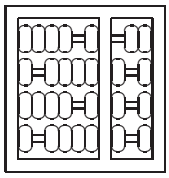

INSTITUTO DE COMPUTAÇÃO
UNIVERSIDADE ESTADUAL DE CAMPINAS

A Shortest-Cycle Based Heuristics for the Multiple Genome Rearrangement Problem

Cleber Mira *Qian Peng* *Joao Meidanis*
Pavel Pevzner

Technical Report - IC-06-023 - Relatório Técnico

November - 2006 - Novembro

A Shortest-Cycle Based Heuristics for the Multiple Genome Rearrangement Problem

Cleber Mira* Qian Peng † Joao Meidanis ‡ Pavel Pevzner §

Abstract

The multiple genome rearrangement problem consists in finding a phylogenetic tree that is the most "similar" to the evolutionary scenario based on rearrangement events involving several species. Bourque and Pevzner devised an algorithm for the multiple genome rearrangement problem that is based on rearrangement events (signed reversals, translocations, fusions, and fissions). However, the heuristics behind their algorithm relies on the time-consuming procedure of searching *good events*. We propose a simplified version of Bourque and Pevzner's heuristics based on experimental results that show that *proper* signed reversals acting on short cycles are frequently good events. We show the results of biological input data sets and randomly generated data sets and argue that the new heuristic is faster than the good events search, without compromising the quality of the solutions for input data sets involving few genomes.

Keywords: Median Problem, Genome Rearrangements, Computational Biology

1 Introduction

The comparison of the gene orders from two species can reveal the evolutionary scenario resulting from rearrangement events in the genomes of each species. By assuming that the evolution follows a parsimonious scenario, the problem of comparing two genomes can be formalized as finding a shortest sequence of rearrangement events that transforms one genome into other [10, 9, 8]. When the comparison involves more than two genomes then the evolutionary scenario is better described by a *phylogenetic tree* whose leaves are the species and the internal nodes are ancestors of that species. Reconstruction of phylogenetic trees based on events affecting a whole genome (genome rearrangements) started to be studied in the last decade [4, 12, 11, 3]. In the past, phylogenetic studies were mainly based on analysis in single genes [13, 6], but since the number of available, complete genomes increased recently, comparison between genomes became preeminent [10, 5].

*Institute of Computing, University of Campinas, 13081-970 Campinas, Brazil. Research supported by CAPES, grant #3041/05-0

†Department of Computer Science and Engineering, University of California, San Diego, La Jolla, California 92093, USA

‡Institute of Computing, University of Campinas, 13081-970 Campinas, Brazil.

§Department of Computer Science and Engineering, University of California, San Diego, La Jolla, California 92093, USA

In this paper, we deal with multichromosomal genomes such that each chromosome is linear.

A genome Π is the set $\{\pi(1), \ldots, \pi(N)\}$ of chromosomes $\pi(i)$, where $1 \leq i \leq N$. Given the sets of integers $E(n) = \{1, \ldots, n\}$ and $SE(n) = \{-1, +1, \ldots, -n, +n\}$, *Chromosomes* are modeled as *signed permutations*, that is a chromosome π is a mapping $\pi : E(n) \rightarrow SE(n)$ such that the function that maps $i \in E(n)$ to $|\pi(i)| \in E(n)$ is a permutation over $E(n)$. Each integer in $SE(n)$ corresponds to a gene and a sign corresponds to the orientation of a gene in the chromosome.

Given a genome $\Pi = \{\pi(1), \ldots, \pi(N)\}$, let

$$\pi = (\pi_1 \ \ldots \ \pi_{i-1} \ \pi_i \ \ldots \ \pi_j \ \pi_{j+1} \ \ldots \ \pi_n),$$

where $1 \leq i \leq j \leq n$, be a chromosome of Π. A *signed reversal* is rearrangement event $\rho(\pi, i, j)$ that inverts the segment from i to j in chromosome π and it changes the orientation of each gene π_k, with $i \leq k \leq j$, that is

