Breakpoint Phylogenies

Mathieu Blanchette
blanchem@iro.umontreal.ca

Guillaume Bourque
bourque@crm.umontreal.ca

David Sankoff[1]
sankoff@ere.umontreal.ca

[1] Centre de recherches mathématiques, Université de Montréal
CP 6128 Succursale Centre-Ville, Montréal, Québec H3C 3J7

Abstract

We describe a number of heuristics for inferring the gene orders of the hypothetical ancestral genomes in a fixed phylogeny. The optimization criterion is the minimum number of breakpoints (pairs of genes adjacent in one genome but not the other) in the gene orders of two genomes connected by an edge of the tree, summed over all edges. The key to the method is an exact solution for trees with three leaves (the median problem) based on a reduction to the Travelling Salesman Problem.

1 Introduction

There have been a number of investigations of phylogeny of $N > 2$ genomes based on the pairwise comparison of the gene orders of these genomes, followed by distance matrix methods (e.g. [8]). Treeing methods based on the direct comparison of all N gene orders, which infer gene order at ancestral nodes [4, 9], have been little used because of the difficulty in generalizing measures of genomic distance to more than two genomes – there are no algorithms available, aside from rough heuristics, for handling even three relatively short genomes. Besides this technical problem,there are conceptual problems inherent in the use of rearrangement-event types of edit-distance, or their $N-$genome generalizations, for the purposes of reconstructing evolutionary history.

This include unwarranted assumptions as to the relative importance (i.e. costs) of reversals, transpositions, translocations and other rearrangement events (cf. [1]) and the fallacy that calculation of an edit distance allows the recoverability of the "true" history of genomic divergence – in fact, there is a proliferation of of optimal edit paths (and severe underestimation of the total number of events generating the divergence, cf. [5]) for moderate or large gene-order distances.

These problems all militate in favour of extending gene-order comparisons to three or more genomes through a much simpler and model-free metric, namely the number of breakpoints.

Consider two genomes $A = a_1 \dots a_n$ and $B = b_1 \dots b_n$ on the same set of genes $\{g_1, \dots, g_n\}$. We say a_i and a_{i+1} are adjacent in A (and a_n and a_1 are adjacent as well in circular genomes). 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.

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 this paper we offer a number of solutions to the problem of inferring ancestral gene order by minimizing the number of breakpoints associated with each edge of a given phylogenetic tree, summed over the entire tree. These involve the solution of the Travelling Salesman Problems (TSP) at each internal vertex of the tree, and an iterative approach to optimizing the entire tree. The approaches differ only in the initialization of the set of genomes associated to the internal vertices. Simulation experiments show that better initialization reduces the chances of converging to a non-global solution.

2 Steiner Points under the Breakpoints Metric.

The problem is formulated as follows: Let T=(V,E) be an unrooted binary tree with $N \geq 3$ leaves and $\Sigma = \{g_1, \cdots, g_n\}$ be a set of genes. Suppose $\{V_1, \cdots, V_N\} \subset V(T)$ are the leaves of the tree and $\{V_{N+1}, \cdots, V_{2N-2}\}$ are the internal vertices of the tree. The data consist, for each leaf $V_i, i = 1, \cdots, N$, of a circular permutation $G^i = g_1^i \cdots g_n^i$ of the genes in Σ, representing a contemporary genome. The task is to find the permutations $G^{N+1}, \cdots, G^{2N-2}$ associated with the internal (ancestral) vertices $V_{N+1}, \cdots, V_{2N-2}$, such that

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

is minimized.

3 The Median and the Travelling Salesman Problem.

The smallest problem of this type is that of finding the median, when $N = 3$: Given three genomes A, B and C, containing the genes in Σ, we want to find **median**(A, B, C), a genome S containing the genes in Σ such that

$$\Phi(S, A) + \Phi(S, B) + \Phi(S, C)$$

is minimized.

This can be reduced to the TSP as follows [2]. 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 three genomes. Set $w(gh) = 3 - u(gh)$. 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 A, for example, then they form a breakpoint in S.

3.1 Genomes with directionality

Our simulations will involve directed 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 [2]. 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. 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 three genomes A, B and C, and $w(gh) = 3 - u(gh)$, 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 $S = s_1 s_2 \ldots s_n$.

