JOURNAL OF COMPUTATIONAL BIOLOGY
Volume 5, Number 3, 1998
Mary Ann Liebert, Inc.
Pp. 555–570

Multiple Genome Rearrangement and Breakpoint Phylogeny

DAVID SANKOFF[1] and MATHIEU BLANCHETTE[2]

ABSTRACT

Multiple alignment of macromolecular sequences generalizes from $N = 2$ to $N \geq 3$ the comparison of N sequences which have diverged through the local processes of insertion, deletion and substitution. Gene-order sequences diverge through non-local genome rearrangement processes such as inversion (or reversal) and transposition. In this paper we show which formulations of multiple alignment have counterparts in multiple rearrangement. Based on difficulties inherent in rearrangement edit-distance calculation and interpretation, we argue for the simpler "breakpoint analysis." Consensus-based multiple rearrangement of $N \geq 3$ orders can be solved exactly through reduction to instances of the Travelling Salesman Problem (TSP). We propose a branch-and-bound solution to TSP particularly suited to these instances. Simulations show how non-uniqueness of the solution is attenuated with increasing numbers of data genomes. Tree-based multiple alignment can be achieved to a great degree of accuracy by decomposing the tree into a number of overlapping 3-stars centered on the non-terminal nodes, and solving the consensus-based problem iteratively for these nodes until convergence. Accuracy improves with very careful initializations at the non-terminal nodes. The degree of non-uniqueness of solutions depends on the position of the node in the tree in terms of path length to the terminal vertices.

1. INTRODUCTION

MULTIPLE ALIGNMENT OF MACROMOLECULAR SEQUENCES, an important topic of algorithmic research for at least 25 years (Sankoff *et al.*, 1973; Sankoff, 1997), generalizes from $N = 2$ to $N \geq 3$ the comparison of N sequences which have diverged through the local processes of insertion, deletion and substitution. Recently there has been much interest in gene-order sequences which diverge through non-local genome rearrangement processes such as inversion (or reversal) and transposition (cf. Pevzner and Waterman, 1995; Waterman, 1995, ch.15.4.2; Setubal and Meidanis, 1997, ch.7; Gusfield, 1997, ch. 19). What would be the analog of multiple alignment under these models of divergence? In this introduction we first review some formulations of multiple alignment and show which have counterparts in multiple rearrangement. We then discuss the difficulties inherent in edit-distance formulations of multiple rearrangement, referring to relevant work, and argue for a potentially simpler approach based on "breakpoint analysis."

[1]Centre de recherches mathématiques, [2]Laboratoire de biologie informatique et théorique, Université de Montréal, Montréal, Québec H3C 3J7, Canada.

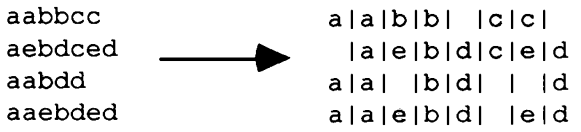

FIG. 1. Multiple sequence alignment using gaps.

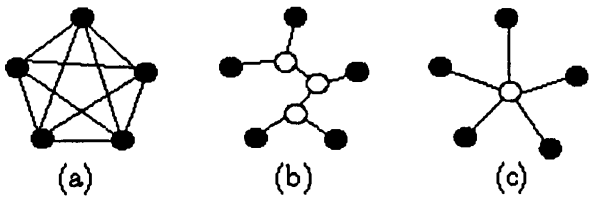

FIG. 2. Three ways of defining column costs in terms of pairwise comparisons.

1.1. Multiple sequence alignment

The goal of multiple sequence alignment is to align the terms of N sequences into a number of relatively homogeneous columns through the judicious insertion of one or more null terms, or gaps, between consecutive terms in some or all of the sequences, as in Figure 1, so as to optimize an objective cost function.

In the simplest case, this objective is just the sum of column costs across all columns of the alignment. Each column cost measures how different the terms in that column are among themselves. For example, in a "complete" comparison the column cost is the number of pairs of sequences which differ in that column, as represented in (a) in Figure 2, in which every vertex (sequence) is compared to every other.

Another definition of column cost depends on a given phylogenetic tree, as in (b) in the figure, in which the leaves are the data sequences and the internal nodes (open dots) are hypothetical ancestral sequences reconstructed by some method from the contemporary sequences. Both data and reconstructed sequences must be aligned and the column cost is just the number of tree branches where the sequences at the two ends have different elements in the column. A special case of the tree-based comparison is the "consensus" comparison, represented in (c), where there is just one reconstructed sequence.

There are many other formulations of the problem which we will not discuss, e.g., the column cost may be $N - M$, where M is the number of occurrences of the most frequent term in a column.

1.2. The analogy with genome rearrangement

A key difference between sequence comparison and gene-order comparison is that in the former, algorithms try to identify corresponding terms in the two sequences being compared and the number of divergence steps then falls out directly, whereas in the latter the correspondence (i.e., alignment) is given and it is the number of steps which must be calculated. (Note that combined alignment and rearrangement problems can be formulated for macromolecular sequences. Cf. Schöniger and Waterman, 1992; Varré *et al.*, 1997. This potentially difficult problem is outside the scope of this paper.)

Thus version (a) of the multiple alignment problem has no analog in gene-order rearrangement, since there is nothing to optimize once the pairwise distances are given. On the other hand, in versions (b) and (c) of the problem, there is something to optimize, namely the ancestral gene orders represented by the open dots. This is the focus of this article.

1.3. Some background on rearrangement distances

The algorithmic study of comparative genomics has focused on inferring the most economical explanation for observed differences in gene orders in two or more genomes in terms of a limited number of rearrangement processes. For single-chromosome genomes, this has been formulated as the problem of calculating an edit distance between two linear orders on the same set of objects, representing the ordering of homologous genes in two genomes. In the most realistic version of the problem, a sign (plus or minus) is associated with each object in the linear order, representing the direction, or orientation, of transcription (sometimes referred to as strandedness) of the corresponding gene. The elementary edit operations may include one or more of:

inversion, or reversal, of any number of consecutive terms in the ordered set, which, in the case of signed orders, also reverses the polarity of each term within the scope of the inversion. Kececioglu and Sankoff (1993, 1995) considered the problem of computing the minimum reversal distance between two given

permutations in the unsigned case, including approximation algorithms and an exact algorithm feasible for moderately long permutations. Bafna and Pevzner (1996) gave improved approximation algorithms for this problem. Caprara (1997a) showed this problem to be NP-complete. Kececioglu and Sankoff (1994) also found tight lower and upper bounds for the signed case and implemented an exact algorithm which worked rapidly for long permutations. Indeed, Hannenhalli and Pevzner (1995a) showed that the signed problem is only of polynomial complexity, and an improved polynomial algorithm was given by Kaplan *et al.* (1997).

transposition of any number of consecutive terms from their position in the order to a new position between any other pair of consecutive terms. This may or may not also involve an inversion. Computation of the transposition distance between two permutations was considered by Bafna and Pevzner (1995). Sankoff *et al.* (1992), Sankoff (1992), Blanchette *et al.* (1996), and Gu *et al.* (1997) implemented and applied heuristics to compute edit distances which combine inversions, transpositions and deletions, in some cases allowing differential weighting of these operations.

In addition, for multi-chromosome genomes, a major role is played by:

reciprocal translocation. Kececioglu and Ravi (1995) began the investigation of translocation distances, and Hannenhalli and Pevzner (1995b) have shown that one formulation of the problem is of polynomial complexity. A relaxed form of translocation distance was proposed by Ferretti *et al.* (1996) and the complexity of its calculation was shown to be NP-complete by DasGupta *et al.* (1997).

