

Genomic Sorting with Length-Weighted Reversals

Ron Y. Pinter¹

pinter@cs.technion.ac.il

Steven Skiena²

skiena@cs.sunysb.edu

- ¹ Department of Computer Science, Technion – Israel Institute of Technology, Haifa 32000, Israel
² Department of Computer Science, State University of New York at Stony Brook, Stony Brook, NY 11794-4400, USA

Abstract

Current algorithmic studies of genome rearrangement ignore the length of reversals (or inversions); rather, they only count their number. We introduce a new cost model in which the lengths of the reversed sequences play a role, allowing more flexibility in accounting for mutation phenomena. Our focus is on sorting unsigned (unoriented) permutations by reversals; since this problem remains difficult (NP-hard) in our new model, the best we can hope for are approximation results. We propose an efficient, novel algorithm that takes (a monotonic function f of) length into account as an optimization criterion and study its properties. Our results include an upper bound of $O(f(n) \lg^2 n)$ for any additive cost measure f on the cost of sorting any n -element permutation, and a guaranteed approximation ratio of $O(\lg^2 n)$ times optimal for sorting a given permutation. Our work poses some interesting questions to both biologists and computer scientists and suggests some new bioinformatic insights that are currently being studied.

Keywords: genome rearrangements, sorting by reversal, length sensitive cost

1 Introduction

Computing the minimum-cost reversal distance between two (genomic) sequences is a problem of considerable importance in comparative genomics, e.g. when reconstructing the evolutionary history of an organism. Reversal (or inversion) mutations occur often in chromosomes, where each reverses the order of genes and/or their components (exons) within a certain interval. The most parsimonious series of reversals transforming one sequence to another corresponds to a likely evolutionary path between them. This analysis has been applied, for example, to drosophila [11, 20], plants [3, 17], viruses [12], and mammals [10, 19].

Traditionally [2, 15], such analysis assumes that each reversal is of unit cost independent of the length of the fragment reversed. This assumption is not completely defensible biologically: a longer genomic reversal will clearly cause more upheaval to the organism, making less likely the organism survives to pass on the mutation. Further, the mechanics of genome reversal may suggest that the probabilities of such events are likely to be dependent on fragment length.

In this paper, we consider algorithms to efficiently sort one sequence into another by reversals under length sensitive cost models. Let the function $f(l)$ denote the cost of a reversal of length l . Under the traditional model, each reversal is of unit cost independent of length, so $f(l) = 1$. We say a function $f(l)$ is *additive* if $f(x) + f(y) = f(x + y)$, *subadditive* if $f(x) + f(y) > f(x + y)$, and *superadditive* if $f(x) + f(y) < f(x + y)$. We will show that distinct sorting by reversal algorithms dominate each of these three classes of cost function.

We present results of two kinds¹:

¹We use the notation $\lg n$ to denote $\log_2 n$.

- First, we consider the problem of minimizing the cost sufficient to sort *any* permutation of n elements. This is equivalent to computing the diameter, under the shortest-path metric, of the *reversal graph*, namely the weighted graph where the vertices represent the length- n permutations and there is an edge (p_1, p_2) of weight $f(l)$ if there exists one l -reversal transforming p_1 to p_2 . Most notably, we give an interesting algorithm proving a graph diameter of $O(f(n) \lg^2 n)$ for additive cost measures.
- Second, we consider the problem of approximating the minimum-cost reversal sequence for a given permutation (of n elements). We seek a heuristic guaranteeing that the resulting sequence costs no more than some slowly growing function of n times that of the optimal sequence. We show that a popular greedy heuristic can yield solutions costing $\Omega(n / \lg^2 n)$ times optimal, but give a heuristic which yields sequences of cost $O(\lg^2 n)$ times optimal — both for additive functions.

We note that the relatively coarse bounds generated by our techniques require extra care when applying them to biological data. However, we see the importance of our work as a first attempt to seriously consider more general reversal cost functions, demonstrating that they lead to interesting algorithmic results and suggesting that they serve as a basis for some interesting bioinformatic studies. This issue is discussed further in our Conclusions (Section 5).