Proof: $\Phi(S, A) + \Phi(S, B) + \Phi(S, C) = \sum_{gh \in S, g \neq -h} w(gh)$

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

Thus S minimizes $\Phi(S, A) + \Phi(S, B) + \Phi(S, C)$ iff s is of minimal weight.

4 Median Algorithm Applied Iteratively to Phylogeny Decomposed into Overlapping Triples.

A general method for the inference of ancestral genomes on a fixed binary tree is the iterative improvement method of [7], as adapted for the genomics context in [9, 3]. Each of the $N-2$ internal vertices, together with its three neighbors, defines a 3-star. The solution to the Steiner point problem will have a reconstructed genome associated with each such vertex, which must be a solution to the median problem determined by these neighbors.

Then the following algorithm, 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 $G^1, \cdots, G^N$
$cost \leftarrow \infty$
$extremities \leftarrow \{1, \cdots, N\}$
$internal \leftarrow \{N+1, \cdots, 2N-2\}$
do for $M = N+1, \cdots, 2N-2$,
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 $G^{N+1}, \cdots, G^{2N-2}$

In each of Sections 4.2, 4.3 and 4.4 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).

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

4.1 Initialization strategies.

The output of this algorithm is not necessarily a global optimum. The main factor in directing convergence towards a global optimum, and the focus of this paper, is the how the initialization is carried out.

A promising initialization, which makes use of the most pertinent input data for each internal node, bases the initial TSP on the three nearest data genomes. In Section 4.2 we will use this latter idea as the basis of one of our heuristics, **three_nearest**. In addition, in Section 4.3, we define 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. Finally, in Section 4.4, we introduce an initialization method which involves 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 more "cautiously", i.e. 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 as the initialization proceeds, and its two neighbours are disregarded.

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.

4.2 Triangulation.

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

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.

4.3 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 leaf V_M of the tree, 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 internal vertices 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 $\mathbf{w}$ can then all be easily found by solving the system of simultaneous equations derived from all the internal vertices of T.

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

routine average_TSP
calculate $\mathbf{w}$ for the vertices in *internal* based on the vertices in *extremities*

4.4 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 $V_{N+1}, \cdots, V_{M-1}$ 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 ste-up routine below follows this direction towards V_M.

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 $remaining$

endwhile

do for all $gh \in \Gamma$

$w_M(gh) \leftarrow w_M^+(gh) - w_M^-(gh)$

enddo

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 value of r, 10 trees were generated, each starting with genome $(12 \cdots n)$ at one vertex and generating neighboring vertices with the appropriate random number of rearrangements until all internal and terminal vertices were assigned a genome. Each rearrangment was randomly chosen to be a transposition or an inversion (cf [1]), of random length.

Once all genomes were generated, the breakpoints on each edge were counted, and the tree was retained only if r was one of our target values for which we had not yet our quota of 10 examples. The genomes from the terminal vertices only then served as input for each of our three algorithms separately.

For solving our TSP problems we used C.Hurwitz' **tsp_solve** software on an Origin 200 computer with a RISC 10000 processor.

6 Results

It can be seen from Figure 1, that at 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 of reconstructed breakpoints actually levels off sharply.