1.4. Extensions to $N \geq 3$

There have been a number of investigations of phylogeny based on the algorithmic comparison of gene order within a number of genomes, using *pairwise* comparisons followed by *distance matrix* methods (e.g., Sankoff *et al.*, 1992). However, treeing methods which involve the optimal reconstruction of gene order at ancestral nodes (Hannenhalli *et al.*, 1995; Sankoff *et al.*, 1996) have been little used because of the computational difficulty in generalizing measures of genomic distance to more than two genomes. Caprara (1997b) has shown that the most promising case, reversal distance for only three signed permutations, is NP-hard. We argue that

this computational difficulty—there are no algorithms guaranteeing exact solutions for even three relatively short genomes—together with

unwarranted assumptions as to the relative importance of different rearrangement events implicit in genome distances such as minimum reversal distance, minimum transposition distance, minimum translocation distance and even distances combining these (cf. Blanchette *et al.*, 1996), as well as

the fallacy that calculation of an edit distance allows the recoverability of the "true" history of genomic divergence—in fact, the severe non-uniqueness of the optimal edit path for moderate or large gene-order distances has much worse (i.e., non-local) consequences than with the classical multiple alignment problem,

the bias demonstrated in simulations, where calculation of genome distance severely underestimates the actual number of events generating moderate or large gene-order differences, and

the sensitivity of these measures to errors and other small changes in the data. These are especially numerous when gene order has been determined by mapping techniques other than complete sequencing (Sankoff *et al.*, 1997)

all militate in favour of extending gene-order comparisons to three or more genomes through a much simpler and model-free metric. In this paper we suggest the number of breakpoints as just such a metric.

1.5. Presentation of the research

In Section 2 we define the breakpoint metric for unoriented and oriented genomes, and set up the analogy to multiple alignment modes (b) and (c) in Figure 2. In Section 3 we review how (c), consensus-based multiple rearrangement, can be solved exactly through reduction to a not unwieldy version of the Travelling Salesman Problem. In Section 4, we use this exact solution applied to simulated data to show how the problem of

non-uniqueness is attenuated with increasing numbers of data genomes. In Section 5, we report how (b), tree-based multiple alignment, can be achieved to a great degree of accuracy by decomposing the tree into a number of overlapping 3-stars centered on the non-terminal nodes, and "steinerizing"—solving the consensus-based problem iteratively for these nodes until convergence. The accuracy depends on very careful initializations at the non-terminal nodes, and can be assessed through simulation. In Section 6 we again investigate non-uniqueness, this time focusing on the effect of the position of the node in the tree in terms of path length to the terminal vertices.

2. BREAKPOINT ANALYSIS

Consider two genomes $A = a_1 \cdots a_n$ and $B = b_1 \cdots b_n$ on the same set of genes $\{g_1, \ldots, g_n\}$. We say a_i and a_{i+1} are adjacent in A. We also consider that a_1 is adjacent to the genome "start" and a_n is adjacent to the "end." For circular genomes, it suffices to consider that a_n and a_1 are adjacent. If two genes g and h are adjacent in A but not in B, they determine a breakpoint in A. We define $\Phi(A, B)$ to be the number of breakpoints in A. This is clearly equal to the number of breakpoints in B.

For two genomes whose gene sets are not identical, to calculate the breakpoints, we first remove all genes that are present in only one of the genomes. We then find the breakpoints for the reduced genomes, now of identical composition. The positions of the breakpoints are well-defined in the reduced genomes. In the full genomes, there is a breakpoint between a_i and a_{i+1} only if this is a breakpoint for the reduced genome. If, as in Figure 3, there is a breakpoint between a_i and a_j in the reduced genome, where $j \neq i + 1$, then there is a corresponding breakpoint in the full genome, but its position is ambiguous. We call it a *hidden* breakpoint; it is somewhere between a_i and a_j, which are not adjacent.

The number of breakpoints between two genomes is not only the most general measure of genomic distance, requiring no assumptions about the mechanisms of genomic evolution (inversion versus transposition versus translocation) underlying the data, but it is also the easiest to calculate. In addition, it has proven relations to the edit distances; e.g., half the number of breakpoints is a lower bound on the reversal distance.

2.1. *Oriented genomes*

Our simulations will involve directed, or oriented, genomes; we assume we know the strandedness, or direction of transcription, of each gene in each genome in the data set. In this case, the notion of breakpoint must be modified to take into account the polarity of the two genes. If gh represents the order of two genes in one genome, then if another genome contains gh or $-h\ -\ g$ there is no breakpoint involved. However, between gh and hg there is a breakpoint, similarly between gh and $-g\ -\ h$, $g\ -\ h$, $-gh$, $h\ -\ g$ or $-hg$. Adjacency is no longer commutative.

2.2. *Tree-based multiple genome rearrangement*

The problem is formulated as follows: Let T = (V, E) be an unrooted tree with $N \geq 3$ leaves and $\Sigma = \{g_1, \ldots, g_n\}$ be a set of genes. Suppose $\{V_1, \ldots, V_N\} \subset V(T)$ are the leaves of the tree and $\{V_{N+1}, \ldots, V_L\}$, where $N < L \leq 2N - 2$, are the internal vertices of the tree. The data consist, for each leaf V_i, $i = 1, \ldots, N$, of a circular permutation $G^i = g^i_1 \cdots g^i_n$ of the genes in Σ, representing the genome of a contemporary

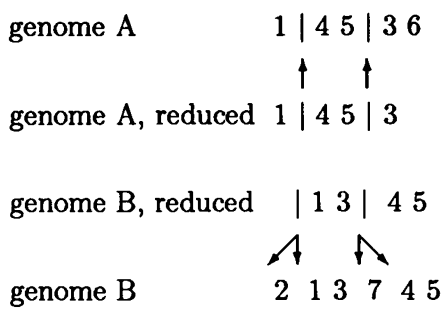

FIG. 3. Defining breakpoints for (circular) genomes with different gene contents. Position of breakpoints (vertical strokes) found first in reduced genomes with identical gene sets. This unambiguously determines breakpoints between 1 and 4 and between 5 and 3 in genome A. Breakpoint between 5 and 1 in genome B is "hidden" by gene 2; that between 3 and 4 is hidden by gene 7.

species. The task is to find the permutations $G^{N+1}, \ldots, G^L$ associated with the internal (ancestral) vertices $V_{N+1}, \ldots, V_L$, such that

$$\sum_{V_i V_j \in E(T)} \Phi(G^i, G^j)$$

is minimized.

2.3. *Binary tree- versus consensus-based multiple genome rearrangement*

We will concentrate on two extreme cases: that of completely resolved, or binary, trees, where $L = 2N - 2$ and all non-terminal nodes are of degree 3; and that of completely unresolved trees, or "stars," where $L = N+1$ and the single non-terminal node has degree N.

3. CONSENSUS-BASED REARRANGEMENT

Though all the breakpoint-based multiple rearrangement problems in Section 2 seem NP-hard, as reported for $N = 3$ by Pe'er and Shamir (1998), for moderate n they are tractable, since they may be reduced to a number of interconnected instances of the Traveling Salesman Problem (TSP). In the case of consensus-based rearrangement, the solution, involving just one TSP, is globally optimal.

We define Γ to be the complete graph whose vertices are the elements of Σ. For each edge gh in $E(\Gamma)$, let $u(gh)$ be the number of times g and h are adjacent in the N data genomes. Set $w(gh) = N - u(gh)$. Then the solution to TSP on (Γ, w) traces out an optimal genome S on Σ, since if g and h are adjacent in S, but not in V_1, for example, then they form a breakpoint in S.