This paper is organized as follows. In Section 2, we summarize previous work in minimum-cost sorting with reversals. In Section 3, we present our results on minimizing the cost sufficient to sort any permutation. In Section 4, we consider the problem of approximating the minimum-cost reversal sequence for a given permutation. We conclude in Section 5 with a discussion of further research and open problems.

2 Previous Work

The problem of computing the reversal distance of a permutation and its applications to comparative genomics has received extensive attention over the last decade. The elements of the permutation denote indivisible portions of the original sequence, representing genomic entities such as exons or whole genes. There are two variants of the problem: the *unsigned* case, in which we disregard the orientation of the elements throughout the reversal process, and the *signed* case, where the directions of the elements do matter. Each case has some pros and cons in terms of the underlying biology; in this paper we focus on the unsigned case.

For the case of unit-cost, unsigned reversals, the problem of computing the reversal distance has been shown to be NP-complete by Caprara [7]; our problem, in which the cost depends on the length of the subsequence being reversed, inherits hardness under more general metrics from this result. Kececloglu and Sankoff [15] gave approximation algorithms on reversal distance that guarantee a ratio at most 2 times optimal, which Bafna and Pevzner [2] improved to a factor of $7/4$ approximation; recently, Berman *et al.* [5] reduced this factor even further, to 1.375. Kececloglu and Sankoff [16] report on the success of heuristics and search in determining reversal distance for chromosomes.

For sake of completeness we mention that Hannenhalli and Pevzner [13] gave a polynomial-time algorithm for the case of unit-cost, signed reversals. An elementary exposition of the Hannenhalli-Pevzner theory appears in [4].

Minimum-cost unsigned reversal sorting has also been studied from the other end of the cost spectrum, under models where the cost increases so dramatically with length that only length-2 reversals can be afforded. Thus each reversal simply transposes adjacent elements. Bubble sort and insertion sort [18] sort any permutation π using exactly one transposition for each inversion in π , thus minimizing the number of reversals. Jerrum [14] presented a polynomial-time algorithm for the much more difficult problem of sorting circular permutations using a minimum number of transpositions.

Chen and Skiena [8] study the problem of sorting permutations and circular permutations using *fixed-length* reversals, giving an algorithm to sort all sortable circular n -permutations using $O(n^2/k + kn)$ k -reversals, while showing there exist permutations requiring $\Omega(n^2/k^2 + n)$ k -reversals to sort. This algorithm thus yields a $O(f(k) \cdot (n^2/k + kn))$ cost-diameter bound for any cost function $f(k)$. The algorithms in this paper will yield substantially tighter bounds over a wide range of cost functions.

Finally, some preliminary results of genome rearrangements that assign a length dependent cost to reversal operations are reported in [6]. Experiments were made both on the mitochondrial genomes of two fungi as well as on randomly generated samples. They do indicate that length could play an important role in biasing certain rearrangement patterns, but they are still somewhat inconclusive regarding the exact cost function that is to be used. Their work has been extended recently by Ajana *et al.* [1], allowing users of a (signed) reversal algorithm to choose one or several possible solutions based on different criteria; this flexibility was shown to be useful for testing certain reversal hypotheses.

3 Bounding the Diameter of the Reversal Graph

Recall that the reversal graph for n -element permutations is the weighted graph where the vertices represent the permutations and there is an edge (p_1, p_2) of weight $f(s)$ if there exists one reversal of a subsequence of elements of length s transforming p_1 to p_2 . By bounding the diameter of the reversal graph, we establish an upper bound on the cost of sorting any n -element permutation. Standard sorting algorithms exhibit interesting performance on highly subadditive and superadditive functions, but not additive measures. The primary result of this section is a new reversal-based sorting algorithm which performs well on additive cost functions.

First, we analyze standard sorting algorithms under reversal cost. We note that a reversal-based version of selection sort performs at most $n - 1$ reversals, a fraction of which are potentially $\theta(n)$ in length. Thus selection sort gives an $O(nf(n))$ diameter algorithm, which is efficient for subadditive functions, particularly $f(s) = 1$ for all s .

Bubblesort and insertion sort perform transpositions of neighboring elements, one for each inversion in the input permutation. This gives an $O(n^2f(2))$ diameter algorithm, which is efficient for highly superadditive functions, particularly $f(s) \geq s^2$.