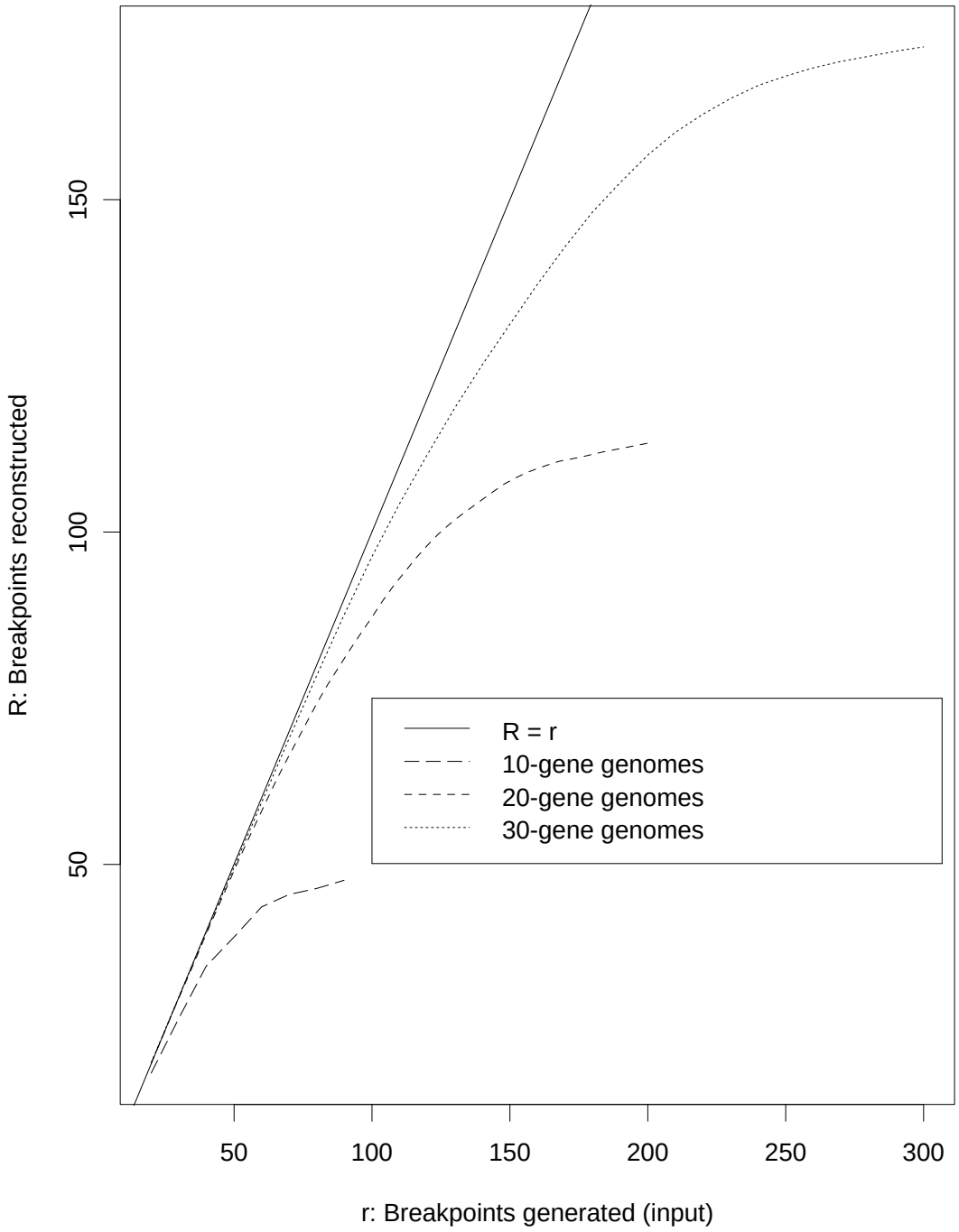

Figure 1. Number of reconstructed breakpoints R (best of three heuristics) as a function of number of breakpoints generated in the input data, for 10-gene, 20-gene and 30-gene genomes. Number of leaves $N = 7$, number of branches, $2N - 3 = 11$. The results of each n replaced by a smooth curve (generated using splines in the SPLUS package).

The accuracy of our initializations can be assessed in Figure 2, 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 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.