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

Translocations are events that exchange segments between two chromosomes. A translocation $\rho(\pi, \sigma, i, j)$ transforms the chromosomes $\pi = (\pi_1 \ \ldots \ \pi_{i-1} \ \pi_i \ \ldots \ \pi_n)$ with $1 \leq i \leq n$ and $\sigma = (\sigma_1 \ \ldots \ \sigma_{j-1} \ \sigma_j \ \ldots \sigma_m)$ with $1 \leq j \leq m$ into chromosomes $(\pi_1 \ \ldots \ \pi_{i-1} \ \sigma_j \ \ldots \ \sigma_m)$ and $(\sigma_1 \ \ldots \ \sigma_{j-1} \ \pi_i \ \ldots \ \pi_n)$, with $(i-1) + (m-j+1)$ and $(j-1) + (n-i+1)$ genes respectively.

Fissions are events that split a chromosome in two. Given the chromosome $\pi = (\pi_1 \ \ldots \ \pi_{i-1} \ \pi_i \ \ldots \ \pi_n)$ with $1 \leq i \leq n$, the fission $\rho(\pi, i)$ transforms π into the chromosomes $(\pi_1 \ \ldots \ \pi_{i-1})$ and $(\pi_i \ \ldots \ \pi_n)$. *Fusions* are events that concatenate chromosomes π and σ, transforming them into the chromosome $(\pi_1 \ \ldots \ \pi_n \ \sigma_1 \ \ldots \ \sigma_m)$.

The *genomic distance*, $d(\Pi, \Gamma)$, between the genomes Π and Γ is the minimum number of reversals, translocations, fissions, and fusions that transforms Π into Γ. Finding the genomic distance can be done in polynomial time by using an algorithm from Hannenhalli and Pevzner [7].

Given a set of m genomes (over the same set of genes), the *Multiple Genome Rearrangement problem* (MGRP) consists in finding a phylogenetic tree T whose leaves are the m genomes such that the overall sum $D(T)$ of rearrangement distances for each pair of nodes in T is minimum.

$$D(T) = \sum_{\Pi, \Gamma \in T} d(\Pi, \Gamma).$$

The special case for three genomes is called the *Median Problem*.

Some algorithms [1, 12] to reconstruct the evolutionary scenario involving multiple genomes used the number of breakpoints as the distance between genomes, although the number of rearrangements events is more biologically relevant [2]. Bourque and Pevzner [3] devised the MGR algorithm for the Multiple Genome Rearrangement problem that is able to handle multichromosomal genomes and it is based on rearrangement events (signed reversals, translocations, fusions, and fissions), instead of breakpoints.

The MGR algorithm is based on the idea that a rearrangement event ρ in a leaf node Π_i, where $1 \leq i \leq m$, that brings Π_i "closer" to a common ancestor genome A of all the m genomes ensures a shortest path between any two genomes in the phylogenetic tree. The rearrangement event ρ is called a *sorting event*[1] when $d(\Pi_i, A) - d(\Pi_i\rho, A) = 1$ for

[1] In order to clarify the discussion, we use a terminology that is different from Bourque and Pevzner [3]. In their terminology both *sorting events* and *good events* are called good events.

$1 \leq i \leq m$. Since the common ancestor A is unknown, it is not clear how to find a sorting event. Bourque and Pevzner [3] argue that a rearrangement event ρ that reduces the overall rearrangement distance between Π_i and the remaining $m-1$ genomes are likely to be a sorting event. The overall rearrangement distance reduction, denoted by $\Delta(\rho, \Pi_i)$ is

$$\Delta(\rho, \Pi_i) = \sum_{j \neq i} d(\Pi_i, \Pi_j) - \sum_{j \neq i} d(\Pi_i \rho, \Pi_j)$$

A rearrangement event ρ in a genome Π_i is a *good event* when $\Delta(\rho, \Pi_i) = m - 1$.

For the special case of the Median problem, the MGR heuristics (called MGR-Median) consists in finding a good event for each of the genomes Π_1, Π_2, and Π_3 and iteratively carrying these rearrangement events until each genome in this triple is transformed into a common ancestor. In each iteration, there may be several good events for a genome (or none at all!). The good event ρ in Π that maximizes the number of good events in $\Pi\rho$ is called a *best event* and it is the rearrangement chosen at each iteration. If there is no good event available for any genome in the triple then a k-depth search is implemented in order to find a sequence of k events $\rho_1, \ldots, \rho_k$ that minimizes the total pairwise distances between the genomes and ρ_1 is the best event chosen.