Now we turn to the most interesting case, which is that of additive functions, particularly $f(s) = s$. For such cost functions, we give an algorithm to sort any permutation of n elements in $O(f(n) \lg^2 n)$ cost using divide and conquer.

The key operation is *MedianEject*. Consider a permutation p on $\{1, \dots, n\}$. Sorting p involves putting element i in position i , for all $1 \leq i \leq n$. Let $p(i)$ denote the element in the i th position of p , and $p^{-1}(i)$ denote the position of element i in p . We say an element x is *wrong-sided* if x and $p^{-1}(x)$ are on different sides of the median $n/2$, i.e. $x \leq n/2$ and $p^{-1}(x) > n/2$ or $x > n/2$ and $p^{-1}(x) \leq n/2$.

We apply *MedianEject* to portions of the permutation, ranging from position a to position b . One round of *MedianEject* moves all wrong-sided elements in the interval $[a, b]$ to the correct side relative to its median in the following manner:

MedianEject(a, b) =
 identify the r maximal runs of wrong-sided elements w.r.t. median $(b - a)/2$.
 for ($i = 1$ to $\lg r$)
 reduce the number of wrong-sided runs by half using
 non-overlapping reversals, none crossing the median.
 with 2 reversals, move remaining wrong-sided runs to median boundary.
 reverse the left and right wrong-sized runs using a single reversal.

Lemma 1 *MedianEject costs $O(f(b - a) \lg r)$ for any additive cost function f .*

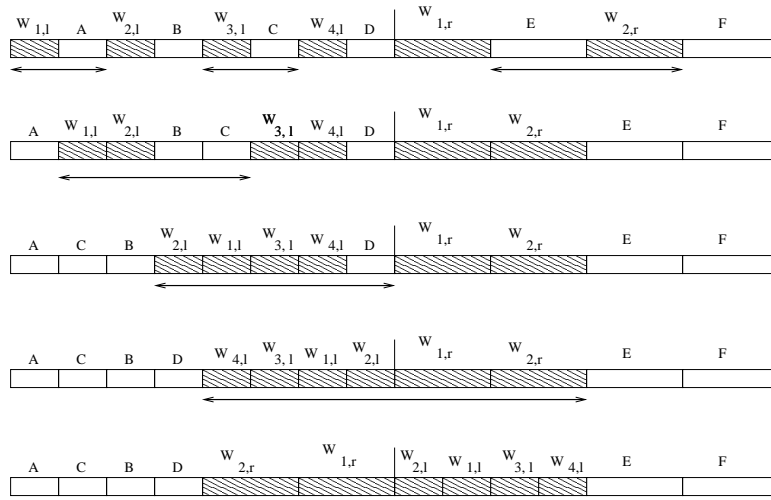


Figure 1: A sample run of MedianEject; orientation of runs not shown.

Proof: We use the following notation. Partition the wrong-sided elements of p into r_l runs of adjacent wrong-sized elements on the left of the median, and r_r runs of adjacent wrong-sized elements on the right. Denote the i th such run on the left (right) as $w_{i,l}$ ($w_{i,r}$), and let $r = \max(r_l, r_r)$.

The reversals employed in MedianEject can be precisely described as follows. In each round, we will perform $\lfloor r_l/2 \rfloor + \lfloor r_r/2 \rfloor$ reversals. The reversals on the left of the median start with the first element of $w_{2i-1,l}$ and end immediately to the left of $w_{2i,l}$, thus merging $w_{2i-1,l}$ and $w_{2i,l}$ into one wrong-sided run. The reversals on the right of the median are analogously defined.

Once each side contains only a single wrong-sided run, we move these runs to the median boundary using non-overlapping reversals. As the number of wrong-sided elements x on each side must be equal, a final reversal of length $2x$ completes the partitioning. An example of MedianEject in action is provided in Figure 1.