For oriented genomes, the reduction of the median problem to TSP must be somewhat different to take into account that the median genome contains g or $-g$ but not both. Let Γ be a complete graph with vertices $V(\Gamma) = \{-g_n, \ldots, -g_1, g_1, \ldots, g_n\}$. For each edge gh in $E(\Gamma)$, let $u(gh)$ be the number of times $-g$ and h are adjacent in the N data genomes and $w(gh) = N - u(gh)$, if $g \neq -h$. If $g = -h$, we simply set $w(gh) = -Z$, where Z is large enough to assure that a minimum weight cycle must contain the edge $-gg$.

Proposition. If $s = s_1, -s_1, s_2, -s_2, \ldots, s_n, -s_n$ is the solution of the TSP on (Γ, w), then the median is given by $S = s_1 s_2 \cdots s_n$.

Proof.

$$\sum_{i=1}^{N} \Phi(S, V_i) = \sum_{gh \in S, g \neq -h} w(gh)$$

$$= nZ + \sum_{gh \in s} w(gh).$$

Thus S minimizes $\sum_{i=1}^{N} \Phi(S, V_i)$ iff s is of minimal weight. ■

3.1. *A lower bound*

To solve this restricted form of TSP, we resort to a branch-and-bound algorithm based on the following lower bound:

Let the *edge-pool* $P \subseteq E(\Gamma)$, be disjoint from the *fragment* $F \subseteq E(\Gamma,)$ and let $score = \sum_{gh \in F} w(gh)$. Define $a(g)$, the *availability* of $g \in V(\Gamma)$, to be 2, 1 or 0, depending on whether g is incident to zero, one, or more than one edge in F, respectively, Let $\mu(g)$ be the sum of the $a(g)$ smallest weights of edges in P incident to g. ($\mu(g)$ is undefined if there are less than $a(g)$ such edges.)

If there is a TSP solution cycle S of weight W_S which includes all the edges in the fragment F and some additional edges drawn from the edge-pool P, let $\nu(g)$ be the sum of the weights of the exactly $a(g)$ edges of S in P incident to g. (In this case $\mu(g)$ is always defined.) Clearly $\mu(g) \leq \nu(g)$.

Now,

$$W_S = score + \sum_{gh \in E(S) \cap P} w(gh)$$

$$= score + \frac{1}{2} \sum_{g | gh \in E(S) \cap P} w(gh)$$

since each edge in $E(S) \cap P$ is counted twice in the sum. Thus

$$W_S = score + \frac{1}{2} \sum_{g|gh \in E(S) \cap P} \nu(g).$$

Defining

$$L(P) = \frac{1}{2} \sum_{g|gh \in E(S) \cap P} \mu(g),$$

$$score + L(P) \leq score + \frac{1}{2} \sum_{g|gh \in E(S) \cap P} \nu(g) = W_S.$$

We use $L(P)$ as a lower bound in a branch-and-bound TSP algorithm. When $P = E(\Gamma)$ and $F = \Phi$ this as a well-known bound on TSP (see, e.g., Minieka, 1978, pp. 272–273). There are a number of other bounds which can be used for the TSP, but this one is of particular interest in that it is a relatively tight bound for the type of TSP originating in breakpoint problems, especialy the case $N = 3$, and it can be modified for use in the case the input genomes have different sets of genes (Sankoff and Blanchette, 1997).

Note that in this algorithm the recursion functions as a "greedy" search until it first finds a cycle, which is necessarily an upper bound. If its cost $U = L(E(\Gamma))$, it is optimal.

TSP algorithm

input: weighted complete graph (Γ, w)
output: solution S to the TSP on (Γ, w)

initialization
  $V(S) \leftarrow V(\Gamma)$
  $F \leftarrow \Phi$
  $P \leftarrow E(\Gamma)$
  score$\leftarrow 0$
  best$\leftarrow \infty$

routine BBTSP (P, F, S, score, best)
  if $|F| = |\Gamma|$ **and** score < best **then**
    store $S = F$ as current best solution
    best$\leftarrow$score
  endif

  if $|F| < |\Gamma|$ **then**
    if $L(P)+$ score < best **then**
      choose $gh \in P$ to try to add to F
      where $a(g) > 0$, $a(h) > 0$ and $w(gh)$ is as small as possible,
      and $F \cup \{gh\}$ is not a cycle on less than $|\Gamma|$ vertices.
      BBTSP ($P - \{gh\}$, $F \cup \{gh\}$, S, score $+ w(gh)$, best)
        BBTSP ($P - \{gh\}$, F, S, score, best)
      endif
    endif
  end

3.2. Differing gene sets

In general, the genomes available for this type of analysis will not all have exactly the same set of genes. In this case, it is not always clear what the ancestral gene sets should contain; and when there is a rule for deriving these sets, the solution to the consensus problem, even in the case $N = 3$, is much more difficult.

We have generalized the lower bound calculation and the BBTSP routine to give the optimal consensus order of N genomes with difering gene sets (Sankoff and Blanchette, 1997). The computing time, however, is much greater, and so in the ensuing sections we will confine our investigation to the case of a common gene set.

4. THE UNIQUENESS OF THE CONSENSUS

The reduction to TSP allows us to obtain global solutions for moderate-sized problems in reasonable time—typically 1 second for reconstructing the consensus of three or more scrambled genomes with $n = 20$ genes, and well less than a minute for $n = 50$, on an Origin 200 computer with a RISC 10000 processor. This enables us to undertake systematic simulation studies. To assess the uniqueness of the solutions to the problem as a function of the number of genomes simultaneously rearranged, we constructed N genomes, each by applying a number ρ of random reversals to a common ancestor $(1, 2, \ldots, 20)$, and then solved the consensus problem. (Recall that each reversal reverses the order of a number of consecutive terms and also changes the sign of each of these terms. It adds at most two new breakpoints. We used reversals not because we privilege them as a model of biological evolution but simply as a convenient way of scrambling permutations.)

We searched for other multiple solutions by permuting the labels of the 20 genes in the input. We repeated each example with 10 different gene labelings. We compared all 10 solutions obtained by averaging their pairwise distances (i.e., the number of breakpoints between the two solution genomes). For each value of N between 2 and 16, and each value of ρ between 1 and 12, we repeated the experiment for 10 different examples and averaged their results, running the program 100 times—10 examples times 10 gene labelings per example.

Figure 4 shows the results of these experiments. We note first that the curves for large ρ are systematically "worse" than those for low ρ; the more scrambled are the data genomes, the less likely they are to have a unique consensus.

More interesting perhaps, is the rapid, almost linear, decrease in non-unicity for each ρ as the number of data genomes increases. For 15 branches or more, there seems to be a unique consensus, no matter how large ρ is. And in all cases examined, this consensus order was none other than $(1, 2, \ldots, 20)$. Of course, for larger genomes, we could expect a larger cut-off point.

5. BINARY TREE-BASED REARRANGEMENT

A general method for the inference of ancestral genomes on a fixed binary tree such as in Figure 1(c) is the iterative improvement method of Sankoff *et al.* (1976), as adapted for the genomics context in Sankoff *et al.*

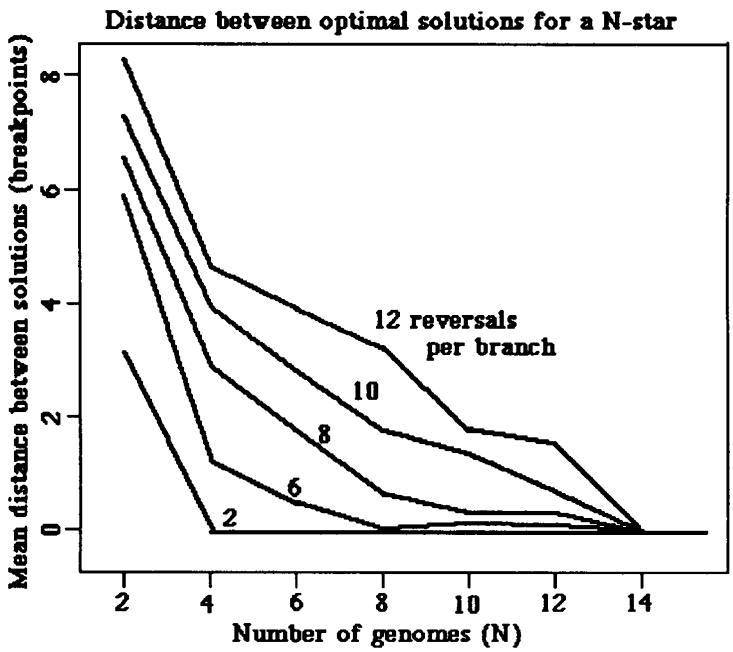

FIG. 4. Diversity of solutions to consensus problem as a function of N.

