], [98] suggested the breakpoint distance as a measure for inferring phylogenies.

The notion of the breakpoint graph can be extended for signed permutations.

III. Sorting by Reversals

The reversal operations were first observed in the third chromosome of wild races of *Drosophila pseudoöbscura* more than 60 years ago [117]. In 1982, Watterson *et al.* gave the first definition of reversal distance problem for circular permutations, where the first gene is considered to be adjacent to the last gene [125]. The first serious strike to the computation of the reversal distance started about ten years ago by Kececioglu and Sankoff [72], [73]. Since then, computing the reversal distance has become a core in computational molecular biology.

Definition 9: Given two signed (unsigned) permutations, the *sorting signed (unsigned) permutation by reversals* problem is to compute a shortest series of reversals to transform one permutation into the other.

From the computational point of view, the only related work in history is a restricted version called *prefix reversals* or *pancake flipping problem* [27], [39], [46], [50], [64], [65], which are reversals of the form $[1, i]$.

A. Exact Algorithms

Kececioglu and Sankoff [72], [73] gave an exact algorithm for sorting unsigned permutation by reversals using a branch-and-bound approach. The algorithm finds an optimal solution in $O(mL(n, n))$ time and $O(n^2)$ space, where m is the size of the branch-and-bound search tree and $L(n, n)$ is the time to solve a linear programming of n variables and n constraints. They introduced the linear programming technique to obtain the lower bound. The upper bound can be found by considering the series of reversals removing the biggest number of breakpoints among all the ones of fixed length. Extending [72], [73], Kececioglu and Sankoff found tight lower and upper bounds for signed circular permutations and implemented an exact algorithm that works very fast for relatively long permutations [74]. They raised some conjectures in [72] and [73].

Conjecture 1: There exists an optimal series of reversals that does not cut strips other than at their first or last element.

Conjecture 2: There exists an optimal series of reversals that never increases the number of breakpoints.

Hannenhalli and Pevzner [59] proved that these two conjectures were correct. In [59], they proposed a polynomial-time algorithm for sorting unsigned permutation by reversals for a special case, where permutations do not have *singletons* (i.e., strips with size one). This demonstrates that singletons present a major obstacle on the way toward an efficient algorithm for sorting unsigned permutation by reversals. Based on the notion of *spin* of a permutation, they gave a new algorithm for sorting unsigned permutation by reversals and showed that the algorithm runs in polynomial-time for permutations with at most $O(\log n)$ singletons. Applications of this algorithm were provided for analyzing rearrangements in maize and green algae.

B. Approximate Algorithms

The pioneer work of Kececioglu and Sankoff [72], [73] proposed the first approximation algorithm for sorting unsigned permutation by reversals with performance ratio 2 that runs in $O(n^2)$ time and $O(n)$ space for n-element permutations. They used a greedy strategy.

Bafna and Pevzner [5] further studied the problem. For the signed case, they gave an approximation algorithm with ratio $(3/2)$ and running time $O(n^{(3/2)})$. For the unsigned case, they gave an approximation algorithm with ratio $(7/4)$ and running time $O(n^2)$. Computer software (*ReversalSort*) based on algorithms in [5] was implemented in [6]. Some experiments were done for plant organelles and mammalian X chromosomes. For the unsigned case, Christie [29] further improved the performance ratio to $(3/2)$ in 1998.

The latest approximation algorithm for the unsigned case was presented recently by Berman *et al.* [12]. They exploited the polynomial-time algorithm [57] for sorting signed permutation by reversals, and developed a new approximation algorithm for maximum cycle decomposition of the breakpoint graph.

Theorem 3: [12] Sorting unsigned permutation by reversals can be approximated within ratio 1.375.

C. Polynomial-Time Algorithms for Sorting Signed Permutations by Reversals

The first polynomial-time algorithm for sorting signed permutation by reversals was given by Hannenhalli and Pevzner [57] (extended version in [60]) that runs in $O(n^4)$ time for permutations of n genes. Previously, two parameters, the number of breakpoints $b(\pi)$ and the size of a maximum cycle decomposition $c(\pi)$ in the breakpoint graph, were shown to be closely related to the reversal distance [5], [72], [73]. Hannenhalli and Pevzner found the third hidden parameter $h(\pi)$, the number of *hurdles*. They showed that $b(\pi) - c(\pi) + h(\pi) \leq d(\pi) \leq b(\pi) - c(\pi) + h(\pi) + 1$, where $d(\pi)$ is the reversal distance. Based on the upper and lower bounds, they successfully solved the problem.

After Hannenhalli and Pevzner [57], many improved algorithms have been proposed. By exploiting a few combinatorial properties of the breakpoint graph of a permutation, Berman and Hannenhalli [11] proposed an $O(n^2\alpha(n))$ implementation of the algorithm in [57], where α is the inverse Ackerman function [118]. A faster and simpler $O(n^2)$ algorithm for sorting signed permutation by reversals was devised by Kaplan *et al.*.

Theorem 4 [69]: Sorting signed permutation by reversals can be computed in $O(n^2)$ time for permutations of size n.

All the above algorithms give the reversal distance as well as the series of reversals.

Bader *et al.* [4] recently presented a linear-time algorithm that only compute the reversal distance between two signed permutations. The algorithm cannot provide any series of reversals.

Theorem 5: Computing the reversal distance between two signed permutations can be done in $O(n)$ time for permutations of size n.

Based on the linear time algorithm, Bader *et al.* [4] also presented an $O(n^2)$ time algorithm to give an optimal series of reversals. The implementation of the $O(n^2)$ algorithm is faster than the implementations of the algorithms in [57] and [69] in practice. The program can be found in [3], [52], and [80].

Without using the breakpoint graph, Bergeron [9] gave a neat presentation of the Hannenhalli and Pevzner's theory [57] for computing the reversal distance between signed permutations. The presentation leads to a $O(n^2)$ time algorithm.

A bit-vector implementation of [9] was presented by Bergeron and Strabourg in [10].

Siepel [115] proposed an efficient algorithm for finding *all* possible shortest series of reversals transforming one permutation into the other.

D. Kaplan's Algorithm: An Example

In this section, we briefly illustrate the Kaplan's algorithm [69] for sorting signed permutation by reversals. We start by introducing some related terminologies.

Let $\pi = \pi_1\pi_2\cdots\pi_n$ be an unsigned permutation of the integers $1, 2, \ldots, n$. The *breakpoint graph* of π is an edge-colored graph $B(\pi)$ with $n+2$ vertices $\{\pi_0, \pi_1, \ldots, \pi_n, \pi_{n+1}\} = \{0, 1, \ldots, n+1\}$ where $\pi_0 = 0$ and $\pi_{n+1} = n+1$, a pair (π_i, π_j) is a *black* edge if (π_i, π_j) is a breakpoint in π (i.e., $|\pi_i - \pi_j| > 1$ and $|i-j| = 1$), and is a *gray* edge if (i, j) is a breakpoint in π^{-1} (i.e., $|\pi_i - \pi_j| = 1$ and $|i-j| > 1$). A cycle in $B(\pi)$ is *alternating* if the colors of every two consecutive edges in the cycle are distinct. In the following, a cycle always means an alternating cycle.

A signed permutation π is a permutation with each element in π associated with a sign "+" or "−." The *image* of a signed permutation π of order n is an unsigned permutation $u(\pi)$ of order $2n$ by replacing the positive element $+x$ in π by $2x-1$ and $2x$, and negative element $-x$ in π by $2x$ and $2x-1$. For any signed permutation π, let $B(\pi) = B(u(\pi))$. Note that in $B(\pi)$, every vertex is either isolated or incident to exactly one black edge and one gray edge. Therefore, there is a unique decomposition of $B(\pi)$ into alternating cycles. Let Let $b(\pi)$ and $c(\pi)$ be the number of breakpoints and the number of cycles in $B(\pi)$, respectively.

Call a reversal $\rho(i, j)$ an *even* reversal on $u(\pi)$ if i is odd and j is even. It is easy to see that an even reversal $\rho(2i+1, 2j)$ on $u(\pi)$ mimics the reversal $\rho(i, j)$ on π. Thus, sorting π by reversals is equivalent to sorting the unsigned permutation $u(\pi)$ by even reversals. In the following, by reversals, we mean even reversals. Say that a reversal is *acting on* a gray edge e if it is acting on the two black edges (corresponding to breakpoints) that are incident to e.