In the i th of the $\lg r$ rounds comprising MedianEject, we perform up to $k = 2r/2^i$ non-overlapping reversals, of length s_j for reversal $1 \leq k \leq j$. Because the reversals are mutually non-overlapping, $\sum_{j=1}^k s_j \leq b-a$. Because f is additive, $\sum_{j=1}^k f(s_j) \leq f(b-a)$, and hence the total cost is $O(f(b-a) \lg r)$. ■

MedianEject is the partitioning operation of the following Quicksort-like algorithm:

```

ReversalSort( $a, b$ ) =
  MedianEject( $a, b$ )
  ReversalSort( $a, \lfloor (b-a)/2 \rfloor$ )
  ReversalSort( $\lceil (b-a)/2 \rceil, b$ )

```

Theorem 2 *ReversalSort runs in $O(f(n) \lg^2 n)$ time for any additive cost function $f(n)$.*

Proof: By the master theorem [9], the recurrence $T(n) = 2T(n/2) + O(f(n) \lg n)$ evaluates to $O(f(n) \lg^2 n)$. ■

At this time, we have no superlinear lower bound for either MedianEject or reversal sorting. However, we conjecture that the interleaved permutation $(n/2 + 1, 1, n/2 + 2, 2, \dots, n, n/2)$ requires $\theta(f(n) \lg n)$ cost to MedianEject and hence sort for additive cost functions.

4 Approximating Distance

From a biological point of view, constructing the least expensive transformation from a given permutation A to another permutation B is more interesting than minimizing diameter. This is because we want to reconstruct the evolutionary history from A and B , a history which presumably took the most parsimonious possible path. By relabeling the genes (or exons) in permutation A according to their position in B , yielding the permutation A' , we can reduce this problem to that of computing the minimum cost reversal sorting of A' .

The problem of computing the minimum cost transformation is NP-complete for the cost function $f(n) = 1$ [7], and presumably remains so under most other cost functions. Hence we consider the problem of approximating reversal cost for additive cost functions.

For the case of unit-cost reversals, a simple greedy breakpoint-merging heuristic comes within a constant factor of the optimal reversal cost [15]. In such a heuristic, we employ the reversal which creates the greatest number of adjacencies by merging runs of consecutive elements. If no such reversals exist, a single reversal which creates such a move is employed.

The natural generalization of this idea for additive cost functions is to take the shortest breakpoint-merging reversal, if one exists. Such a heuristic can yield very poor results, however:

Theorem 3 *The greedy breakpoint-merging heuristic can yield a reversal schedule whose cost is $\Omega(n/\lg^2 n)$ times optimal.*

Proof: Consider the following construction for $n = 17$, consisting of two runs of interleaved triples, one run that is increasing and properly oriented, the other run — reversed and decreasing:

1, 16, 15, 14, 2, 3, 4, 13, 12, 11, 5, 6, 7, 10, 9, 8, 17

There are only two breakpoint-removing reversals, one bonding 7-8 at cost 3, the other bonding 16-17 at cost 15. Repeatedly reversing the rightmost interval increases the resulting run by 3 elements per iteration, at a cost of $3i$ for the i th such reversal:

1, 16, 15, 14, 2, 3, 4, 13, 12, 11, 5, 6, 7, 8, 9, 10, 17

1, 16, 15, 14, 2, 3, 4, 13, 12, 11, 10, 9, 8, 7, 6, 5, 17

1, 16, 15, 14, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 17

1, 16, 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 17

1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17

The total cost of this strategy is $\sum_{i=1}^{n/3} 3i = n(n/3 - 1)/2 = \theta(n^2)$, as opposed to the $O(n \lg^2 n)$ cost achievable by Theorem 2.

Observe that the same cost is achieved by repeatedly employing the longer of the two breakpoint reversals. This construction is straightforward to generalize for larger n . ■

We now show that for all permutations, the reversal sorting algorithm yields a cost which is $O(\lg^2 n)$ times optimal for any additive cost function. Our analysis requires the definition of a weighted graph $G(p)$ associated with a given permutation p . The vertices of $G(p)$ will be the n elements (positions) of p . There will be an edge (i, j) in $G(p)$ where $i = p^{-1}(j)$ for all $1 \leq j \leq n$. The weight of this edge is $f(|i - j|)$.