In the MGRP case, the MGR heuristics finds the best event at each iteration. The best event is carried out and if two genomes converge into an ancestor, one of the genomes is removed. If there is no good event for any genome then the MGR-Median solver heuristics is used in a triple such that its pairwise distance is minimum and two of the genomes u and v in the triple are linked by an edge e in the tree, while the third genome w is not in the tree. The genome w is chosen to be the "closest" to the tree and the edge $e = (u, v)$ is chosen depending on the solution of the Median problem for u, v, and w. Given the common ancestor M recovered by the MGR-Median solver, the edge e is the one whose cost $C(u, v)$ is minimum, where $C(u, v) = d(u, M) + d(v, M) + d(w, M) - d(u, v)$. Finally, a sequential addition procedure is performed: the edge e is split and the triple is linked to the common ancestor M.

The MGR algorithm overcomes the limitation of breakpoint based algorithms since it uses as parameter the distance based on biologically meaningful rearrangement events and it was shown that it generates more accurate trees. However, the performance of the heuristics behind MGR algorithm is heavily influenced by the search for good events and best events. We propose a simplified version of Bourque and Pevzner's heuristics based on empirical evidence that among events acting on gray edges of short cycles in the breakpoint graph of a pair of genomes there are frequently good reversals and among these good reversal there is often at least one proper reversal. Instead of searching for a good event in all the events in a genome, we perform a search in the events acting on gray edges of the breakpoint graph of a pair of genomes. If a good event is found among the events acting on gray edges then the search finishes, the good event is properly stored in a list of good events and a new search for good events is performed in another pair of genomes. In order to avoid to start a k-depth search procedure in the case that there is no good event, we store the proper reversals that decreases by one the distance from the current genome to at least one other genome (safe reversals). Moreover, we avoid the search for a best event, unless no good event or safe reversal was found among all the events for all the pairs of genomes. These simple modifications allow some improvement in input data sets involving few genomes with a small number of chromosomes despite a large amount of genes.

The paper is organized as follows. In Section 2 we present a summary of the main concepts of the multiple genome rearrangement distance problem, defining formally the

breakpoint graph for a pair of permutations, showing how to represent a genome as a permutation. In Section 3 we present the short cycles based heuristics. We show empirical data that supports the heuristic. In Section 4 we show the results of experiments involving biological and randomly generated data and we compare to the heuristics of Bourque and Pevzner [3]. We summarize the results in Section 5.

2 Proper Reversals and Short Cycles

In this section, we present some basic concepts used in genome rearrangements [7]. Given a permutation $\pi = (\pi_1 \ \ldots \ \pi_n)$ over $\{1, \ \ldots, \ n\}$, its *extended permutation* is the permutation $(\pi_0 \ \pi_1 \ \ldots \ \pi_n \ \pi_{n+1})$ over $\{0, 1, \ \ldots, \ n, \ n+1\}$ where $\pi_0 = 0$ and $\pi_{n+1} = n+1$. For the extended permutation π, the pair of consecutive elements π_i and π_{i+1}, for $0 \leq i \leq n$, is a *breakpoint* when $|\pi_i - \pi_{i+1}| \neq 1$, otherwise the pair is called an *adjacency*. The *breakpoint graph* $G(\pi)$ of the extended permutation π is the graph whose vertices set is $\{0, \pi_1, \ldots, \pi_n, n+1\}$ and two vertices π_i and π_j are linked by a *black edge* when $|i-j| = 1$ and by a *gray edge* when π_i and π_j is an adjacency. A signed permutation $\vec{\pi}$ over $\{-1, \ +1, \ -2, \ +2, \ \ldots, \ -n, \ +n\}$ can be transformed into a permutation over $\{1, \ 2, \ \ldots, \ 2n\}$ by replacing each element $+x$ by $2x-1, 2x$ and each element $-x$ by $2x, 2x-1$. The permutation π that outcomes from the transformation of $\vec{\pi}$ is the *image* of $\vec{\pi}$. The breakpoint graph $G(\vec{\pi})$ of $\vec{\pi}$ is defined as the breakpoint graph of the image of $\vec{\pi}$. Since every vertex in $G(\vec{\pi})$ is incident to exactly one black edge and one gray edge then $G(\vec{\pi})$ is composed by disjoint cycles. The number of cycles of $G(\vec{\pi})$ is denoted by $c(\vec{\pi})$. A signed reversal applicable to a signed permutation is mimicked by a reversal (it does not change the sign) in the image. Indeed, it is possible to find an equivalent sequence of signed reversals that transforms a signed permutation into another for any sequence of reversals that transforms the images of those signed permutations [7].