(1996), Ferretti *et al.* (1996). Each of the $N-2$ internal vertices, together with its three neighbors, defines a 3-star. The solution to the tree-based multiple rearrangement problem will have a reconstructed genome associated with each such vertex, which must be a solution to the consensus-based problem determined by these neighbors.

The strategy is to start with an initial tree where some genome is assigned to each internal genome, then to improve (steinerize) one of these ancestral genomes at a time by solving the consensus problem for the 3-star consisting of its immediate neighbours (the "median problem"), iterating across the tree until convergence. Of course, there is no guarantee that convergence will occur at a global optimum.

Without loss of generality, we may assume that the internal vertices are numbered in such a way that of the three neighbors of each vertex, two either precede it in the list or are leaves. This assures that if genomes for the internal vertices are inferred one by one according to this numbering, the set of untreated vertices, as it shrinks, at all times forms a connected tree.

Then the algorithm **optimize_tree**, in which we leave unspecified how to set up the initial TSP for each genome to be reconstructed, converges to a (local) optimum:

algorithm optimize_tree

input $G^1, \ldots, G^N$
cost $\leftarrow \infty$
extremities $\leftarrow \{1, \ldots, N\}$
internal $\leftarrow \{N+1, \ldots, 2N-2\}$
do for $M = N+1, \ldots, 2N-2$,
  set_up_TSP for G^M
  solve TSP for G^M
  remove the two neighbors of V_M preceding it in
    the vertex numbering from *extremities*
  transfer V_M from *internal* to *extremities*
enddo
routine iterate_median
output $G^{N+1}, \ldots, G^{2N-2}$

In each of Sections 5.2, 5.3 and 5.4 below, the **set_up_TSP** instruction will be replaced by a specific routine. The **iterate_median** routine is independent of the set-up strategy in the initialization; in fact all three approaches to be used are identical for 3-leaf trees (i.e., the median problem).

iterate_median

routine iterate_median
  while $C = \sum_{V_i V_j \in E(T)} \Phi(G^i, G^j) < cost$,
    $cost \leftarrow C$
    do for $M = N+1, \ldots, 2N-2$,
      $G^* \leftarrow$ **median**(G^h, G^j, G^k), where V_h, V_j and V_k
          are the three neighbors of V_M
      if $\sum_{\{h,j,k\}} \Phi(G^*, G^x) < \sum_{\{h,j,k\}} \Phi(G^M, G^x)$
        $G^M \leftarrow G^*$
      endif
    enddo
  endwhile
end

5.1. Initialization strategies

The output of this algorithm is not necessarily a global optimum. The main factor in directing convergence towards a global optimum is the how the initialization is carried out. We can identify at least six distinct approaches to initialization, which can be grouped into three levels of increasing likelihood that they fall into the domain of attraction of a global optimum. Thus each internal genome can be assigned:

"arbitrarily":

(1) a fixed, arbitrary permutation, e.g. $(1, 2, \ldots, n)$, or
(2) a different random permutation

"reasonably":

(3) the permutation representing a nearest data genome, or
(4) the consensus of three nearest data genomes

"with much effort":

(5) by setting up and solving an initial TSP at each internal node, where the edge-weights are the average of the corresponding edge-weights at the three neighbouring nodes, found by solving a system of linear equations, or
(6) by setting up and solving an initial TSP at each internal node, where the edge-weights are calculated by dynamic programming, minimizing the number of times a given adjacency has to be created or disrupted within the tree to be present or absent, respectively, at that node.

It can be seen in **optimize_tree** that rather than initializing all internal nodes at once, they are initialized one at a time, starting with an internal node with two terminal node neighbours. Once it is initialized, it is treated as a terminal node (i.e., in *extremities*), and the two neighbours are disregarded, as the initialization proceeds with another internal vertex.

In the following subsections we formalize initializations (4), (5) and (6).

5.2. *Triangulation*

Then we can replace the **set_up_TSP** instruction in **optimize_tree** by the following:

three_nearest

routine three_nearest
  let V_h, V_j, V_k be the three vertices in *extremities*
    closest to V_M on three disjoint paths leading from V_M
  define TSP for G^M, based on V_h, V_j, V_k.
end

5.3. *Trees of TSPs*

Instead of setting up the TSP at each internal vertex as a function of the three closest previously solved genomes, we can define a TSP on the basis of the three immediately neighboring TSPs. For each vertex $V_M \in$ *extremities*, we set

$$w_M(gh) = \begin{cases} 1 & \text{if } gh \text{ is not in } G^M \\ 0 & \text{if } gh \text{ is in } G^M. \end{cases}$$

We then determine the weights for the vertices in *internal* as follows:

$$w_M(gh) = \frac{1}{3}(w_h(gh) + w_j(gh) + w_k(gh)),$$

for each $gh \in \Gamma$, where V_h, V_j and V_k are the three neighbors of V_M. The weight system **w** can then all be easily found by solving the system of simultaneous equations derived from all the vertices $\in$ *internal*.

We can replace the **set_up_TSP** instruction in **optimize_tree** by the following:

average_TSP

routine average_TSP
  calculate **w** for the vertices in the *internal*
    based on the vertices in *extremities*
end

5.4. *Minimizing Adjacency Disruptions*

Our third heuristic focuses first on each pair of genes in Σ and tries to minimize the number of times this pair is inferred to have been directly affected by rearrangement of the genome. Dynamic programming is used to calculate the weights for the TSP.

For any internal vertex V_M, suppose we have already calculated a genome for vertices $V_{N+1}, \ldots, V_{M-1}$ and we wish to do so for V_M. We impose a direction on all edges of the tree, namely the direction leading to V_M. Then V_M has three edges leading to it, all other internal vertices have two, and leaves have none. The dynamic programming routine included in the set-up routine below follows this direction towards V_M.

adjacency_parsimony

routine adjacency_parsimony
  direct all edges in $E(T)$ towards M
  do for $i \in$ *extremities* and all $gh \in \Gamma$
    $w_i^+(gh) \leftarrow 0$ if $ij \in G^i$, $w_i^+(gh) = 1$ if $ij \notin G^i$.
    $w_i^-(gh) \leftarrow 1$ if $ij \in G^i$, $w_i^-(gh) = 0$ if $ij \notin G^i$.
  enddo
  remain $\leftarrow$ *internal*
  while *remain* $\neq \Phi$
    find $i \geq M$, $i \in$ *remain*, such that for all vertices j leading to i,
      $j \notin$ *remain*
    do for all $gh \in \Gamma$
      $w_i^+(gh) \leftarrow \Sigma_{V_j \text{ leads to } V_i} \min(w_j^+(gh), 1 + w_j^-(gh))$
      $w_i^+(gh) \leftarrow \Sigma_{V_j \text{ leads to } V_i} \min(w_j^-(gh), 1 + w_j^+(gh))$
    enddo
    remove i from *remain*
  endwhile
  do for all $gh \in \Gamma$
    $w_M(gh) \leftarrow w_M^+(gh) - w_M^-(gh)$
  enddo
end

5.5. *The Simulations*

To assess and compare the three approaches to initializing the iteration of the median algorithm, a series of simulations were carried out. The parameters were N, the number of terminal vertices in the tree, n, the number of genes in each genomes, and r, the total number of breakpoints between all pairs of adjacent genomes in the tree. Here, we illustrate with the results for $N = 7$ and $n = 20$. The total number of rearrangements r was varied from 20 to 300 in steps of 10.