For an arbitrary reversal ρ on a permutation π, define $\Delta b(\pi, \rho) \equiv b(\pi\rho) - b(\pi)$ and $\Delta c(\pi, \rho) \equiv c(\pi\rho) - c(\pi)$. Call a reversal ρ *proper* if $\Delta b(\pi, \rho) - \Delta c(\pi, \rho) = -1$. Call a gray edge e *oriented* if a reversal acting on e is proper and *unoriented* otherwise. A cycle in $B(\pi)$ is called *oriented* if it contains an oriented gray edge, and it is *unoriented* otherwise.

For a permutation π, associate with a gray edge (π_i, π_j) in the interval $[i, j]$. The *overlap graph* $OV(\pi)$ of a permutation π is defined as follows: the vertex set of $OV(\pi)$ is the set of gray edges in $B(\pi)$, and two vertices are connected if the intervals associated with their gray edges overlap. A connected component of $OV(\pi)$ that contains an oriented edge is called an *oriented* component, otherwise it is called an *unoriented* component.

Let $\pi_{i_1}, \pi_{i_2}, \ldots, \pi_{i_k}$ be the subsequence of $0, \pi_1, \ldots, \pi_n, n+1$ consisting of those elements incident to the gray edges in $B(\pi)$ that occur in unoriented components of $OV(\pi)$ (Note that each gray edge of $B(\pi)$ corresponds to a vertex in $OV(\pi)$). Order $\pi_{i_1}, \pi_{i_2}, \ldots, \pi_{i_k}$ on a circle CR such that π_{i_j} follows $\pi_{i_{j-1}}$ for $2 \le j \le k$ and π_{i_1} follows π_{i_k}.

Let M be an unoriented component in $OV(\pi)$ and $E(M) \subset \{\pi_{i_1}, \pi_{i_2}, \ldots, \pi_{i_k}\}$ be the set of endpoints of the gray edges in $B(\pi)$, where the gray edges are vertices in M. An unoriented component M is a *hurdle* if the elements of $E(M)$ occur consecutively on CR. Two hurdles H_1 and H_2 are *consecutive* if $E(M_1)$ and $E(M_2)$ occur consecutively in CR. Call a reversal *merging* two hurdles H_1 and H_2 if it acts on two breakpoints, one incident with a gray edge in H_1 and the other incident with a gray edge in H_2.

A hurdle is *simple* if when one deletes it from $OV(\pi)$, no other unoriented component becomes a hurdle; otherwise, it is *super hurdle*. A *fortress* is a permutation with an odd number of hurdles, all of which are superhurdles. Denote by $h(\pi)$ the number of hurdles in a permutation π.

Call a reversal *safe* if it acts on an oriented gray edge e and does not create new hurdles.

Hannenhalli and Pevzner proved the following duality theorem.

Theorem 6 [57]: Let $d(\pi)$ be the minimum number of reversals required to sort a permutation π. Then

$$d(\pi) = \begin{cases} b(\pi) - c(\pi) + h(\pi) + 1, & \text{if } \pi \text{ is a fortress} \\ b(\pi) - c(\pi) + h(\pi), & \text{otherwise.} \end{cases}$$

Following the theory developed by Hannenhalli and Pevzner [57], for a given permutation π with $h(\pi) > 0$, one can perform $t = \lceil (h(\pi)/2) \rceil$ reversals and transform π into a permutation π' such that $h(\pi') = 0$ and $d(\pi') = d(\pi) - t$. Kaplan's algorithm clears the hurdles first based on the following two lemmas.

Lemma 7 [69]: Let π be a permutation with an even number, say $2k$, of hurdles. Any sequence of $k-1$ reversals each of which merges two nonconsecutive hurdles followed by a reversal merging the remaining two hurdles will transform π into π' such that $d(\pi') = d(\pi) - k$ and π' has only oriented components.

Lemma 8 [69]: Let π be a permutation with an odd number, say $2k+1$, of hurdles. If at least one hurdle H is simple then a reversal acting on two breakpoints incident with edges in H transforms π into π' with $2k$ hurdles such that $d(\pi') = d(\pi) - 1$. If π is fortress then a sequence of $k-1$ reversals merging pairs of nonconsecutive hurdles followed

Kaplan's Algorithm

Input A signed permutation π.
Output A sequence of minimum number of reversals transforming π into the identity.
1. Construct the overlap graph $OV(\pi)$.
2. Clear the hurdles.
3. **while** π is not the identity **do**
   (a) Find a safe reversal ρ.
   (b) Update π and $OV(\pi)$ according to ρ.

Fig. 1. Kaplan's algorithm for sorting signed permutations.

by two additional merges of pairs of consecutive hurdles (one merges two original hurdles and the next merges a hurdle created by the first and the last original hurdle) will transform π into π' such that $d(\pi') = d(\pi) - (k+1)$ and π' has only oriented components.

The remaining work is to repeatedly find a safe reversal transforming the hurdle-free permutation π' into the identity permutation. Remember, π' is obtained from π by clearing the hurdles. Kaplan's algorithm is given in Fig. 1.

E. Hardness for Sorting Unsigned Permutation by Reversals

Sorting an unsigned permutation by reversals was proved to be NP-hard by Caprara in 1997.

Theorem 9 [21], [23]: Sorting an unsigned permutation by reversals is NP-hard.

In [21], Caprara first showed the NP-hardness of the *maximum alternating cycle decomposition problem* with a reduction from an NP-hard problem called *maximum Eulerian cycle decomposition* [66]. Then, they gave a reduction from the maximum alternating cycle decomposition problem to an sorting an unsigned permutation by reversals.

The problem was shown to be MAX SNP-hard by Berman and Karpinski [13].

Theorem 10 [13]: Sorting unsigned permutation by reversal is MAX SNP-hard.

Theorem 10 implies that the problem does not admit a polynomial-time approximation scheme (PTAS) unless $\mathrm{P} = \mathrm{NP}$. In fact, they showed a stronger result:

Theorem 11 [13]: Sorting unsigned permutation by reversals cannot be approximated within 1.0008 unless $\mathrm{P} = \mathrm{NP}$.

Another interesting problem is called *sorting words by reversals*.

Sorting words by Reversals:

Input: A string $w = w_1 w_2 \dots w_k, w_i \in \{1, 2, \dots, m\}$ for $i = 1, 2, \dots, k$ and $k > m$.

Output: A shortest sequence of reversals transforming w into a sorted string $y_1 y_2 \dots y_n$ such that for $i = 1, 2, \dots, n-1$, $y_i \leq y_{i+1}$.

The main difference between sorting unsigned permutations by reversals and sorting words by reversals is that a word is allowed to appear more than once in the string.

To illustrates the flavor of the NP-hardness proofs, we will given a reduction from the problem of sorting unsigned permutations by reversals to the problem of sorting words by reversals [93].

The reduction: Given an arbitrary permutation π of the identity $\iota = (1, 2, \dots, n)$, we can construct a string $w = 1\pi n$ in polynomial-time such that $w_1 = 1, w_{n+2} = n$, and $w_{i+1} = \pi_i$ for each $1 \leq i \leq n$. w is an instance of the problem of sorting words by reversals. It is easy to see that a shortest sequence of reversals for π indicates a shortest sequence of reversals for w, and vice versa. Since sorting unsigned permutations by reversals is NP-hard, the NP-hardness of sorting words by reversals is obtained immediately.

F. Sorting by Reversals With Insertions and Deletions

When different genes are allowed in the two given chromosomes, insertions and deletions of genes (blocks of permutations) are necessary in the procedure of transforming one chromosome into the other. El-Mabrouk [41] studied the following problem.

Definition 10: Let G and H be two chromosomes. Assume that there are some genes that appear in one chromosome and do not appear in the other. Let $\mathcal{A}$ be the set of genes in both G and H. The *sorting signed permutation by reversals with insertions and deletions* problem is to find the minimum number of reversals, insertions, and deletions of gene blocks to transform G into H. Here, deletions and insertions are not applied on $\mathcal{A}$.

Recently, El-Mabrouk [41] extended Hannenhalli and Pevzner's polynomial-time approach in [57] and presented an exact algorithm with time complexity $O(n^2)$ for the sorting signed permutation by reversals with insertions and deletions problem.

G. Other Results

For sorting unsigned permutation by reversals, a special case, wherein the reversal distance of the given permutations is twice the number of breakpoints has been considered. Tran [120] showed that the restricted version can be solved in polynomial-time. This disproves the conjecture in [73] and [93] that the restricted version is NP-hard.

Chen and Skiena [26] considered the problem of sorting by fixed-length reversals. They gave a complete solution of the number of equivalence classes of n permutations under k-reversal for all n and k, and proved the upper and lower bounds on the diameter of the circular permutation group induced by k-reversals.