Given a breakpoint graph $G(\pi)$ of π, a gray edge (π_i, π_j) of $G(\pi)$ is *oriented* when $i+j$ is even, otherwise it is *unoriented*. A cycle is oriented when it contains an oriented edge. We say that a reversal $\rho(i,j)$ *cuts* the black edges (π_{i-1}, π_i) and (π_j, π_{j+1}) in π. A reversal ρ *acts on* a gray edge e when it cuts the black edges incident to e. We denote the *difference* in the number of cycles due the application of a reversal ρ in a permutation π as $\Delta c(\rho) = c(\pi\rho) - c(\pi)$. The reversal ρ acting on a oriented edge is called a *proper reversal* and $\Delta c(\rho) = +1$ when ρ is a proper reversal. Given a cycle C of the breakpoint graph $G(\pi)$ the *length* of the cycle C, denoted by $l(C)$, is the number of black edges in the cycle. A cycle C in $G(\pi)$ is a *short cycle* when $l(C) \leq 4$.

A multichromosomal genome Π can be represented by a permutation using some transformations of its chromosomes. Given two chromosomes $\pi = (\pi_1 \ \ldots \ \pi_n)$ and $\sigma = (\sigma_1 \ \ldots \ \sigma_n)$ of Π, the *fusion* $\pi + \sigma$ of π and σ is $\pi + \sigma = (\pi_1 \ \ldots \ \pi_n \ \sigma_1 \ \ldots \ \sigma_n)$ and the fusion $\pi - \sigma$ is $\pi - \sigma = (\pi_1 \ \ldots \ \pi_n \ -\sigma_n \ \ldots \ -\sigma_1)$. Consider a ordering $(\pi(1), \ \ldots, \ \pi(N))$ of the chromosomes in Π and a *flip* vector $s = (s(1), \ s(2), \ \ldots, \ s(N))$ where $s(i) \in \{-1, +1\}$, the *concatenate* of Π with respect to s is the permutation $\Pi(s) = s(1)\pi(1) + \ldots + s(N)\pi(N)$. Since each chromosome π is equally represented by $\pi = (\pi_1 \ \ldots \ \pi_n)$ and $-\pi = (-\pi_n \ \ldots \ -\pi_1)$ and there are $N!$ orderings of the chromosomes in Π, then there are $2^N N!$ concatenates that model the same genome Π. Considering that a genome $\Gamma = (\gamma_1, \ \ldots, \ \gamma_N)$ is a *ordered genome*; that is, there is a fixed ordering for its chromosomes, then we assume that the concatenate $\gamma = \gamma_1 + \ldots + \gamma_N$ is the *identity permutation*. Hannenhalli and Pevzner [7] show that a minimum sequence of reversals that transforms a concatenate π of Π into the identity γ

can be mimicked by a (not necessarily optimal) sequence of reversals and translocations that transforms Π into Γ. They prove that the minimum sequence among all the sequences of events that transforms the concatenates of Π into γ is an optimal sequence. The problem of finding a sequence of rearrangement events that transforms one genome Π into the genome Γ is reduced to find the concatenate of Π such that the minimum sequence of reversals that transform it into γ is minimum over all concatenates.