For each target value of r, ten sets of simulated genomes were required. Starting with genome $(1\ 2 \cdots n)$ at one vertex, we generated genomes for neighbouring vertices with an appropriate random number of rearrangements until all internal and terminal vertices were assigned a genome. Each rearrangement was randomly chosen to be a transposition or an inversion (cf. Blanchette *et al.*, 1996) of random length.

Once all genomes were generated, the breakpoints on each edge were counted, and the simulated example was retained only if r was the target values, until the quota of 10 examples was filled.

For each example, only the genomes from the terminal vertices served as input for each of our three algorithms separately.

5.6. *Results*

It can be seen from Figure 5, that when the average number of breakpoints per edge approaches $\frac{1}{2}n$, the algorithm tends to reconstruct evolutionary histories more parsimonious than those actually responsible for the data. After $\frac{2}{3}n$, the number of reconstructed breakpoints actually levels off sharply. Note that in this and subsequent figures in this section, all curves are smoothed by the SPLUS **lowess** function.

The accuracy of our initializations can be assessed in Figure 6, which gives the improvement to the objective R obtained by the iteration step as a function of r for the three heuristics. This improvement is generally less

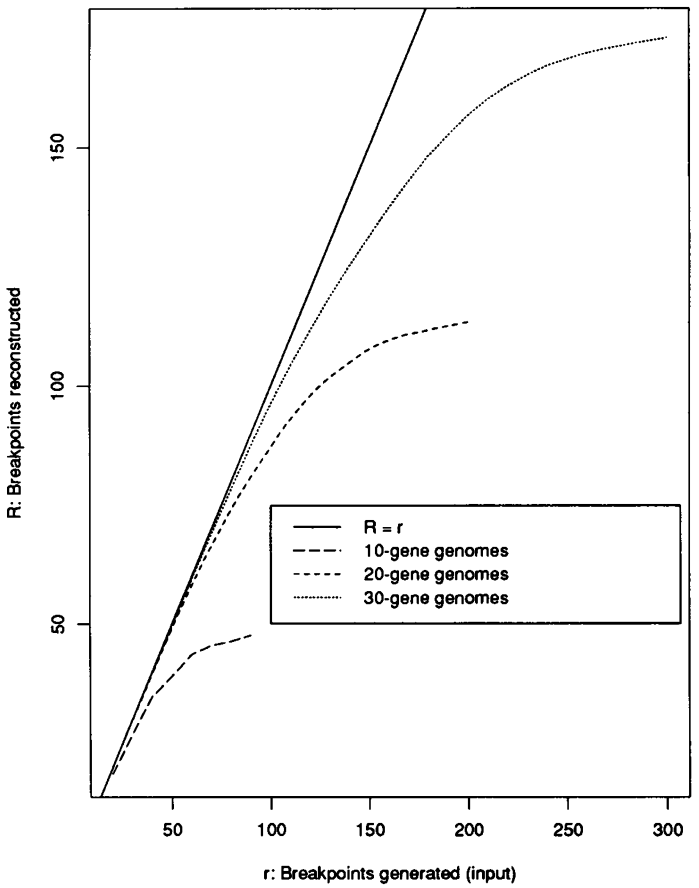

FIG. 5. Number of reconstructed breakpoints R (best of three heuristics) as a function of number of breakpoints generated in the input data, for 10-gene, 20-gene and 30-gene genomes. Number of leaves $N = 7$; number of branches $2N - 3 = 11$.

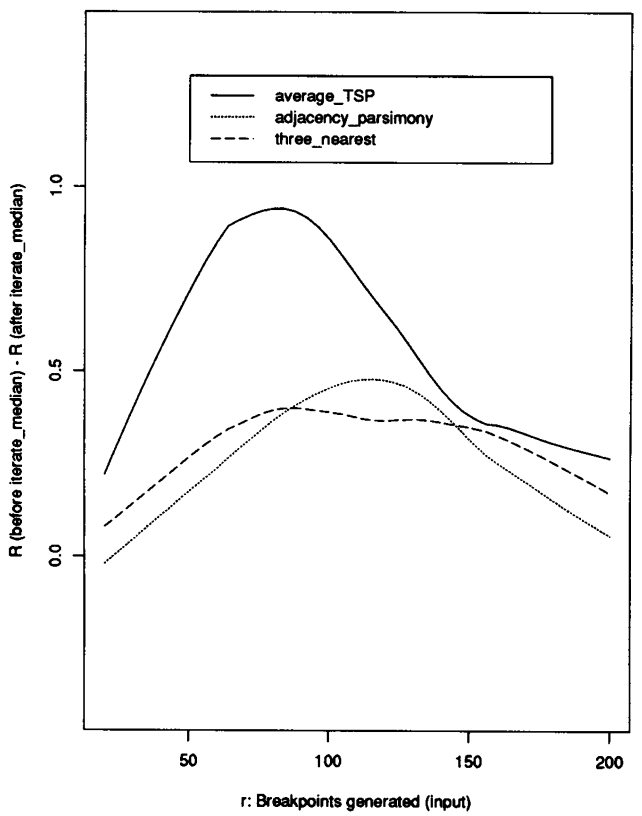

FIG. 6. Decrease in number of reconstructed breakpoints R, for each heuristic, following iteration step, as a function of number of breakpoints generated in the input data. $n = 20$, $N = 7$.

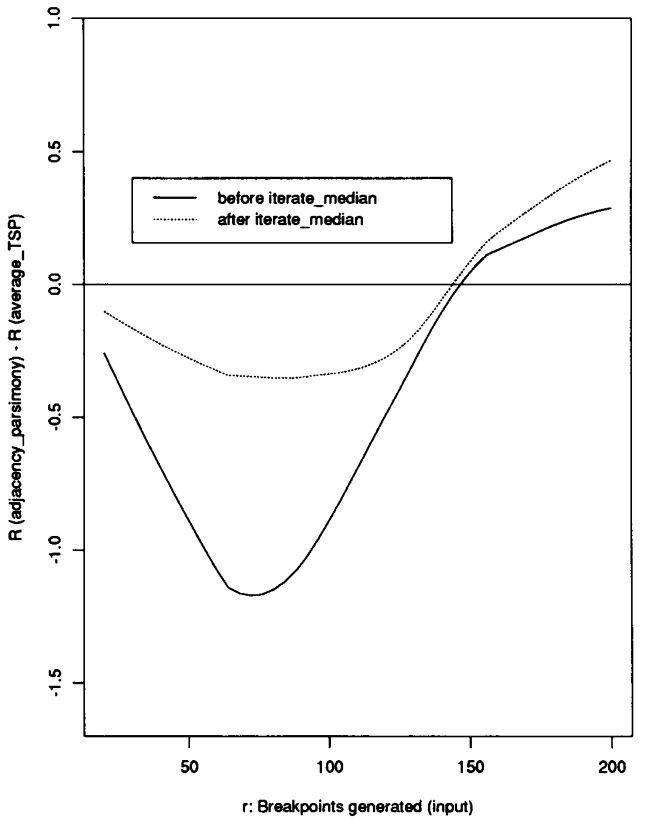

FIG. 7. Difference between results of **adjacency_parsimony** and **average_TSP** as a function of r, before and after iterative improvements. $n = 20$, $N = 7$.

than $\frac{1}{2}\%$, reaching more than 1% for the **average_TSP** initialization only for values of r where, as we shall see, this routine performs relatively poorly.