Caprara *et al.* [25] devised a fast practical algorithm for sorting unsigned permutation by reversals based on linear programming relaxation. Their algorithm is efficient to the real-world instances with proven optimality and to random instances with $n \leq 100$. An $O(n^3 n!)$ exact algorithm determining the reversal distances for all permutations of size n was presented by Heath and Vergara [61]. The algorithm was used for testing several conjectures.

IV. Sorting by Transpositions

Transposition was first studied by Bafna and Pevzner [7], [8]. It should be clearly pointed out that a transposition operation never changes the sign of a gene, and it just moves one segment

of genes to another place in the chromosome. Thus, we can always assume that the sign of each gene in both chromosomes is the same. Thus, we only have the *signed* version here.

Definition 11: Given two permutations, the sorting by transpositions problem is to find a shortest series of transpositions to transform one permutation into the other.

A. General Case

Bafna and Pevzner [7], [8] studied the problem of sorting permutations by transpositions, derived lower bounds on transposition distance between chromosomes, and proposed the first approximation algorithm with performance ratio 1.5 and time complexity $O(n^2)$. Christie [30] devised an alternative 1.5-approximation algorithm running in time $O(n^4)$. Christie's algorithm is easier to understand.

Theorem 12 [7], [8], [30]: There exist approximation algorithms with ratio 1.5 for sorting by transpositions problem.

Guyer *et al.* [49] proposed some heuristics based on the analysis of subsequences. Experimental results show that their algorithm often produces near-optimal solutions. The following is one of the major open problems in this area.

Open Problem 1: Does there exist a polynomial-time algorithm to solve the sorting by transpositions problem?

B. Special Cases

When restrictions are imposed on this problem, efficient exact algorithms or approximation algorithms with improved performance ratio can be obtained. Jerrum [67] presented a polynomial-time algorithm for sorting by transpositions when only pairwise adjacent elements were permitted to exchange. Aigner and West [1] showed that the problem is solvable in polynomial-time when only the first element of a permutation can be swapped with its adjacent blocks. Heath and Vergara [62] considered the restricted version by bounding the length of the two adjacent blocks being exchanged. They showed that the problem is solvable in $O(n^2)$ time when one of the blocks is a single element and the other block is unrestricted. For the case that the total length of the two adjacent blocks being exchanged is bounded by some function proportional to n, they reduced the general problem of sorting by transpositions to the bounded one, indicating that it is at least as difficult as the general case. Detailed investigations were performed for the restricted version when the total length of the two adjacent blocks being swapped is 3 (denote as sorting by short block-moves). An approximation algorithm with performance ratio $(4/3)$ for the sorting by short block-moves problem was devised by Heath and Vergara [63]. Polynomial-time algorithms were also provided for sorting by short block-moves problems when further restrictions are introduced [62], [63].

C. Sorting by Reversals and Transpositions

Bafna and Pevzner [7], [8] suggested the sorting problem that considers reversals and transpositions simultaneously as an approach for understanding the genomes rearrangements related to mammalian genome evolution, viral evolution, and so on.

Definition 12: Given two signed (unsigned) permutations, the sorting signed (unsigned) permutation by reversals and transpositions problem is to find a shortest series of reversals and transpositions to transform one permutation into the other.

The *signed (unsigned) reversal and transposition distance* between two signed (unsigned) permutations is the minimum number of reversals and transpositions transforming one permutation into the other.

Definition 13: Given two signed (unsigned) permutations, the sorting signed (unsigned) permutation by reversals, transpositions and inverted transpositions problem is to find a shortest series of reversals, transpositions and inverted transpositions to transform one permutation into the other.

Walter *et al.* [121] presented a ratio-3 approximation algorithm for computing the unsigned reversal and transposition distance and a ratio-2 approximation algorithm for computing the signed reversal and transposition distance, both running in time $O(n^2)$.

In [47], Gu *et al.* proposed a greedy heuristic for the sorting signed permutations by reversals, transpositions, and inverted transpositions problem. An $O(n^2)$ approximation algorithm with performance ratio 2 was proposed by Gu *et al.* [48], for the sorting signed permutation by reversals, transpositions, and inverted transpositions problem.

Lin and Xue [79] studied the problem of sorting signed permutations by combined operations. They gave unified $O(n^2)$ approximation algorithms with performance ratio 2 for the sorting signed permutations by reversals and transpositions problem, and the sorting signed permutations by reversals, transpositions, and inverted transpositions.

Wang and Warnow [123] developed a technique called the inverse of the expected number of breakpoints (*IEBP*) to estimate the *true evolutionary distance* between two genomes (signed or unsigned, circular or linear), and later the technique was refined by Wang [122] with a more accurate method, namely *Exact-IEBP*, for signed genomes. The true evolutionary distance between two genomes that Wang and Warnow considered is the minimum number of reversals, transpositions and inverted transpositions required to transform one genome into the other.

The computational complexity for all the problems discussed in this subsection remains open.

D. Weighted Sorting by Reversals and Transpositions

Scientists have observed that in practice, transpositions occur with about half the frequency of reversals [16]. This inspired research to consider the weighted problem of genome rearrangements. Recently, a weighted problem of sorting by reversals and transpositions; i.e., a transposition is weighted twice that of a reversal, was studied. Using Hannenhalli and Pevzner's exact algorithm [57] for signed permutations as a basis, Eriksen [43] presented a PTAS for this problem under the restriction that the given permutations are signed and circular. Based on classical results of permutation group theory, Dias and Meidanis [37] studied another weighted problem of sorting by fusion, fission, and transposition simultaneously, where transpositions have twice the weighting of

fusions and fissions, and devised a polynomial-time algorithm with time complexity $O(n^2)$ for finding the minimum weight series of fusions, fissions, and transpositions when the given genomes are represented as circularly ordered sequences of genes. This is the first polynomial-time algorithm involving transpositions.

E. Sorting by Block-Interchanges

In [28], Christie introduced the following problem.

Definition 14: Given two permutations, the sorting by block-interchanges problem is to find a shortest series of block-interchanges transforming one permutation into another.

A block-interchange can be viewed as a generalization of transposition. In a block-interchange, two nonintersecting blocks or substrings of any length are swapped in the permutation, whereas only adjacent substrings are allowed to be swaped in a transposition.

Christie showed that this problem can be solved in polynomial-time.

Theorem 13 [28]: There is an $O(n^2)$ time algorithm for the sorting by block-interchanges problem.

V. Sorting by Translocations

Kececioglu and Ravi [71] were the first to study the *translocation distance* from computation point of view. Contrary to reversals and transpositions, a translocation exchanges material of two chromosomes in a genome. When the maps identifying the location of genes and other markers of interest along the chromosomes are available, e.g., man and mouse, the study of sorting by translocations becomes necessary and practical [89]. There are two versions: the signed case and unsigned case.

Definition 15: Given two signed (unsigned) genomes, both involving the same number of chromosomes and genes, the signed (unsigned) sorting by translocations problem is to find a shortest series of translocations transforming one genome into the other.

Kececioglu and Ravi [71] started the algorithmic study of the unsigned sorting by translocations problem and gave an approximation algorithm with performance ratio 2 and time complexity $O(n^2)$, where n is the total number of distinct genes in a genome.

Theorem 14 [71]: There is a polynomial-time approximation algorithm with performance ratio 2 for the unsigned sorting by translocations problem.

If the segments being swapped by a translocation are of equal length, the problem can be solved in $O(n)$ time [71].

Hannenhalli [53] solved the signed sorting by translocations problem.

Theorem 15 [53]: The signed sorting by translocations can be solved in polynomial-time.

Hannenhalli's algorithm runs in $O(n^3)$ time. Zhu [126] recently gave a faster algorithm that runs in $O(n^2 \log n)$ time.

The complexity of the unsigned sorting by translocations problem is still left open.

Open Problem 2: Is unsigned sorting by translocations problem NP-hard?

A. Sorting by Reversals and Translocations

In [71], Kececioglu and Ravi considered the following problems:

Definition 16: Given two signed (unsigned) genomes, both involving the same number of chromosomes and genes, the signed (unsigned) sorting by reversals and translocations problem is to find a shortest series of reversals and translocations transforming one genome into the other.

In [71], Kececioglu and Ravi showed that there is a 2-approximation algorithm for the unsigned sorting by reversals and translocations. They also gave an approximation algorithm with performance ratio $(3/2)$ for signed sorting by reversals and translocations. Both approximation algorithms have time complexity $O(n^2)$, where n is the total number of distinct genes in a genome.