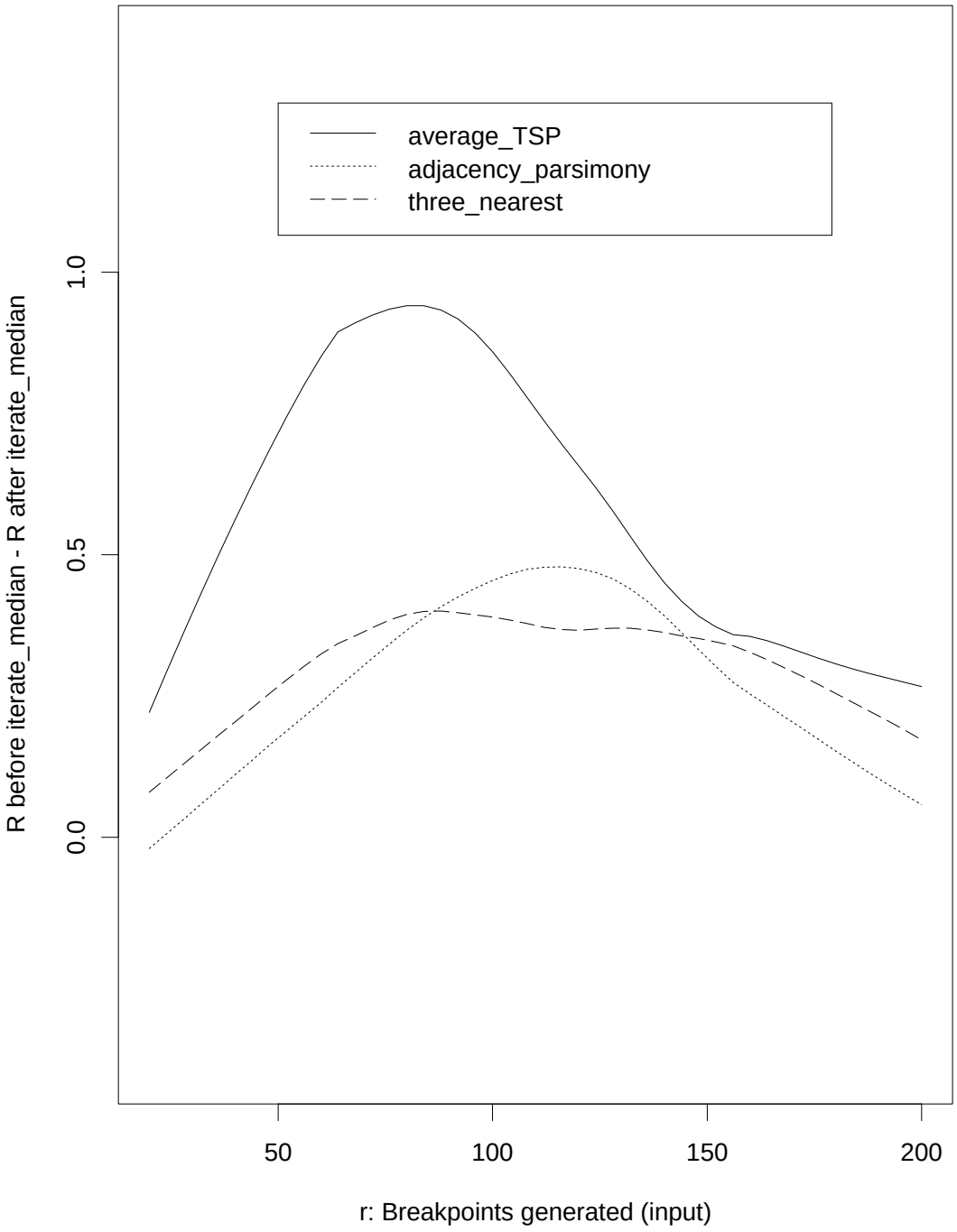

Figure 2. Decrease in number of reconstructed breakpoints R for each heuristic obtained through iteration step, as a function of number of breakpoints generated in the input data. $n = 20, N = 7$. Results for each heuristic replaced by a spline fit.

Figure 3 compares the performance of the two heuristics **average_TSP** and **adjacency_parsimony** (both outperform **three_nearest**) over a range of evolutionary divergences. 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.