The importance of this graph is in establishing a lower bound on the optimal cost of sorting. Both the number of out-of-position elements of p (i.e. the number of vertices of $G(p)$ incident on non-zero weight edges) and the distance of the single most out of position element (i.e. the cost of the maximum

weight edge) yield lower bounds on the optimal cost. However, such bounds can be very far from the optimal sorting cost. Consider p' , the permutation obtained from the identity permutation by making $\sqrt{n}/2$ element swaps as follows: for $1 \leq i \leq \sqrt{n}/2$, exchange elements $p'_{(2i-2)\sqrt{n}}$ and $p'_{(2i-1)\sqrt{n}}$. Such a permutation contains only \sqrt{n} displaced elements and a maximum edge of length \sqrt{n} , yet requires $\Omega(n)$ cost under the additive cost function $f(n) = n$.

Instead, we bound the optimal cost in terms of the weight of the heaviest non-crossing matching in $G(p)$. We say a matching $M(G(p))$ is non-crossing if for all pairs of edges $(i, j), (k, l) \in M(G(p))$, there does not exist an x such that $i \leq x \leq j$ and $k \leq x \leq l$. Such a maximal matching can easily be found using dynamic programming.

Lemma 4 *The weight of $M(G(p))$ is a lower bound on the reversal-sorting cost for permutation p under additive weight functions.*

Proof: Consider the simpler task of just placing the elements defining edges from $M(G(p))$ into their proper position. This task can be done in cost $f(w)$, where w is the total weight of $M(G(p))$, by performing the reversals defined by the edges in the matchings. Because none of the intervals overlap or nest, no longer reversal can be helpful to move multiple elements into the proper position; because the cost function is additive we cannot benefit by using shorter reversals. ■

To argue that the weight of $M(G(p))$ is a good lower bound, we will bound certain properties of p and $G(p)$ in the size of this matching.

Lemma 5 *Let E_k denote the k th edge of $M(G(p))$, where $1 \leq k \leq |M(G(p))|$. Let $\delta(E_k, i, j)$ be a function which equals 1 if E_k intersects the interval $[i, \dots, j]$ and is zero otherwise. Then edge $(i, j) \notin G(p)$ if*

$$|j - i| > \sum_{k=1}^{|M(G(p))|} f(E_k) \delta(E_k, i, j)$$

Proof: By definition, $M(G(p))$ is the maximum cost non-crossing matching. Hence such an edge (i, j) cannot exist in $G(p)$, for if so we could remove all intersected matching edges and insert (i, j) into $M(G(p))$ to yield a higher cost non-crossing matching. ■

Lemma 6 *The number of out-of-position elements in p is at most $3|M(G(p))|$.*

Proof: First, we note that $3|M(G(p))|$ out-of-position elements is the most achievable if $M(G(p))$ consists of widely-spaced edges. In such a matching, any graph edge intersecting matching edge E_k must have an endpoint at most $|E_k| - 1$ positions to the left or right of E_k .

We now consider the possible locations of out-of-position elements in the case of two nearby matching edges. Let a and b refer both to the names and lengths of the left and right segments, and d to the gap between them. There are two cases, shown in Figure 2. First, as in the case of widely-separated matching edges, the elements within a of the left segment or b of the right segment; although these intervals may now overlap, and hence the union size be reduced. Second, there are intervals to the left and right where edges may span a length $> \max(a, b)$ without being in the matching as they intersect both a and b . These intervals are denoted l and r , respectively. $d \leq a + b$, as otherwise the segments are widely-spaced. Then:

$$l \leq (a + b) - a - d \leq b - d$$

$$r \leq (a + b) - b - d \leq a - d$$

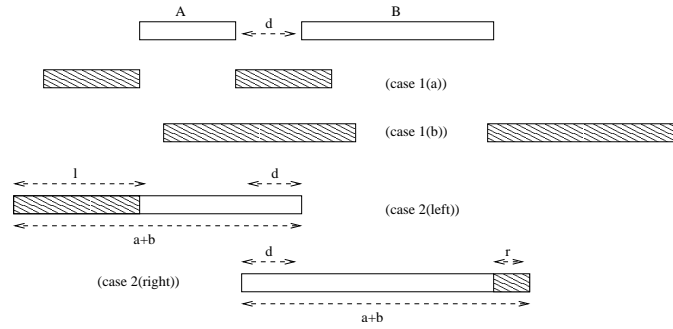


Figure 2: Penumbra of out-of-position elements cast by segments a and b .