B. Sorting by Reversals, Translocations, Fusions, and Fissions

When the number of chromosomes of the two given genomes is different, fusions, and fissions are necessary for transforming one into the other.

Definition 17: Given two signed (unsigned) genomes, both involving the same number of genes, the signed (unsigned) sorting by reversals, translocations, fusions, and fissions problem is to find a shortest series of reversals, translocations, fusions, and fissions transforming one genome into the other.

Hannenhalli and Pevzner [58] considered the signed case. They gave a polynomial-time algorithm for the problem of signed sorting by reversals, translocations, fusions and fissions.

Theorem 16 [58]: The signed sorting by reversals, translocations, fusions, and fissions problem can be solved in $O(n^4)$ time, where n is the total number of genes in a genome.

Tesler [119] implemented a program based on [51], [57], and [58] for the signed by reversals, translocations, fusions, and fissions problem.

VI. Duplicate Genes

When duplicate genes are permitted in chromosomes (originate from genome duplication; see [84] and the references therein), a gene may have a number of copies in each chromosome. In this case, we use a *string* to represent the chromosome and *characters* to represent the genes. Christie and Irving [31] considered the problems of *sorting strings by reversals* and *sorting strings by transpositions*.

Definition 18: Let S and T be two strings on an alphabet Σ. For each character $v \in \Sigma$, the number of occurrences of v in S and T is the same. The sorting string by reversals problem is to compute a shortest series of reversals transforming S into T.
Here genes are represented by characters without any sign.

Definition 19: Let S and T be two strings on an alphabet Σ. For each character $v \in \Sigma$, the number of occurrences of v in S and T is the same. The sorting string by transpositions problem is to compute a shortest series of transpositions transforming S into T.
Again, each gene is represented by a character without any sign.

Note that the definitions of reversal and transposition on strings are similar to those on permutations.

The NP-hardness of unsigned sorting permutation by reversals immediately implies that sorting string by reversals is NP-hard for unbounded alphabet. Christie and Irving [31] proved the following.

Theorem 17: The sorting string by reversals problem is NP-hard even when the alphabet size is 2.

The complexity of sorting string by transpositions is still left open. They also provided lower bound and upper bound for both sorting string by reversals and sorting string by transpositions.

Another related version is *sorting string by block-interchanges* which is first studied in [30].

Definition 20: Let S, and T be two strings on an fixed-size alphabet Σ. For each character $v \in \Sigma$, the number of occurrences of v in S and T is the same. The sorting string by block-interchanges problem is to compute a shortest series of block-interchanges transforming S into T.

In [30], Christie showed the following theorem.

Theorem 18: The sorting strings by block-interchanges problem is NP-hard, even when the alphabet size is 2.

Previously, we assume that the numbers of occurrences of each gene in S and T are the same. Sankoff [97] proposed a model that deals with the case where a gene could have different numbers of occurrences in S and T. Given an alphabet Σ, let G and H be two strings (chromosomes) of signed ("+" or "−") characters (genes) from Σ. Each character (whether associated with "+" or "−") in Σ occurs at least once in both G and H. For each chromosome, an *exemplar* string is constructed by deleting all but one occurrence of each duplicated genes. For two exemplar strings, each gene in Σ appears exactly once.

Definition 21: Given an alphabet Σ and two strings G and H of signed characters from Σ, the *exemplar breakpoint distance* between G and H is the minimum number of breakpoints over all choices of exemplar strings g and h.

Definition 22: Given an alphabet Σ and two strings G and H of signed characters from Σ, the *exemplar signed reversal distance* between G and H is the minimum reversal distance between two exemplar strings. Here we have to consider all choices of exemplar strings.

Sankoff [97] gave efficient branch and bound algorithms for both the exemplar breakpoint distance and the exemplar signed reversal distance. Bryant [19] discussed the complexity of the exemplar distance and showed that the calculation of the exemplar signed reversal distance and exemplar breakpoint distance are both NP-hard.

VII. Syntenic Distance

The *syntenic distance* is used when ignoring the order of genes on a chromosome, or the order is presumed to be unknown. In this case, a chromosome can be represented by a gene set and a genome is thus a collection of sets. The *syntenic distance* was first introduced by Ferretti *et al.* [45] and was defined to be the minimum number of fusions, fissions, and translocations required to transform one genome into the other.

Definition 23: An unordered chromosome is a set of genes. An unordered genome is a collection of unordered chromosomes.

Definition 24: Given two unordered chromosomes $S = S_1 \cup S_2$ and $T = T_1 \cup T_2$, where at most one of S_1, S_2, T_1, and T_2 is empty and S and T are disjoint.

A translocation $\rho(S, T)$ acting on S and T produces two chromosomes $S_1 \cup T_2$ and $T_1 \cup S_2$.

A fusion $\rho(S, T)$ acting on S and T produces one chromosome $S \cup T$.

A fission $\rho(S)$ acting on S produces two chromosomes S_1 and S_2.

Definition 25: Given two unordered genomes G_s and G_t containing the same set of genes, the syntenic distance of G_s and G_t is the minimum number of fusions, fissions, and translocations to transform G_s into G_t.

Ferretti *et al.* [45] were the first to consider the *syntenic distance* between two genomes. They provided a heuristic attempting to compute/approximate the syntenic distance and provided empirical evidence for the syntenic distance measure.

The problem of computing the syntenic distance between two genomes was proved to be NP-hard by Das Gupta *et al.* [35], [36].

Theorem 19 [35], [36]: It is NP-hard to compute the syntenic distance between two unordered genomes.

In [35] and [36], the authors also gave a simple $O(nk)$ approximation algorithm with performance ratio 2, where n and k are the number of genes and number of chromosomes, respectively.

Linben-Nowell [76] proved that the unanalyzed heuristic given by Ferretti *et al.* [45] is never worse than the approximation algorithm proposed by Das Gupta *et al.*, indicating that the heuristic is in fact an approximation algorithm with performance ratio 2. A number of properties of combinatorial structures concerning syntenic distance model can be found in [76], [78], and [95].

Based on a tight connection between syntenic distance and the *incomplete gossip problem* (a novel generalization of the classical gossip problem), Liben-Nowell [77] gave an exact $O(2^{O(n \log n)})$ algorithm computing the syntenic distance between two genomes that contain at most n chromosomes. If the syntenic distance is bounded, their algorithm requires $O(2^{O(d \log d)})$ time, improving upon the previous best known $O(2^{O(d^2)})$ exact algorithm in [36].

Kleinberg and Liben-Nowell [75] showed that the maximum syntenic distance is $2n - 2$ between any pair of genomes with n genes.

El-Mabrouk and Sankoff [42] considered to infer the posthybridization rearrangement in a hybrid genome when the gene orders on its genomes and some knowledge of the two parent genomes are given. Hybridization through interspecific fertility is also discussed in [42].

VIII. Median Problem and Phylogenetic Tree Reconstruction

In the previous sections, we considered the genome rearrangements between two genomes. When studying more than two genomes, the key problem arising is to reconstruct a most

parsimonious phylogenetic tree, where each leaf is associated with a given genome.

A. Median Problem

When the topology of the phylogenetic tree is restricted to a star, the problem is called the *median problem*.

Definition 26: Given a set of chromosomes (genomes) $S = (G_1, G_2, \ldots, G_m)$ and a genome rearrangement distance metric $d(,)$ defined on pairs of chromosomes (genomes), the median problem is to find a chromosome (genome) G such that $\sum_{i=1}^{m} d(G_i, G)$ is minimized.

If the number of given chromosomes (genomes) in the median problem is k, we denote the problem as the k-*median* problem.

The genome rearrangement distance metric $d(,)$ can be the reversal distance, transposition distance, breakpoint distance, or syntenic distance. The chromosomes/genomes can be signed or unsigned.

1) Median Problem for Reversal Distance and Transposition Distance: In [114], for the three-median problem, Sankoff *et al*. described some approximation algorithms with constant performance ratios for (signed and unsigned) reversal distance and for the unsigned transposition distance. They also proposed a local optimal heuristic for the three-median problem for (signed and unsigned) reversal distance and unsigned transposition distance.

Hannenhalli *et al*. described gave a bounded exhaustive search approach for signed reversal distance and signed reversal plus transposition distance for three-median problem. It works well on some chromosomes with short length (for example, a chromosome contains less than seven genes) [55].

For signed reversal distance, Caprara [22] proved the following.

Theorem 20 [22]: The three-median problem for singed reversal distance is MAX SNP-hard.