Figure 7 compares the performance of the two heuristics **average_TSP** and **adjacency_parsimony** (both outperform **three_nearest**) over a range of evolutionary divergence. It is striking that for small r, **adjacency_parsimony** performs distinctly better, even after both initializations benefit from the iterative improvements, while for large r it is the **average_TSP** which is clearly superior.

To address the question of global optimality, we count how many heuristics give the minimum solution for R. In Figure 8, we see that (except for genomes that have diverged very little) around 1.6 heuristics, on the average, seem to obtain the minimum. Assuming a doubly-attained minimum is a global solution (not always true, of course), and since **adjacency_parsimony** and **average_TSP** are the ones that tend to achieve the lowest values, we can conjecture that individually they attain global optimality about half of the time, for this range of parameter values.

Summarizing, "much effort" paid off with one to two percent better results (i.e., fewer breakpoints over the entire tree) for moderate to highly divergent data. The dynamic programming approach (6) leads to better results than method (5) for moderately divergent data while the latter is superior for highly divergent data, approaching randomness, in each case by about one half of one percent.

Once the number of breakpoints generated per tree branch reached about half the number of genes, underestimation begins to be manifested, rapidly worsening so that when the number of breakpoints per branch reaches two-thirds the number of genes, this number is underestimated by about 30%.

6. UNIQUENESS IN TREE-BASED REARRANGEMENT

To further our investigation of multiple optima in reconstructed genomes, we simulated genomic rearrangement in the tree in Figure 9.

We again used genomes of size 20 and generated trees containing a total of B breakpoints over all their edges. We applied methods (4), (5) and (6) of Section 5 to calculate the multiple rearrangement for each

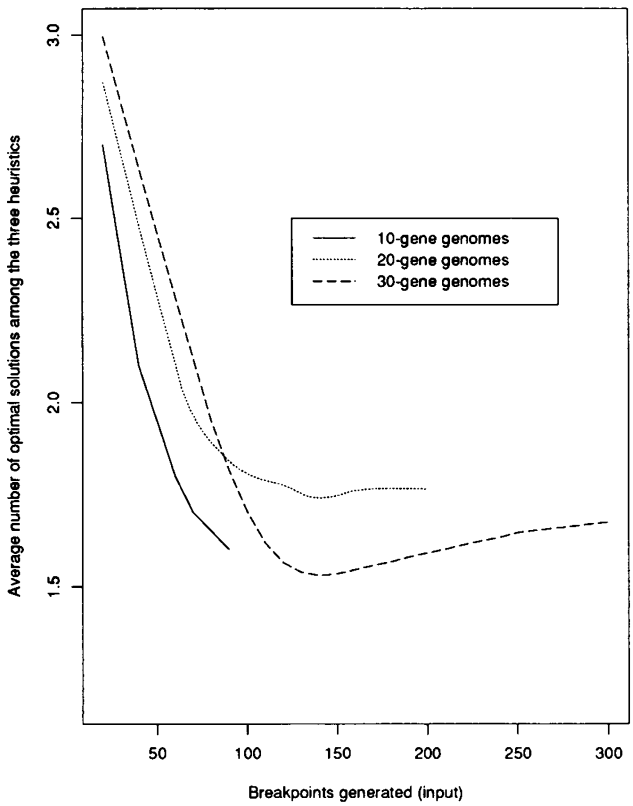

FIG. 8. Number of heuristics (out of three) attaining optimal solution as a function of number of breakpoints generated in the input data, for 10-gene, 20-gene and 30-gene genomes. $N = 7$.

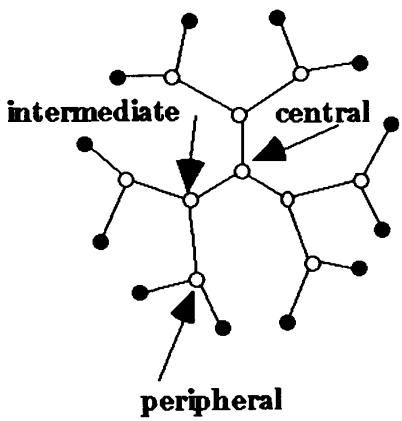

FIG. 9. Tree with three depths of internal vertices.

tree. When at least two heuristics found the same minimum total cost (which happened at least 70% of the time, even for highly divergent data), we calculated the distances between the genomes reconstructed by each method at each internal node.

This experiment was repeated 10 times (and the results averaged) for each value of B between 40 and 230. The results appear in Figure 10.

The results of these simulations indicate that multiple solutions for peripheral nodes are relatively close to each other for low and moderate divergence. But with 10 breakpoints per branch, on an average, somewhat more than 200 breakpoints in all, multiple solutions could be non-negligibly far from each other—about 7 breakpoints between them on the average. The situation is progressively worse as we get deeper into the tree, so that there are considerably more breakpoints (around 15) between two solutions for the central genome than between two neighboring nodes on the tree.

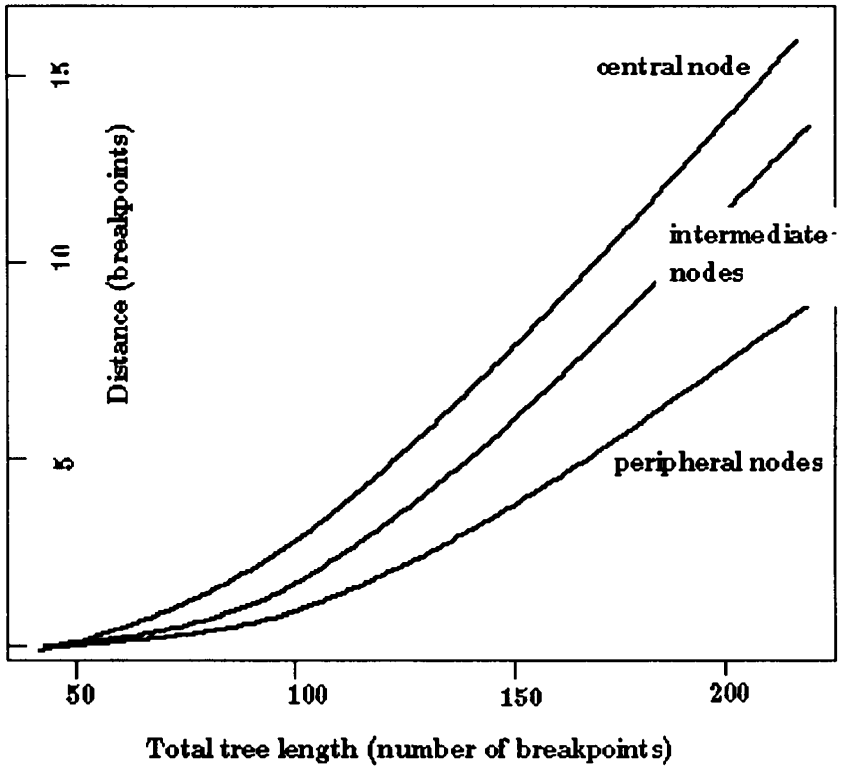

FIG. 10. Effect of vertex depth on dispersion of multiple optima.

7. CONCLUSIONS

This work establishes the computational feasibility of exact breakpoint analysis as a method of multiple genome rearrangement, in contrast to the difficulties with edit distance-based approaches.