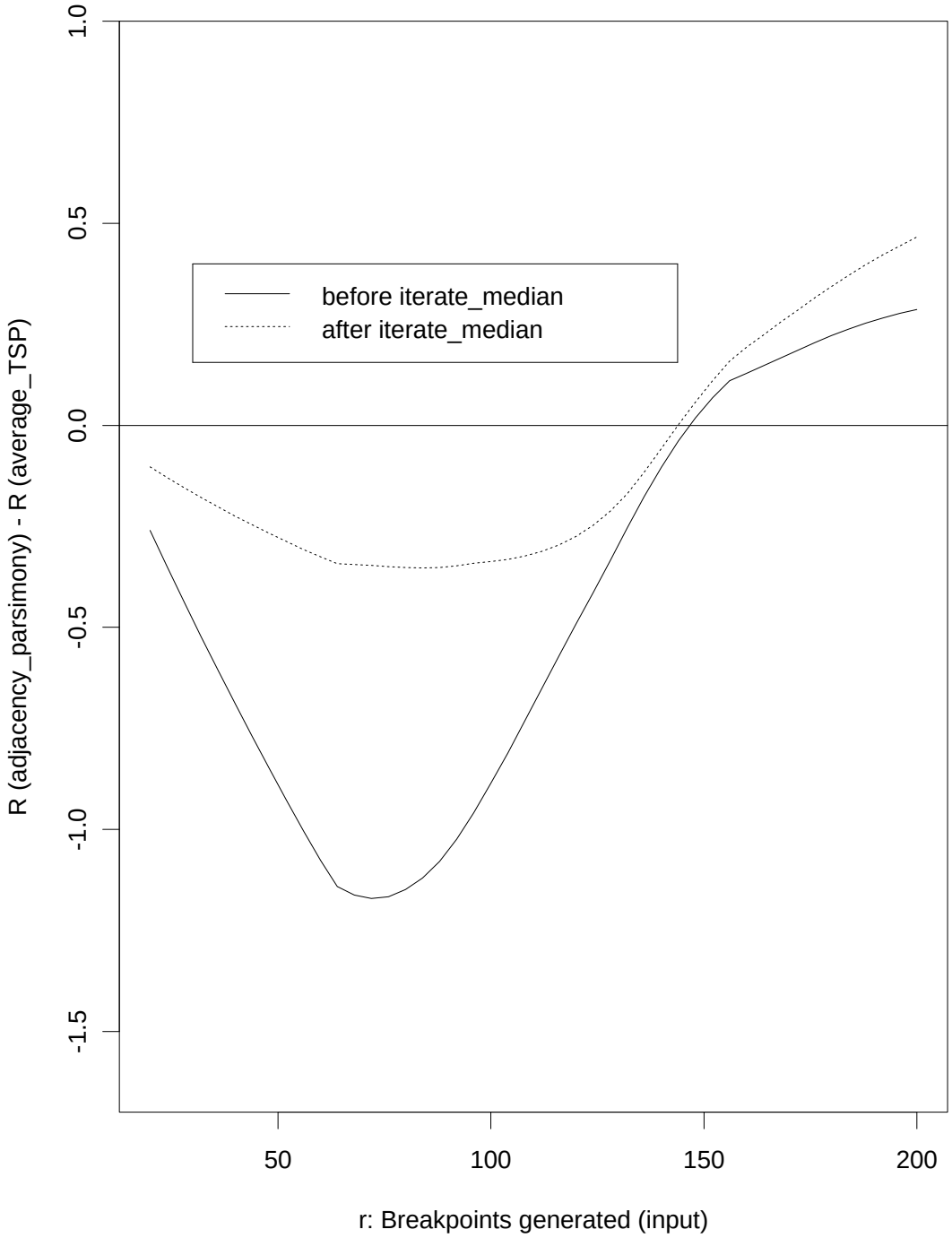

Figure 3. Difference between results of **adjacency_parsimony** and **average_TSP** as a function of r, before and after iterative improvements. $n = 20, N = 7$. Results for each set of differences replaced by a spline fit.

To address the question of global optimality, we count how many heuristics give the minimum solution for R. In Figure 4, 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 valid, of course),and since **adjacency_parsimony** and **average_TSP** are the ones that tend to achieve the lowest values, we can estimate that individually they attain global optimality about half of the time.