Caprara [22] also provided a $(2 - (2/m))$-approximation algorithm for the median problem running in time $O(n^2m^2)$, where m and n are the number of given chromosomes and the number of genes in a given chromosome, respectively.

Effective heuristics for the k-median problem of signed reversal distance was proposed by Caprara [24]. For signed reversal distance, Siepel and Moret [116] derived a branch-and-bound algorithm finding an optimal median chromosome in reasonable time for 3 signed chromosomes of medium size.

2) Median Problem for Breakpoint Distance: Sankoff *et al*. In [15], [98] demonstrated that the reversal distance and similar distance metrics have certain weaknesses making them inappropriate for studying complex phylogeny inference trees. They suggested the feasible *breakpoint distance* metric. Cosner *et al*. [33], [34] confirmed the usefulness of breakpoint analysis for phylogeny inference. Evidence for the feasibility of breakpoint distance can also be found in animal mitochondrial phylogeny [17].

Sankoff and Blanchette [98] developed several efficient heuristics for the three-median problem on signed and unsigned breakpoint distance. In [15], [98], and [100], Sankoff and Blanchette demonstrated that the median problem for breakpoint distance can be solved exactly for moderate size (signed or unsigned) chromosomes.

Pe'er and Shamir [91] and Bryant [18] independently settled the complexity of the three-median problem for the breakpoint distance.

Theorem 21 [91], [18]: The three-median problem for signed and unsigned breakpoint distance is NP-hard.

For signed breakpoint distance, Pe'er and Shamir [92] designed a polynomial-time approximation algorithm with performance ratio $(7/6)$ for the three-median problem, and a $(11/8)$-approximation algorithm for four-median problem.

Theorem 22 [92]: For signed breakpoint distance, there are polynomial-time approximation algorithms with performance ratios $(7/6)$ and $(11/8)$ for three-median problem and four-median problem, respectively.

3) Median Problem for Syntenic Distance: When the order of chromosomes is unknown, DasGupta *et al*. [36] studied the three-median problem for syntenic distance. They proved the following.

Theorem 23 [36]: The three-median problem for syntenic distance is NP-hard.

B. Phylogenetic Tree Reconstruction

The median problem is a special case of the *phylogenetic tree problem*, which is mathematically defined as follows:

Definition 27: Given a fixed phylogeny (tree) T with m terminals (leaves), and a set of m chromosomes (genomes), one for each terminal, the phylogenetic tree problem asks to find a set of chromosomes (genomes), one for each internal node, such that the total weight $w(T)$ of the tree is minimized. Here $w(T) = \sum_{(x,y)\in T} d(x, y)$, (x, y) is an edge in T and $d(,)$ is a genome rearrangement distance (e.g., reversal distance, translocation distance, transposition distance, and syntenic distance).

Sankoff *et al*. [114] proposed a heuristic for signed and unsigned reversal distances and transposition distance. They combine the local optimization approach in [114] and the *iterative method* in [107].

In [99], Sankoff and Blanchette proposed a heuristic for breakpoint distance that works well for moderate size signed circular chromosomes. Extensions of [15], [98], and [99] are given in [100]. Other heuristics can be found in [15], [34], and [83].

C. Phylogenetic Tree Reconstruction for Genomes With Unequal Gene Contents

When the genomes contain different sets of genes, both the median problem and the phylogenetic tree problem become more difficult. Sankoff *et al*. considered the problems in [105] and [106].

Definition 28: Given a chromosome (genome) A and a set Σ of genes, $A|_\Sigma$ is the induced chromosome (genome) from A with all the remaining genes in Σ.

For a chromosome (genome) A, $\mathcal{G}(A)$ denotes the set of genes in A.

Definition 29: Given two chromosomes (genomes) A and B containing some different genes, the breakpoint distance

$d(A,B)$ between A and B is the number of breakpoints of $A|_{\mathcal{G}(B)}$ and $B|_{\mathcal{G}(A)}$, the normalized breakpoint distance $d_n(A,B)$ between A and B is defined as $d_n(A,B) = (1/|\mathcal{G}(A) \cap \mathcal{G}(B)|)d(A,B)$, where the subscript n represents "normalized."

Definition 30: Given a set $\mathcal{A}$ of m chromosomes $A_1, A_2, \ldots, A_m$ with unequal gene contents, the breakpoint median problem with unequal gene contents for $A_1, A_2, \ldots, A_m$ is to find a chromosome X such that $\Psi(X,\mathcal{A}) = \sum_{i=1}^{m} d_n(X, A_i)$ is minimized, where $\mathcal{G}(X) = \mathcal{G}(A_1) \cup \mathcal{G}(A_2) \cup \cdots \cup \mathcal{G}(A_m)$.

Definition 31: Given a fixed phylogeny T, and a set $\mathcal{A}$ of m chromosomes (genomes) $A_1, A_2, \ldots, A_m$, one for each leaf, the breakpoint phylogenetic tree problem is to find chromosomes (genomes), one for each internal node, such that the total weight $w(T) = \sum_{(x,y)\in T} d_n(x,y)$ of the tree is minimized. (If x is a chromosome (genome) associated with an internal node, and x_1 and x_2 are the two chromosomes (genomes) associated with its two children, we have $\mathcal{G}(x) = \mathcal{G}(x_1) \cup \mathcal{G}(x_2)$.)

The breakpoint median problem (with unequal gene contents) and the breakpoint phylogenetic tree problem (with unequal gene contents) are obviously NP-hard, since they are NP-hard even when each chromosome (genome) contains the same set of genes [18], [91].

Sankoff *et al.* [105], [106] proposed an efficient heuristic for the breakpoint median problem for chromosomes (genomes) with unequal gene contents. By incorporating the heuristic for the median problem, Sankoff *et al.* obtained an efficient heuristic for breakpoint phylogenetic tree problem for chromosomes (genomes) with unequal gene contents. Their algorithms were applied successfully to the study of early eukaryote evolution. Bryant [20] derived lower bounds for the two problems. Experiments showed that the lower bounds were close to the upper bounds established by those heuristics in [106].

A survey mainly on multiple genome rearrangement phylogeny inference is presented by Blanchette [14]. For the notion and applications of *phylogenetic invariant* for genome rearrangement, corresponding papers may be found in [101]–[104]. IEBP-related methods can be found in [82], [122], and [123].

IX. Conclusion

In this paper, we have introduced the basic concepts for genome rearrangement and have reviewed most of the algorithmic results for problems concerning the measures of genomic distances and phylogenetic tree reconstruction. In practice, many of the algorithms have been applied to real biological data and have given good insights about the evolution of the species considered. In theory, most of the results for genome rearrangement problems are based on breakpoint graphs, indicating that it is a powerful tool for such problems.

The time complexity of the basic operation—transposition—is still open. Inferring phylogenetic trees based on genome rearrangement is a relatively new and difficult area. Many problems there remain open. With the various sequencing projects in progress, the whole genome of various organisms will be completely sequenced, and new and interesting problems will be proposed.

Acknowledgment

The authors would like to thank the referees for their helpful suggestions.

References

[1] M. Aigner and D. B. West, "Sorting by insertion of leading element," *J. Combinatorial Theor. A*, vol. 45, pp. 306–309, 1987.

[2] L. A. Andersson *et al.*, "Comparative genome organization of vertebrates," *Mammalian Genome*, vol. 7, pp. 717–734, 1996.

[3] D. A. Bader, B. M. E. Moret, and M. Yan, *GRAPPA: Genome Rearrangements Analysis Under Parsimony and Other Phylogenetic Algorithms*. [Online]. Available: http://www.cs.unm.edu/~moret/GRAPPA/

[4] ——, "A linear-time algorithm for computing inversion distance between signed permutations with an experimental study," in *Proc. 7th Int. Workshop Algorithms and Data Structures*, Aug. 2001, pp. 365–376.

[5] V. Bafna and P. Pevzner, "Genome rearrangements and sorting by reversals," in *Proc. 34th Annu. Symp. Foundations of Computer Science*, Nov. 1993, pp. 148–157.

[6] ——, "Sorting by reversals: Genome rearrangements in plant organelles and evolutionary history of x chromosome," *Mol. Biol. Evol.*, vol. 12, pp. 239–246, 1995.

[7] ——, "Sorting permutations by transpositions," in *Proc. 6th Annu. ACM-SIAM Symp. Discrete Algorithms*, Jan. 1995, pp. 614–623.

[8] ——, "Sorting by transpositions," *SIAM J. Discr. Math.*, vol. 11, no. 2, pp. 272–289, 1998.