For the consensus problem on N genomes, we analyzed the diversity of optimal solutions in terms of the breakpoint distances amongst them, as an assessment of the reliability of reconstructed gene orders. Reliability was degraded as input genomes were increasingly rearranged, but improved dramatically with N.

For the more general breakpoint phylogeny, an exact algorithm is not available, but we tested three initializations for solving it by iterative improvement. We showed that the initializations were very precise, within one percent or so of the best solution. The obverse of this is that the iterative step leads to a small, but non-negligible, improvement.

One initialization worked better for low-divergence data and one is superior for high-divergence data. Studying the rate of coincidental solutions among the three heuristics enabled us to estimate how frequently the methods are likely to achieve global optima.

Though breakpoint distance does not directly measure the number of rearrangement events, we can still characterize how parsimony leads to underestimation of the evolutionary divergence in the phylogeny through which the data were generated.

In analyzing the breakpoint distances among equivalent local optima, we must conclude that non-uniqueness remains a major consideration in genomic reconstruction, especially for the "deep" vertices in a phylogeny, but we would conjecture that this is less of a problem in breakpoint analysis than with other approaches.

An important assumption in this work has been the fixed set of genes present in the data genomes. This is unrealistic in many contexts, but relaxing it makes computation for exact multiple rearrangement and genomic reconstruction (such as in the Sankoff and Blanchette (1997) solution) much more costly.

ACKNOWLEDGMENTS

Research supported by grants to DS from the Natural Sciences and Engineering Research Council of Canada (NSERC) and the Canadian Genome Analysis and Technology program, and a NSERC fellowship

for graduate studies to MB. DS is a Fellow of the Canadian Institute for Advanced Research. Parts of this paper were presented at COCOON 97 (Sankoff and Blanchette, 1997), the Genome Informatics Workshop 97 (Blanchette *et al.*, 1997), and RECOMB'98.

REFERENCES

Bafna, V., and Pevzner, P.A. 1996. Genome rearrangements and sorting by reversals. *SIAM Journal of Computing* 25, 272–289.

Bafna, V., and Pevzner, P.A. 1995. Sorting by transpositions, *Proceedings of the 6th Annual ACM-SIAM Symposium on Discrete Algorithms*, 614–623.

Blanchette, M., Bourque, G., and Sankoff, D. 1997. Breakpoint phylogenies, 25–34. *In* Miyano, S., and Takagi, T., eds., *Genome Informatics 1997*, Universal Academy Press, Tokyo.

Blanchette, M., Kunisawa, T., and Sankoff, D. 1996. Parametric genome rearrangement. *Gene-Combis* (online) and *Gene* 172, GC11–17.

Caprara, A. 1997a. Sorting by reversals is difficult, 75–83. In *Proceedings of the First Annual International Conference on Computational Molecular Biology (RECOMB 97)*, ACM, New York.

Caprara, A. 1997b. Formulations and complexity of multiple sorting by reversals. ms., University of Bologna.

DasGupta, B., Jiang, T., Kannan, S., Li, M., and Sweedyk, Z. 1997. On the complexity and approximation of syntenic distance, 99–108. In *Proceedings of the First Annual International Conference on Computational Molecular Biology (RECOMB 97)*, ACM, New York.

Ferretti, V., Nadeau, J.H., and Sankoff, D. 1996. Original synteny, 159–167. *In* Hirschberg, D., and Myers, G., eds., *Combinatorial Pattern Matching, 7th Annual Symposium. Lecture Notes in Computer Science* 1075, Springer Verlag, New York.

Gu, Q.-P., Iwata, K., Peng, S., and Chen, Q.-M. 1997. A heuristic algorithm for genome rearrangements, 268–269. *In* Miyano, S., and Takagi, T., eds., *Genome Informatics 1997*, Universal Academy Press, Tokyo.

Gusfield, D. 1997. *Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology.* Cambridge University Press.

Hannenhalli, S. 1995. Polynomial algorithm for computing translocation distance between genomes, 162–176. In *Combinatorial Pattern Matching. 6th Annual Symposium. Lecture Notes in Computer Science*, Springer Verlag, New York.

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

Hannenhalli, S., and Pevzner, P.A. 1995a. Transforming cabbage into turnip. (polynomial algorithm for sorting signed-permutations by reversals), 178–189. In *Proceedings of the 27th Annual ACM-SIAM Symposium on the Theory of Computing.*

Hannenhalli, S., and Pevzner, P.A. 1995b. Transforming men into mice (polynomial algorithm for genomic distance problem), 581–592. In *Proceedings of the IEEE 36th Annual Symposium on Foundations of Computer Science.*

Kaplan, H., Shamir, R., and Tarjan, R.E. 1997. Faster and simpler algorithm for sorting signed permutations by reversals. In *Proceedings of the 8th Annual ACM-SIAM Symposium on Discrete Algorithms.*

Kececioglu, J., and Ravi, R. 1995. Of mice and men. Evolutionary distances between genomes under translocation, 604–613. In *Proceedings of the 6th Annual ACM-SIAM Symposium on Discrete Algorithms.*

Kececioglu, J., and Sankoff, D. 1993. Exact and approximation algorithms for the inversion distance between two chromosomes, 87–105. *In* Apostolico, A., Crochemore, M., Galil, Z., and Manber, U., eds., *Combinatorial Pattern Matching. 4th Annual Symposium. Lecture Notes in Computer Science* 684, Springer Verlag, New York.

Kececioglu, J., and Sankoff, D. 1994. Efficient bounds for oriented chromosome inversion distance, 307–325. *In* Crochemore, M., and Gusfield, D., eds., *Combinatorial Pattern Matching. 5th Annual Symposium. Lecture Notes in Computer Science* 807, Springer Verlag, New York.

Kececioglu, J., and Sankoff, D. 1995. Exact and approximation algorithms for sorting by reversals, with application to genome rearrangement. *Algorithmica* 13, 180–210.

Minieka, E. 1978. *Optimization Algorithms for Networks and Graphs, Industrial Engineering vol. 1.* Marcel Dekker, New York.

Pe'er, I., and Shamir, R. 1998. The median problems for breakpoints are NP-complete. M.S., University of Washington.

Pevzner, P.A., and Waterman, M.S. 1995. Open combinatorial problems in computational molecular biology, 158–173. In *Proceedings of the 3rd Israel Symposium on the Theory of Computing and Systems.*

Sankoff, D. 1992. Edit distance for genome comparison based on non-local operations, 121–135. In *Combinatorial Pattern Matching. 3rd Annual Symposium. Lecture Notes in Computer Science* 644, Springer Verlag, New York.

Sankoff, D. 1997. The early introduction of dynamic programming into computational biology, 403–413. *In* Vinet, L., ed., *Advances in the Mathematical Sciences - CRM's 25 years. CRM Proceedings and Lecture Notes* 11, American Mathematical Society, Providence, RI.

Sankoff, D., and Blanchette, M. 1997. The median problem for breakpoints in comparative genomics, 251–263. *In* Jiang, T., and Lee, D.T., eds., *Computing and Combinatorics, Proceedings of COCOON '97, Lecture Notes in Computer Science* 1276, Springer Verlag, New York.

Sankoff, D., Cedergren, R.J., and Lapalme, G. 1976. Frequency of insertion-deletion, transversion, and transition in the evolution of 5S ribosomal RNA. *J. Mol. Evol.* 7, 133–149.

Sankoff, D., Ferretti, V., and Nadeau, J.H. 1997. Conserved segment identification. *Journal of Computational Biology* 4, 559–565.

Sankoff, D., Leduc, G., Antoine, N., Paquin, B., Lang, B.F., and Cedergren, R.J. 1992. Gene order comparisons for phylogenetic inference: Evolution of the mitochondrial genome. *Proceedings of the National Academy of Sciences USA* 89, 6575–6579.