The window of locations of out-of-position elements is of size

$$W = \max(l, a) + \max(r, b) + d = \max(b - d, a) + \max(a - d, b) + d$$

There are several cases. When $d \leq a + b$, the segments are widely-spaced and $W = 2a + 2b$. Assume wlog that $b \geq a$. When $b \leq d < a + b$, $W \leq a + b + d < 2a + 2b$. When $a \leq d < b$, $W \leq \max(b - d + a + b, a + 2b) < 2a + 2b$. When $a \leq a$, $W \leq \max(b - a, a) + b + a < 2a + 2b$. Thus this quantity is maximized for widely-spaced segments.

In the case of there being intermediate matching edges between a and b , we note that an edge spanning a and b must be of length less than the sum of all intersected matching edges by Lemma 5. In such an event, the intermediate matching edges can be collected and moved to border either a or b and reduce it to the two segment case. ■

Define the *penumbra* of $M(G(p))$ to be the set of positions where out-of-position elements potentially lie (as defined in the proof of Lemma 5) unioned with all positions overlapped by edges of $M(G(p))$.

Lemma 7 *No element outside of the penumbra moves during the execution of MedianEject.*

Proof: Note that the penumbra consists of disjoint intervals of elements. No edge in $G(p)$ can join elements in two disjoint intervals, because such an edge would imply the existence of a larger non-crossing matching.

We note that the MedianEject algorithm will never move an element outside of its given penumbra interval. If the median does not intersect a penumbra interval, then no reversals are performed by MedianEject and the permutation remains unchanged. If the median does intersect a penumbra interval, the only elements moving lie within this penumbra interval, so no other elements are effected. ■

Lemma 7 implies that every round of non-overlapping reversals costs at most $3|M(G(p))|$ throughout the execution of reversal-sort. Hence:

Corollary 1 *The cost of the each round of MedianEject is $O(|M(G)| \lg n)$, and therefore reversal sort costs $O(|M(G)| \lg^2 n)$.*

Theorem 8 *The reversal sort heuristic solution is at most a factor of $O(\lg^2 n)$ times the optimal transformation cost.*

5 Conclusions

In this paper, we have extended the sorting by reversals problem to non-unit cost models, provided a biological motivation for the model, and provided initial algorithmic results. Our work suggests several directions for future work, both theoretical and biological as well as the interactions thereof.

Theoretically, the most interesting open problems are improving our lower bound for diameter under additive cost functions and finding a better (constant factor?) approximation algorithm. It would also be interesting to study the effect on our methods of changes to the strictly additive cost models: consider both (mildly) sub- and super-additive models as well as not necessarily monotonic costs (see below). Another interesting issue is the study of length sensitive reversals in the signed case: the effects of length as cost on the Hannenhalli-Pevzner theory and its derivatives seem highly non-trivial.

Biologically, the interesting questions revolve around identifying the most realistic cost function for evolutionary analysis. It is known that chromosome breaks tend to occur more frequently at “hot spots”, as marked by certain conditions such as Fragile X. The proper cost models are likely to be somewhat species-specific; e.g. reversals crossing the centromere of a chromosome appear to be very rare in most species. Moreover, combining weight systems like ours with constraint systems (e.g. of allowed and forbidden reversal regions) seems like a formidable challenge.

The study of cost models gives rise to further interplay between the theoretical and biological aspects. In this paper, we have assumed that the cost of a reversal increases monotonically in length, and our algorithms significantly depend on this property. It may be that a bimodal cost function may be more biological realistic. Very short reversals probably appear rarely in evolution, as they are likely to either cause damage to the exons or effect only non-coding regions. Very long reversals likely occur rarely as well, as they typically will cross the centromere or effect many essential genes.

An interesting research direction would be to analyze the length distribution of reversal sequences produced by standard algorithms on whole genome data, as well as accepted, published reversal sequences. Such analysis would establish whether standard algorithms yield reversal sequences with similar properties, and whether there exist reversal sequences of similar length but widely varying cost.

Finally, in order to obtain a better understanding of the relationship between “natural” and “algorithmic” reversals, we are in the process of conducting a bioinformatic experiment in which we compute the distance among several whole genomes using various cost functions as a measure, and then comparing the results to other information, such as phylogenetic trees. A preliminary phase of this study is well underway and we shall present initial results at the workshop.