[9] A. Bergeron, "A very elementary presentation of the Hannenhalli-Pevzner theory," in *Proc. 12th Annu. Symp. Combinatorial Pattern Matching*, Jul. 2001, pp. 106–117.

[10] A. Bergeron and F. Strasbourg, "Experiments in computing sequences of reversals," in *Proc. 1st Workshop Algorithms in Bioinformatics*, Aalborg, Denmark, Aug. 2001, pp. 164–174.

[11] P. Berman and S. Hannenhalli, "Fast sorting by reversals," in *Proc. 7th Annu. Symp. Combinatorial Pattern Matching*, Jun. 1996, pp. 168–185.

[12] P. Berman, S. Hannenhalli, and M. Karpinski, "Electronic Colloq. Computational Complexity," ECCC Rep. TR01-47 2001.

[13] P. Berman and M. Karpinski, "On some tighter inapproximability results," in *Proc. 26th Int. Colloq. Automata, Languages, and Programming*, Jul. 1999, pp. 200–209.

[14] M. Blanchette, "Evolutionary puzzles: An introduction to genome rearrangement," in *Proc. 9th Int. Conf. Conceptual Structures*, 2001, pp. 1003–1011.

[15] M. Blanchette, G. Bourque, and D. Sankoff, "Breakpoint phylogenies," in *Proc. Genome Informatics 1997*, 1997, pp. 25–34.

[16] M. Blanchette, T. Kunisawa, and D. Sankoff, "Parametric genome rearrangement," *J. Comput. Bio.*, vol. 172, pp. 11–17, 1996.

[17] ——, "Gene order breakpoint evidence in animal mitochondrial phylogeny," *J. Mol. Evol.*, vol. 49, pp. 193–203, 1999.

[18] D. Bryant, "The complexity of the breakpoint median problem" Univ. Montréal Center de Recherches Mathematiques, Montreal, QC, Canada, Tech. Rep. CRM-2597, 1998.

[19] ——, "The complexity of calculating exemplar distances," in *Comparative Genomics: Empirical and Analytical Approaches to Gene Order Dynamics, Map Alignment and the Evolution of Gene Families*: (Series in Computational Biology), D. Samkoff and J. H. Nadeau, Eds. Norwell MA: Kluwer, 2000, vol. 1, pp. 207–211

[20] ——, "A lower bound for the breakpoint phylogeny problem," in *Proc. 11th Symp. Combinatorial Pattern Matching*, Jun. 2000, pp. 235–247.

[21] A. Caprara, "Sorting by reversals is difficult," *Proc. 1st Annu. Int. Conf. Research in Computational Molecular Biology*, pp. 75–83, Jan. 1997.

[22] ——, "Formulations and hardness of multiple sorting by reversals," *Proc. 3rd Annu. Int. Conf. Research in Computational Molecular Biology*, pp. 84–93, Apr. 1999.

[23] ——, "Sorting permutations by reversals and Eulerian cycle decomposition," *SIAM J. Discr. Math.*, vol. 12, no. 1, pp. 91–110, 1999.

[24] ——, "On the practical solution of the reversal median problem," in *Proc. 1st Workshop Algorithms in Bioinformatics*, Aalborg, Denmark, Aug. 2001, pp. 238–251.

[25] A. Caprara, G. Lancia, and S. K. Ng, "Fast practical solution of sorting by reversals," in *Proc. 11th Annu. ACM-SIAM Symp. Discrete Algorithms*, Jan. 2000, pp. 12–21.

[26] T. Chen and S. Skiena, "Sorting with fixed-length reversals," *Discr. Appl. Math.*, vol. 71, pp. 269–295, 1996.

[27] D. A. Christie, "On the problem of sorting burnt pancakes," *Discr. Appl. Math.*, vol. 61, pp. 105–120, 1995.

[28] ——, "Sorting permutations by block-interchanges," *Inf. Process. Lett.*, vol. 60, pp. 165–169, 1996.

[29] ——, "A 3/2-approximation algorithm for sorting by reversals," in *Proc. 9th Annu. ACM-SIAM Symp. Discrete Algorithms*, Jan. 1998, pp. 244–252.

[30] ——, "Genome rearrangement problems," Ph.D. dissertation, Dept. Comput. Sci., Glasgow Univ., Glasgow, U.K., 1998.

[31] D. A. Christie and R. W. Irving, "Sorting strings by reversals and by transpositions," *SIAM J. Discr. Math.*, vol. 14, no. 2, pp. 193–206, 2001.

[32] N. G. Copeland, N. A. Jenkins, D. J. Gilbert, J. T. Eppig, L. J. Maltals, J. C. Miller, W. F. Dietrich, A. Weaver, S. E. Lincoln, R. G. Steen, J. H. Nadeau, and E. S. Lander, "A genetic linkage map of the mouse: Current applications and future prospects," *Science*, vol. 262, pp. 57–65, 1993.

[33] M. E. Cosner, R. K. Jansen, B. M. E. Moret, L. A. Raubeson, L. S. Wang, T. Warnow, and S. Wyman, "An empirical comparison of phylogenetic methods on chloroplast gene order data in campanulaceae," in *Comparative Genomics: Empirical and Analytical Approaches to Gene Order Dynamics, Map Alignment and the Evolution of Gene Families* (Series in Computational Biology), D. Samkoff and J. H. Nadeau, Eds. Norwell, MA: Kluwer, 2000, vol. 1, pp. 99–121.

[34] ——, "A new fast heuristic for computing the breakpoint phylogeny and experimental phylogenetic analyses of real and synthetic data," in *Proc. 8th Int. Conf. Intelligent Systems for Molecular Biology*, Aug. 2000, pp. 104–115.

[35] B. DasGupta, T. Jiang, S. Kannan, M. Li, and Z. Sweedyk, "On the complexity and approximation of syntenic distance," in *Proc. 1st Annu. Int. Conf. Research in Computational Molecular Biology*, Jan. 1997, pp. 99–108.

[36] ——, "On the complexity and approximation of syntenic distance," *Discr. Appl. Math.*, vol. 88, no. 1–3, pp. 59–82, 1998.

[37] Z. Dias and J. Meidanis, "Genome rearrangements distance by fusion, fission, and transposition is easy," in *Proc. 8th Int. Symp. String Processing and Information Retrieval*, Nov. 2001, pp. 250–253.

[38] T. Dobzhansky and A. H. Sturtevant, "Inversions in the chromosomes of *drosophila pesudoobscura*," *Genetics*, vol. 23, pp. 28–64, 1938.

[39] H. Dweighter, "Problem e2569," *Amer. Math. Mon.*, vol. 82, p. 1010, 1975.

[40] J. Ehrlich, D. Sankoff, and J. H. Nadeau, "Synteny conservation and chromosome rearrangements during mammalian evolution," *Genetics*, vol. 147, pp. 289–296, 1997.

[41] N. El-Mabrouk, "Genome rearrangement by reversals and insertions/deletions of contiguous segments," in *Proc. 11th Annu. Symp. Combinatorial Pattern Matching*, Jun. 2000, pp. 222–234.

[42] N. El-Mabroul and D. Sankoff, "Hybridization and genome rearrangement," in *Proc. 10th Annu. Symp. Combinatorial Pattern Matching*, Jul. 1999, pp. 78–87.

[43] N. Eriksen, "$(1+\epsilon)$-approximation of sorting by reversals and transpositions," in *Proc. 1st Workshop on Algorithms in Bioinformatics*, Aug. 2001, pp. 227–237.

[44] S. Even and O. Goldreich, "The minimum-length generator sequence problem is np-hard," *J. Alg.*, vol. 2, pp. 311–313, 1981.

[45] V. Ferretti, J. H. Nadeau, and D. Sankoff, "Original synteny," *Proc. 7th Annu. Symp. Combinatorial Pattern Matching*, pp. 159–167, Jun. 1996.

[46] W. H. Gates and C. H. Papadimitriou, "Bounds for sorting by prefix reversal," *Discr. Math.*, vol. 27, pp. 47–57, 1979.

[47] Q. P. Gu, K. Iwata, S. Peng, and Q. M. Chen, "A heuristic algorithm for genome rearrangements," in *Proc. Genome Inf. 1997*, 1997, pp. 268–269.

[48] Q. P. Gu, S. P. Peng, and H. Sudborough, "A 2-approximation algorithm for genome rearrangements by reversals and transpositions," *Theor. Comput. Sci.*, vol. 210, pp. 327–339, 1999.