Sankoff, D., Morel, C., and Cedergren, R.J. 1973. Evolution of 5S RNA and the non-randomness of base replacement. *Nature New Biology* 245, 232–234.

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

Schöniger, M., and Waterman, M.S. 1992. A local algorithm for DNA alignment with inversions. *Bull. Math. Biol.* 54, 521–536.

Setubal, J., and Meidanis, J. 1997. *Introduction to Computational Molecular Biology.* PWS Publishing, Boston.

Varré, J.-S., Delahaye, J.-P., and Rivals, E. 1997. The transformation distance, 352–353. *In* Miyano, S., and Takagi, T., eds., *Genome Informatics 1997*, Universal Academy Press, Tokyo.

Waterman, M.S. 1995. *Introduction to Computational Biology.* Chapman and Hall, London.

Address reprint requests to:
David Sankoff
Centre de recherches mathématiques
Université de Montréal
CP 6128 Succursale Centre-Ville
Montréal, Québec H3C 3J7
Canada

sankoff@ere.umontreal.ca

Received for publication March 1, 1998; accepted as revised May 20, 1998.

This article has been cited by:

1. J. Fostier, S. Proost, B. Dhoedt, Y. Saeys, P. Demeester, Y. Van de Peer, K. Vandepoele. 2011. A Greedy, Graph-Based Algorithm for the Alignment of multiple Homologous Gene Lists. *Bioinformatics* . [CrossRef]
2. C. Zheng. 2010. Pathgroups, a dynamic data structure for genome reconstruction problems. *Bioinformatics* **26**:13, 1587-1594. [CrossRef]
3. Tao Jiang. 2010. Some Algorithmic Challenges in Genome-Wide Ortholog Assignment. *Journal of Computer Science and Technology* **25**:1, 42-52. [CrossRef]
4. David Sankoff, Chunfang Zheng, Adriana Muñoz, Zhenyu Yang, Zaky Adam, Robert Warren, Vicky Choi, Qian Zhu. 2010. Issues in the Reconstruction of Gene Order Evolution. *Journal of Computer Science and Technology* **25**:1, 10-25. [CrossRef]
5. Andrew Wei Xu . 2009. A Fast and Exact Algorithm for the Median of Three Problem: A Graph Decomposition ApproachA Fast and Exact Algorithm for the Median of Three Problem: A Graph Decomposition Approach. *Journal of Computational Biology* **16**:10, 1369-1381. [Abstract] [PDF] [PDF Plus]
6. Denis Bertrand , Mathieu Blanchette , Nadia El-Mabrouk . 2009. Genetic Map Refinement Using a Comparative Genomic ApproachGenetic Map Refinement Using a Comparative Genomic Approach. *Journal of Computational Biology* **16**:10, 1475-1486. [Abstract] [PDF] [PDF Plus]
7. Jian Ma , Aakrosh Ratan , Brian J. Raney , Bernard B. Suh , Louxin Zhang , Webb Miller , David Haussler . 2008. DUPCAR: Reconstructing Contiguous Ancestral Regions with DuplicationsDUPCAR: Reconstructing Contiguous Ancestral Regions with Duplications. *Journal of Computational Biology* **15**:8, 1007-1027. [Abstract] [PDF] [PDF Plus]
8. Denis Bertrand , Mathieu Lajoie , Nadia El-Mabrouk . 2008. Inferring Ancestral Gene Orders for a Family of Tandemly Arrayed GenesInferring Ancestral Gene Orders for a Family of Tandemly Arrayed Genes. *Journal of Computational Biology* **15**:8, 1063-1077. [Abstract] [PDF] [PDF Plus]
9. William Arndt , Jijun Tang . 2008. Improving Reversal Median Computation Using Commuting Reversals and Cycle InformationImproving Reversal Median Computation Using Commuting Reversals and Cycle Information. *Journal of Computational Biology* **15**:8, 1079-1092. [Abstract] [Full Text] [PDF] [PDF Plus]
10. W. Xu, B. Alain, D. Sankoff. 2008. Poisson adjacency distributions in genome comparison: multichromosomal, circular, signed and unsigned cases. *Bioinformatics* **24**:16, i146-i152. [CrossRef]
11. Jianxiu Hao. 2007. An algorithm for reversal median problem. *Journal of Mathematical Chemistry* **42**:4, 849-857. [CrossRef]
12. Dr. Laxmi Parida . 2006. Using PQ Structures for Genomic Rearrangement PhylogenyUsing PQ Structures for Genomic Rearrangement Phylogeny. *Journal of Computational Biology* **13**:10, 1685-1700. [Abstract] [PDF] [PDF Plus]
13. Dr. Firas Swidan , Michal Ziv-Ukelson , Ron Y. Pinter . 2006. On the Repeat-Annotated Phylogenetic Tree Reconstruction ProblemOn the Repeat-Annotated Phylogenetic Tree Reconstruction Problem. *Journal of Computational Biology* **13**:8, 1397-1418. [Abstract] [PDF] [PDF Plus]
14. Li-San Wang, Tandy Warnow, Bernard M. E. Moret, Robert K. Jansen, Linda A. Raubeson. 2006. Distance-Based Genome Rearrangement Phylogeny. *Journal of Molecular Evolution* **63**:4, 473-483. [CrossRef]
15. Joe Felsenstein. 2006. WHAT MATHEMATICIANS WILL SEE OF PHYLOGENIES. *Evolution* **60**:4, 872-874. [CrossRef]
16. Joe Felsenstein. 2006. WHAT MATHEMATICIANS WILL SEE OF PHYLOGENIES1. *Evolution* **60**:4, 872. [CrossRef]
17. Weiping Wang, Bo Liao, Tianming Wang, Wen Zhu. 2006. A graphical method to construct a phylogenetic tree. *International Journal of Quantum Chemistry* **106**:9, 1998-2005. [CrossRef]
18. Ward C Wheeler. 2003. Search-based optimization. *Cladistics* **19**:4, 348-355. [CrossRef]
19. Alberto Caprara. 2003. The Reversal Median Problem. *INFORMS Journal on Computing* **15**:1, 93-113. [CrossRef]
20. Bret Larget, Donald L. Simon, Joseph B. Kadane. 2002. Bayesian phylogenetic inference from animal mitochondrial genome arrangements. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **64**:4, 681-693. [CrossRef]

21. B.M.E. Moret, Li-San Wang, T. Warnow. 2002. Toward new software for computational phylogenetics. *Computer* **35**:7, 55-64. [CrossRef]

22. Alberto Caprara. 2002. Additive Bounding, Worst-Case Analysis, and the Breakpoint Median Problem. *SIAM Journal on Optimization* **13**:2, 508. [CrossRef]

23. David A. Bader , Bernard M.E. Moret , Mi Yan . 2001. A Linear-Time Algorithm for Computing Inversion Distance between Signed Permutations with an Experimental StudyA Linear-Time Algorithm for Computing Inversion Distance between Signed Permutations with an Experimental Study. *Journal of Computational Biology* **8**:5, 483-491. [Abstract] [PDF] [PDF Plus]

24. David Sankoff , David Bryant , Mélanie Deneault , B. Franz Lang , Gertraud Burger . 2000. Early Eukaryote Evolution Based on Mitochondrial Gene Order BreakpointsEarly Eukaryote Evolution Based on Mitochondrial Gene Order Breakpoints. *Journal of Computational Biology* **7**:3-4, 521-535. [Abstract] [PDF] [PDF Plus]

25. David Sankoff , Mathieu Blanchette . 1999. Phylogenetic Invariants for Genome RearrangementsPhylogenetic Invariants for Genome Rearrangements. *Journal of Computational Biology* **6**:3-4, 431-445. [Abstract] [PDF] [PDF Plus]