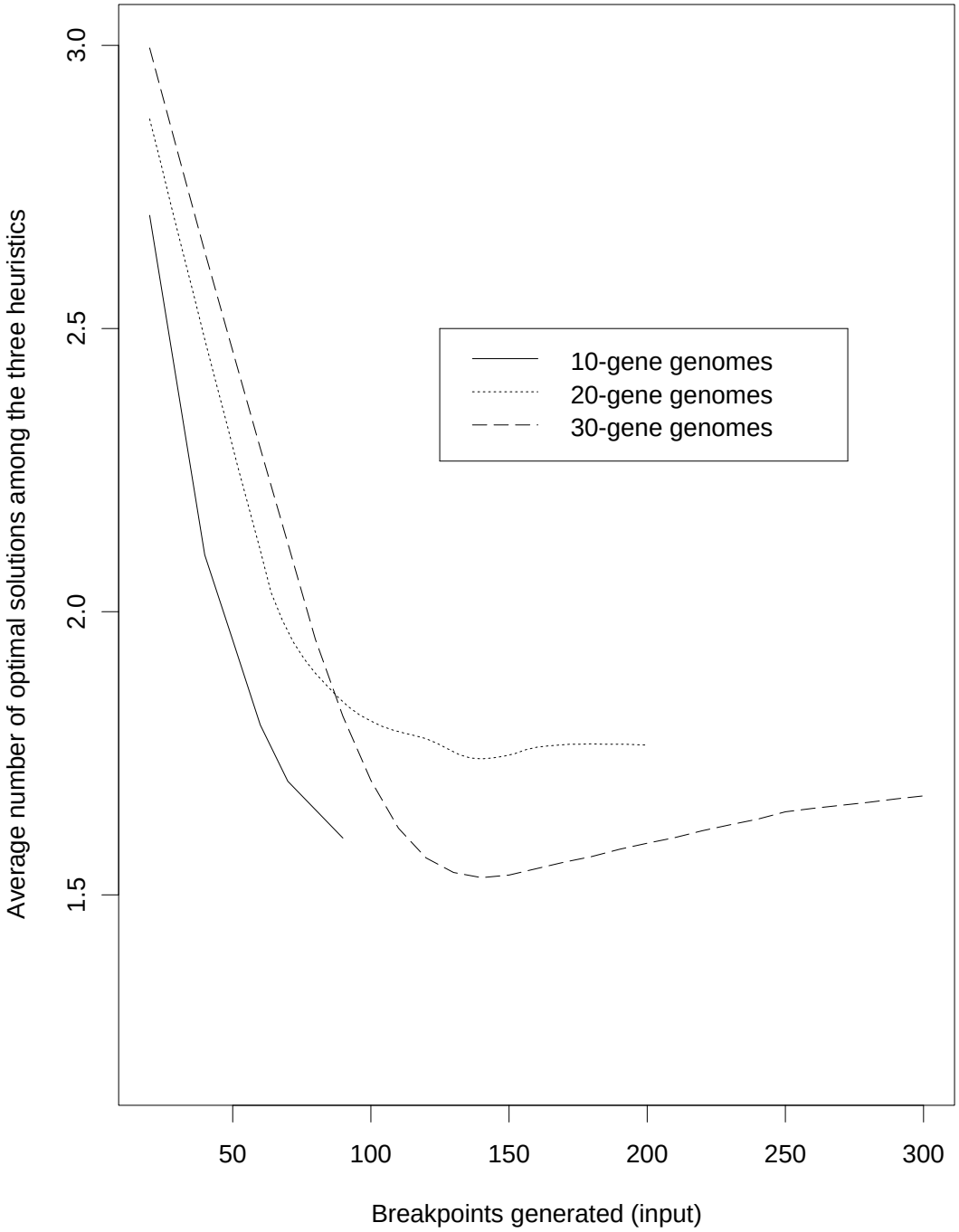

Figure 4. Number of heuristics (out of three) attaining optimal solution as a function of number of

breakpoints generated in the input data, for 10-gene, 20-gene and 30-gene genomes. $N = 7$. The results for each n replaced by a spline fit.

7 Summary and Conclusions.

We have proposed and tested three initializations for solving the breakpoint phylogeny problem 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.

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

We have found at what point parsimony leads to underestimation of the number of events generating the data. In another paper [6], we analyze the multiplicity of equivalent local minima and the breakpoint distances amonst them, as an assessment of the reliability of reconstructed gene orders.

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 the median problem, and hence, phylogenetic reconstruction, much more difficult [2]. Further work involves non-binary trees, as reported in [6].

Acknowledgements

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.

References

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

[2] Blanchette, M. and Sankoff, D., "The median problem for breakpoints in comparative genomics," *Computing and Combinatorics, Proceeedings of COCOON '97.*(T. Jiang and D.T. Lee, eds) Lecture Notes in Computer Science 1276, Springer Verlag, 251-263.

[3] V. Ferretti, J.H. Nadeau and D.Sankoff. Original synteny. *Combinatorial Pattern Matching. Seventh Annual Symposium* (D.Hirschberg and G.Myers, ed.) Lecture Notes in Computer Science 1075, Springer Verlag, 159-167, 1996.

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

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

[6] D. Sankoff and M. Blanchette. Multiple genome rearrangment. Manuscript, Centre de recherches mathématiques.

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

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

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