References

- [1] Y. Ajana, J. Lefebvre, E. Tillier, and N. El-Mabrouk. Exploring the set of all minimal sequences of reversals — an application to test the replication-directed reversal hypothesis. In *2nd International Workshop on Algorithms in Bioinformatics (WABI'02)*, LNCS 2452, pages 300–315, 2002.
- [2] V. Bafna and P. A. Pevzner. Genome rearrangements and sorting by reversals. In *34th IEEE Symposium on Foundations of Computer Science*, pages 148–157, 1993.
- [3] V. Bafna and P. A. Pevzner. Sorting by reversals: Genome rearrangements in plant organelles and evolutionary history of X chromosome. *Molecular Biology and Evolution*, 1994.
- [4] A. Bergeron. A very elementary presentation of the Hannenhalli-Pevzner theory. In *Proc. 12th Symp. Combinatorial Pattern Matching (CPM)*, volume 2089, pages 106–117. Springer-Verlag Lecture Notes in Computer Science, 2001.

- [5] P. Berman, S. Hannenhalli, and M. Karpinski. 1.375-approximation algorithm for sorting by reversals. In *10th European Symposium on Algorithms*, pages 200–210, 2002.
- [6] M. Blanchette, T. Kunisawa, and D. Sankoff. Parametric genome rearrangement. *Gene*, 172:GC 11–17, 1996.
- [7] A. Caprara. Sorting by reversals is difficult. In *Proc. First Annual International Conference on Computational Molecular Biology (RECOMB'97)*, pages 75–83. ACM Press, 1997.
- [8] T. Chen and S. Skiena. Sorting with fixed-length reversals. *Discrete Applied Mathematics*, 71:269–295, 1996.
- [9] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms*. MIT Press, Cambridge, MA, 1990.
- [10] M. Davisson. X-linked genetic homologies between mouse and man. *Genomics*, 1:213–227, 1987.
- [11] T. Dobzhansky and A.H.Sturtevant. Inversions in the chromosomes of *drosophila pseudoobscura*. *Genetics*, 23:28–64, 1938.
- [12] S. Hannenhalli, C. Chappey, E. Koonin, and P. A. Pevzner. Scenarios for genome rearrangements: Herpesvirus evolution as a test case. In *Proc. of 3rd Intl. Conference on Bioinformatics and Complex Genome Analysis*, 1994.
- [13] S. Hannenhalli and P. A. Pevzner. Transforming cabbage into turnip: Polynomial algorithm for sorting signed permutations by reversals. *J. ACM*, 46:1–27, 1999.
- [14] M. Jerrum. The complexity of finding minimum-length generator sequences. *Theoretical Computer Science*, 36:265–289, 1985.
- [15] J. Kececioğlu and D. Sankoff. Exact and approximation algorithms for the inversion distance between two permutations. In *Proc. of 4th Ann. Symp. on Combinatorial Pattern Matching*, Lecture Notes in Computer Science 684, pages 87–105. Springer Verlag, 1993.
- [16] J. Kececioğlu and D. Sankoff. Efficient bounds for oriented chromosome inversion distance. In *Proc. of 5th Ann. Symp. on Combinatorial Pattern Matching*, pages 307–325. Springer-Verlag LNCS 807, 1994.
- [17] E. B. Knox, S. R. Downie, and J. D. Palmer. Chloroplast genome rearrangements and evolution of giant lobelias from herbaceous ancestors. *Mol. Biol. Evol.*, 10:414–430, 1993.
- [18] D. E. Knuth. *The Art of Computer Programming, Vol. III: Sorting and Searching*. Addison-Wesley, Reading, MA, 1973.
- [19] J. H. Nadeau and B. A. Taylor. Lengths of chromosomal segments conserved since divergence of man and mouse. *Proc. Natl. Acad. Sci. USA*, 81:814–818, 1984.
- [20] A. H. Sturtevant and T. Dobzhansky. Inversions in the third chromosome of wild races of *drosophila pseudoobscura*, and their use in the study of the history of the species. *Proc. Nat. Acad. Sci.*, 22:448–450, 1936.