3 Heuristic

In this section we present some empirical properties about the relation between proper reversals and good events

Figure 1 shows the results of experiments involving randomly generated triples of genomes that support the idea that the odds of a good event be a proper reversal and that this proper reversal acts on the gray edge of a short cycle are both big. The experiment consists of generating three unichromosomal genomes by carrying a sequence of k randomly chosen reversals in the permutation $(1\ 2\ 3\ \ldots\ n)$ where $n = 300$. Using the generated data set, a search for all the events that acts on the gray edges of the breakpoint graph for each pair of genomes sums up the number of **proper reversals**, **good events**, **proper, good reversals**, and rearrangement events acting on edges of short cycles (**events on short cycles**). These numbers are the result of the sum for the three pairs of genomes. The number of observed good events for the randomly generated data is small, even for a small number of genomes, because events acting on grey edges are the only ones being searched. An important finding in the simulation is that the number of good events that are also proper reversals is significant. In roughly 60% of the experiments at least one of the good events was found among the proper reversals. This information is significant because it supports the idea that, in most cases, it is sufficient to search the proper reversals only in order to find a good event. The number of proper reversals is bounded by the number of genes in the genomes, instead of the total number of events that is bounded by the square of the number of genes. It is also significant that most of the good events in the table involved a vertice belonging to a short cycle. This observation offers an alternative search that priorizes the edges of short cycles.

A search for good events that involves looking into fewer rearrangement events than the total number of rearrangement events reduces the running time of finding a good event, however this search may increase the odds of occurring a *miss*, that is, the search might not find a good event among the events acting on gray edges, although there is a good event in the genome. Given a set of m genomes, a reversal ρ is a *safe reversal* when $\Delta(\rho, \Pi) = 1$ for some Π in the set of m genomes. Safe reversals are can be chosen to be the rearrangement event to be carried in case there is a miss, since they appear more frequently among the rearrangement events.

The Short Cycles heuristic to solve the MGR problem is a simple modification of the MGR algorithm devised by Bourque and Pevzner [3] and the new heuristic uses the former algorithm for dealing with a special case only. The heuristics relies on the empirical evidence that there is a proper reversal that is a good reversal with high probability and that good events acting on gray edges often involves a vertex of a short cycle. In order to speed up the identification of a good event, the search is restricted to events acting on gray edges of short cycles only.

For each pair of genomes in the set of genomes, an *optimally capped breakpoint graph*, that is a breakpoints graph for an optimal concatenate for one of the genomes in the

k	Proper Reversals	Proper, good Reversals	Good Events	Events(Short Cycles)
10	37	0	0	108
20	74	0	0	196
30	115	0	0	228
40	145	0	0	272
50	194	0	0	276
60	220	0	0	242
70	246	0	0	216
80	263	5	8	202
90	296	6	9	186
100	307	9	15	148
110	350	6	13	164
120	358	7	14	98
130	374	5	8	100
140	352	25	41	62
150	397	24	49	48
160	366	25	64	58
170	417	69	119	38
180	412	17	43	28
190	401	28	50	26
200	421	60	98	14
210	430	58	111	10
220	432	19	43	12
230	406	25	56	12
240	422	41	77	6
250	435	91	182	4
260	429	27	44	12
270	444	18	46	4
280	438	27	46	14
290	412	64	122	6
300	448	61	129	4

Table 1: The results from a simulation for a randomly generated data set are summarized in the table. The data set consists of three unichromosomal genomes that are obtained by k reversals each from the identity permutation (1 2 ... 300) ($n = 300$). The simulation was performed 10 times and the numbers in the table are averages over the number of repetitions. The columns show respectively, the number of events used to generate the data set, the number of proper reversals; the number of proper, good reversals; the number of good events; the number of rearrangement events acting on gray edges of short cycles.

pair, is constructed to obtain an array of the gray edges of the graph. The length of a cycle in the graph is assigned to each corresponding vertex that belongs to that cycle. All the events defined by gray edges that belong to a short cycle are checked in order to find out if they are good events. Firstly, the heuristics finds which kind of event acts on the current gray edge. If the event is a fission, fusion, or translocation then we check whether it is a good event and, in that case, the event is inserted in the list of *good events*. All the proper reversals that are also safe reversals, but not good reversals, are inserted in a list of *eligible reversals* while the good reversals are inserted in the list of good reversals. The event that is chosen to be carried out is randomly chosen in the list of good reversals if there is some event in the list, otherwise a event is picked in the list of eligible reversals. If the list of eligible reversals is empty then we do a search for a good event using the heuristics of Bourque and Pevzner [3], that is checking all possible events in the genomes and possibly doing a k-depth search whether there is no good event.