[49] S. A. Guyer, L. S. Heath, and J. P. C. Vergara, "subsequence and run heuristics for sorting by transpositions," Dept. Comput. Sci., Virginia Polytechnic Inst. State Univ., Blacksburg, VA. Tech. Rep. TR 97-20, 1997.

[50] E. Gyori and E. Turan, "Stack of pancakes," *Studia Scientiarum Mathematicarum Hungarica*, vol. 13, pp. 133–137, 1978.

[51] S. Hannenhalli, "Polynomial algorithm for computing translocation distance between genomes," in *Proc. 6th Annu. Symp. Combinatorial Pattern Matching*, Jul. 1995, pp. 162–176.

[52] ——, "Software for computing inversion distance between signed gene orders," 1995. [Online]. Available: http://www-hto. usc.edu/plain/people/Hannenhalli.html

[53] ——, "Polynomial-time algorithm for computing translocation distance between genomes," *Discr. Appl. Math.*, vol. 71, pp. 137–151, 1996.

[54] S. Hannenhalli, C. Chappey, E. Koonin, and P. Pevzner, "Algorithms for genome rearrangements: Herpesvirus evolution as a test case," in *Proc. 3rd Int. Conf. Bioinformatics and Complex Genome Analysis*, 1994.

[55] ——, "Genome sequence comparison and scenarios for gene rearrangements: A test case," *Genomics*, vol. 30, pp. 299–311, 1995.

[56] S. Hannenhalli and P. Pevzner, "Towards a computational theory of genome rearrangement," in *Lecture Notes in Computer Science*, vol. 1000, Berlin, Germany: Springer-Verlag, 1995, pp. 184–202.

[57] ——, "Transforming cabbage into turnip (polynomial algorithm for sorting signed permutations by reversals)," in *Proc. 27th Annu. ACM Symp. Theory of Computing*, 1995, pp. 178–189.

[58] ——, "Transforming men into mice: Polynomial algorithm for genomic distance problem," in *Proc. 36th Annu. IEEE Symp. Foundations of Computer Science*, 1995, pp. 581–592.

[59] ——, "To cut . . . or not to cut (applications of comparative physical maps in molecular evolution)," in *Proc. 7th Annu. ACM-SIAM Symp. Discrete Algorithms*, Jan. 1996, pp. 304–313.

[60] ——, "Transforming cabbage into turnip: Polynomial algorithm for sorting signed permutations by reversals," *J. ACM*, vol. 46, no. 1, pp. 1–27, 1999.

[61] L. S. Heath and J. P. C. Vergara, "Some experiments on the sorting by reversals problem," Comput. Sci., Virginia Polytech. Inst. State Univ., Blacksburg, VA, Tech. Rep. TR 95-16, 1995.

[62] ——, "Sorting by bounded block-moves," *Discr. Appl. Math.*, vol. 88, pp. 181–206, 1998.

[63] ——, "Sorting by short block-moves," *Algorithmica*, vol. 28, pp. 323–352, 2000.

[64] M. H. Heydari and I. H. Subdorough, "On the diameter of the pancake network," *J. Alg.*, vol. 25, no. 1, pp. 67–94, 1997.

[65] ——, "On sorting by prefix reversals and the diameter of pancake networks," in *Proc. Heinz Nixdorf Symp. Parallel Algorithms and Architectures*, 1992, pp. 218–227.

[66] I. Holyer, "The np-completeness of some edge partition problem," in *SIAM J. Comput.*, vol. 10, 1981, pp. 713–717.

[67] M. Jerrum, "The complexity of finding minimum-length generator sequences," *Theor. Comput. Sci.*, vol. 36, pp. 265–289, 1985.

[68] T. Jiang, Y. Xu, and M. Q. Zhang, *Current Topics in Computational Molecular Biology,* Bejing, China, Tsinghua Univ. Press/Cambridge, MA, MIT Press, 2002.

[69] H. Kaplan, R. Shamir, and R. E. Tarjan, "A faster and simpler algorithm for sorting signed permutations by reversals," in *Proc. 8th Annu. ACM-SIAM Symp. Discrete Algorithms*, Jan. 1997, pp. 344–351.

[70] S. Karlin, E. S. Mocarski, and G. A. Schachtel, "Molecular evolution of herpesviruse: Genomic and protein sequence comparisons," *J. Virol.*, vol. 68, pp. 1886–1902, 1994.

[71] J. Kececioglu and R. Ravi, "Of mice and men: Algorithms for evolutionary distances between genomes with translocation," in *Proc. 6th Annu. ACM-SIAM Symp. Discrete Algorithms*, Jan. 1995, pp. 604–613.

[72] J. Kececioglu and D. Sankoff, "Exact and approximation algorithms for the inversion distance between two chromosomes," in *Proc. 4th Annu. Symp. Combinatorial Pattern Matching*, Jun. 1993, pp. 87–105.

[73] ——, "Exact and approximation algorithms for sorting by reversals, with application to genome rearrangement," *Algorithmica*, vol. 13, pp. 180–210, 1995.

[74] ——, "Efficient bounds for oriented chromosome inversion distance," in *Proc. 5th Annu. Symp. Combinatorial Pattern Matching*, Jun. 1994, pp. 307–325.

[75] J. Kleinberg and D. Liben-Nowell, "The syntenic dismeter of the space of n-chromosome genomes," in *Comparative Genomics: Gene Order Dynamics, Map Alignment and the Evolution of Gene Families* (Series in Computational Biology), D. Samkoff and J. H. Nadeau, Eds. Norwell MA: Kluwer, 2000, vol. 1, pp. 187–197.

[76] D. Liben-Nowell, "On the structure of syntenic distance," in *Proc. 10th Annu. Symp. Combinatorial Pattern Matching*, Jul. 1999, pp. 50–65.

[77] ——, "Gossip is synteny: Incomplete gossip and an exact algorithm for syntenic distance," in *Proc. 12th Annu. ACM-SIAM Symp. Discrete Algorithms*, Jan. 2001, pp. 177–185.

[78] D. Liben-Nowell and J. Kleinberg, "Structural properties and tractability results for linear synteny," in *Proc. 11th Annu. Symp. Combinatorial Pattern Matching*, Jun. 2000, pp. 248–263.
[79] G. H. Lin and G. L. Xue, "Signed genome rearrangement by reversals and transpositions: Models and approximations," in *Proc. 5th Annu. Int. Computing and Combinatorics Conf.*, Jul. 1999, pp. 71–80.
[80] I. Mantin and R. Shamir, *Applet: An Algorithm for Sorting Signed Permutations by Reversals,* (1999) [Online]. Available: http://www.math.tau.ac.il/~rshamir/GR/
[81] D. J. McGeoch, "Molecular evolution of large DNA viruses of eukaryotes," *Seminars Virol.*, vol. 3, pp. 399–408, 1992.
[82] B. M. E. Moret, L. S. Wang, T. Warnow, and S. K. Wyman, "New approaches for reconstructing phylogenies from gene order data," *Bioinformatics*, vol. 17, pp. S165–S173, 2001.
[83] B. M. E. Moret, S. Wyman, D. A. Bader, T. Warnow, and M. Yan, "A new implementation and detailed study of breakpoint analysis," in *Proc. 6th Pacific Symp. Biocomputing*, Jan. 2001, pp. 583–594.
[84] J. H. Nadeau and D. Sankoff, "Comparable rates of gene loss and functional divergence after genome duplications early in vertebrate evolution," *Genetics*, vol. 147, pp. 1259–1266, 1997.
[85] ——, "Counting on comparative maps," *Trends Genetics*, vol. 14, no. 12, pp. 495–501, 1998.
[86] ——, "The lengths of undiscovered conserved segments in comparative maps," *Mammalian Genome*, vol. 9, pp. 491–495, 1998.
[87] J. H. Nadeau and B. A. Taylor, "Lengths of chromosomal segments conserved since divergence of man and mouse," *Proc. Nat. Acad. Sci.*, vol. 81, pp. 814–818, 1984.
[88] S. J. O'Brien, *Genetics Maps: Locus Maps of Complex Genomes*, 6th ed. Cold Spring Harbor, ME: Cold Spring Harbor Lab. Press, 1993.
[89] S. J. O'Brien, J. E. Womack, L. A. Lyons, K. J. Moore, N. A. Jenkins, and N. G. Copeland, "Anchored reference loci for comparative genome mapping in mammals," *Nature Genetics*, vol. 3, pp. 103–112, 1993.
[90] J. D. Palmer and L. A. Herbon, "Plant mitochondrial DNA evolves rapidly in structure, but slowly in sequence," *J. Mol. Evol.*, vol. 28, pp. 87–97, 1988.
[91] I. Pe'er and R. Shamir, "The median problems for breakpoints are np-complete," Electronic Colloquium. Computational Complexity, ECCC Rep. TR98-071, 1998.
[92] ——, "Approximation algorithms for the median problem in the breakpoint model," in *Comparative Genomics: Gene Order Dynamics, Map Alignment and the Evolution of Gene Families*: (Series in Computational Biology), D. Samkoff and J. H. Nadeau, Eds. Norwell MA: Kluwer, 2000, vol. 1, pp. 225–241
[93] P. Pevzner and M. Waterman, "Open combinatorial problems in computational molecular biology," in *Proc. 3rd Israel Symp. Theory of Computing and Systems*, 1995, pp. 148–173.
[94] P. A. Pevzner, *Computational Molecular Biology: An Algorithmic Approach*. Cambridge, MA: MIT Press, 2000.
[95] N. Pisanti and M. F. Sagot, "Further thoughts on the syntenic distance between genomes," *Algorithmica*, vol. 34, no. 2, pp. 157–180, 2002.
[96] D. Sankoff, "Edit distance for genome comparison based on non-local operations," in *Proc. 3rd Annu. Symp. Combinatorial Pattern Matching*, 1992, pp. 121–135.
[97] ——, "Genome rearrangement with gene families," *Bioinformatics*, vol. 15, no. 11, pp. 909–917, 1999.
[98] D. Sankoff and M. Blanchette, "The median problem for breakpoints in comparative genomics," in *Proc. 3rd Annu. Int. Computing and Combinatorics Conf.*, Aug. 1997, pp. 251–264.
[99] ——, "Multiple genome rearrangement," in *Proc. Annu. Int. Conf. Research in Computational Molecular Biology*, Mar. 1998, pp. 243–247.
[100] ——, "Multiple genome rearrangement and breakpoint phylogeny," *J. Comput. Biol.*, vol. 5, pp. 555–570, 1998.
[101] ——, "Comparative genomics via phylogenetic invariants for jukes-cantor semigroups," in *Proc. Int. Conf. Stochastic Models*, Ottawa, ON, Canada, L. Grostiza and G. Ivanoff, 1999.
[102] ——, "Phylogenetic invariants for genome rearrangement," *J. Comput. Biol.*, vol. 6, no. 3/4, pp. 431–445, 1999.
[103] ——, "Phylogenetic invariants for metazoan mitochondrial genome evolution," *Genome Inf. 1999*, pp. 22–23, 1999.
[104] ——, "Probability models for genome rearrangement and linear invariants for phylogenetic inference," in *Proc. 3rd Annu. Int. Conf. Research in Computational Molecular Biology*, Apr. 1999, pp. 302–309.
[105] D. Sankoff, D. Bryant, M. Deneault, B. F. Lang, and G. Burger, "Early eukaryote evolution based on mitochondrial gene order breakpoints," in *Proc. 4th Annu. Int. Conf. Research in Computational Molecular Biology*, Apr. 2000, pp. 254–262.
[106] ——, "Early eukaryote evolution based on mitochondrial gene order breakpoints," *J. Comput. Biol.*, vol. 7, no. 3/4, pp. 521–535, 2000.
[107] D. Sankoff, R. J. Cedergren, and G. Lapalme, "Frequency of insertion-deletion, transversion, and transposition in the evolution of 5s ribosomal RNA," *J. Mol. Evol.*, vol. 7, pp. 133–149, 1976.
[108] D. Sankoff and N. El-Mabrouk, "Genome rearrangement," in *Current Topics in Computational Molecular Biology*, T. Jiang, Y. Xu, and Q. Zhang, Eds. Cambridge, MA: MIT Press, 1992, pp. 132–155.
[109] D. Sankoff, V. Ferretti, and J. H. Nadeau, "Conserved segment identification," in *Proc. 1st Annu. Int. Conf. Research in Computational Molecular Biology*, Jan. 1997, pp. 252–256.
[110] D. Sankoff, G. Leduc, N. Antoine, B. Paquin, B. F. Lang, and R. Cedergen, "Gene order comparisons for phylogenetic inference: Evolution of the mitochondrial genome," in *Proc. Nat. Acad. Sci.*, vol. 89, 1992, pp. 6575–6579.
[111] D. Sankoff and J. H. Nadeau, "Conserved synteny as a measure of genomic distance," *Discr. Appl. Math.*, vol. 71, pp. 247–257, 1996.
[112] ——, *Comparative Genomics: Empirical and Analytical Approaches to Gene Order Dynamics, Map Alignment and the Evolution of Gene Families* (Series in Computational Biology). Norwell MA: Kluwer, 2000, vol. 1, pp. 225–241.
[113] D. Sankoff, M. N. Parent, I. Marchand, and V. Ferretti, "On the Nadeau-Taylor theory of conserved chromosome segments," in *Proc. 8th Annu. Symp. Combinatorial Pattern Matching*, 1997, pp. 262–274.
[114] D. Sankoff, G. Sundaram, and J. Kececioglu, "Steiner points in the space of genome rearrangements," *Int. J. Foundations Comput. Sci.*, vol. 7, no. 1, pp. 1–9, 1996.
[115] C. Siepel, "An algorithm to enumerate all sorting reversals," in *Proc. 6th Annu. Int. Conf. Computational Biology*, Washington, DC, Apr. 18–21, 2002, pp. 281–290.
[116] A. C. Siepel and B. M. E. Moret, "Finding an optimal inversion median: Experimental results," in *Proc. 1st Workshop Algorithms in Bioinformatics*, Aug. 2001, pp. 189–203.
[117] A. H. Sturtevant and T. Dobzhansky, "Inversions in the third chromosome of wild races of *drosophila pseudoobsura*, and their use in the study of the history of the species," in *Proc. Nat. Acad. Sci.*, vol. 22, 1936, pp. 448–450.
[118] R. E. Tarjan, "Efficiency of a good but not linear set union algorithm," *J. ACM*, vol. 22, no. 2, pp. 215–225, 1975.
[119] G. Tesler, "Grimm: Genome rearrangements web server," *Bioinformatics*, vol. 18, pp. 492–493, 2002.
[120] N. Tran, "An easy case of sorting by reversals," in *Proc. 8th Annu. Symp. Combinatorial Pattern Matching*, 1997, pp. 83–89.
[121] M. E. M. T. Walter, Z. Dias, and J. Meidanis, "Reversal and transposition distance of linear chromosomes," in *Proc. 5th South Amer. Symp. String Processing and Information Retrieval*, Sep. 1998, pp. 96–102.
[122] L. S. Wang, "Exact-iebp: A new technique for estimating evolutionary distances between whole genomes," in *Proc. 1st Workshop Algorithms in Bioinformatics*, Aug. 2001, pp. 175–188.
[123] L. S. Wang and T. Warnow, "Estimating true evolutionary distances between genomes," in *Proc. 33rd Annu. ACM Symp. Theory of Computing*, Jul. 2001, pp. 637–646.
[124] M. S. Waterman, *Introduction to Computational Biology*. London, U.K.: Chapman and Hall, 1995.
[125] G. A. Watterson, W. J. Ewens, T. E. Hall, and A. Morgan, "The chromosome inversion problem," *J. Theor. Biol.*, vol. 99, pp. 1–7, 1982.
[126] D. M. Zhu and S. H. Ma, "Improved polynomial-time algorithm for computing translocation distance between genomes," *Chinese J. Comput.*, vol. 25, no. 2, pp. 189–196, 2002, In Chinese.

Zimao Li received the B.S. degree in mathematics and the M.Eng. degree in computer spaces from the Shandong University, Shandong, China, in 1996 and 1999, respectively, and the Ph.D. degree in computer science from the City University of Hong Kong, Hong Kong, in 2002.

His current research interests are design and analysis of algorithms, complexity theory, and computational biology.

Lusheng Wang (M'02) received the Ph.D. degree from McMaster University, Hamilton, ON, Canada, in 1995.

Currently, he is an Associate Professor in the Department of Computer Science, City University of Hong Kong. His research interests include algorithms, bioinformatics, and computational biology.

Kaizhong Zhang (M'91) received the M.S. degree in mathematics from Peking University, Beijing, China, in 1981, and the M.S. and Ph.D. degrees in computer science from the Courant Institute of Mathematical Sciences, New York University, New York, in 1986 and 1989, respectively.

He is currently a Full Professor in the Department of Computer Science, University of Western Ontario, London, ON, Canada. His research interests include bioinformatics, algorithms, image processing, and databases.