The Table 3 shows a pseudo-code implementation of the heuristic.

```
HeuristicSOC(Set of genomes Π)
  For each pair of genomes Π1 and Π2 in the set of genomes Π do:
    Find an optimal capped breakpoint graph B(Π1, Π2);
    Calculate the length of each cycle in the graph;
    Assign the length to each vertex that belongs to that cycle;
    For each event ρ acting on a gray edge belonging to a short cycle do:
        If ρ is a safe reversal
          then insert ρ into the list of eligible events;
        If ρ is a good event
          then insert ρ into the list of good events and stop the search.
   If the list of good event is not empty
    then pick randomly a rearrangement event in this list;
   else if the list of eligible events is not empty
      then pick randomly a rearrangement event in this list;
      else
       call Bourque and Pevzner (H5) heuristic;
       pick the first good event found;
  Carry the chosen rearrangement event;
```

Table 2: The heuristic of Short Cycles.

Complexity analysis The quality of the solution and the running time are influenced by the number of reversals that are searched. In the short cycles heuristic the reversals acting on gray edges are the only ones that are initially verified. If there is a good event among these reversals then it is included in a list of rearrangements and the search in the current pair of genomes is finished. On the other hand, if there is no good event among those reversals in any of the pairs of genomes then a thoroughly search is made using heuristics H5 (all the events in each genome are analyzed).

Finding an optimally capped breakpoint graph for a given pair of genomes takes $O(n^4)$ running-time [7]. Measuring the lengths of all cycles in the breakpoint graph is

$O(n)$. Finding which type is an event acting on a gray edge is $O(n)$. Checking if an event is a good event or a safe reversal is $O(mn^4)$ because it takes $O(n^4)$ to . Checking if there are events in the list of good events of list of eligible events and picking randomly an event in one of these lists is $O(1)$. Heuristic H5 takes $O(mn^6)$ running-time. Then the worst case running-time for the Short Cycles heuristic is $O(m^3n^5 + mn^6)$. The best case is $O(m^3n^5)$ and it happens when there is a scenario containing good events only and all these good events act on gray edges of short cycles. Since usually $m << n$ in biological data sets and experiment results support that randomly generated data sets often contain at least one good event that acts on a gray edge of a short cycle then the Short Cycles heuristic is expected to be more efficient than the heuristic proposed by Bourque and Pevzner [3]. However, experiments presented in the next section show that biological data sets do not have the same properties of randomly generated data sets.

4 Experiments and Analysis

In this section, we present comparison of the performance between the Short Cycles heuristic and other heuristics implemented by Bourque and Pevzner [3] in the MGR algorithm. The biological data sets used were the same in Bourque and Pevzner [3].

The algorithm MGR was devised to support different strategies to find the good event that will be carried in a certain iteration. These strategies are summarized below:

1. Look at reversals only initially and picks first good reversal (H3).
2. Look at reversals only initially and pick first good reversal that doesn't decrease the score in the list of eligible events (H4).
3. Look at reversals initially, take shortest reversal (H5).
4. Select good events that affect edges not suggested in other good events (H6).
5. Select rearrangement events based on a score that depends on pairwise distances of the genomes (H7).
6. Select rearrangement events based on the decrease in the number of breakpoints (H8).
7. Look at events acting on gray edges of short cycles; store safe reversals, pick first good event (H10).

The strategies from H3 to H8 were implemented by Bourgue and Pevzner [3] and the strategy H10 is the heuristic based on short cycles. All the strategies are based on the simple idea of looking for a good event, however other parameters as the pairwise distance, number of breakpoints, and length of a rearrangement event may be also used together to choose an event that could bring some improvements in running time or quality of solution.

Biological Data Sets Running time results from biological data sets for different heuristics are summarized in the Table 3. The data sets consist of the biological data used by Bourque and Pevzner [3]. These data sets are: Herpes viruses (a); human X chromosome (b); Metazoan mtDNA (c); Campanulaceae cpDNA (d); Human, Cat and Mouse (e); Human, Sea Urchin and Fruit Fly (f). The running time was calculated by the gnu-linux **time** application. The columns show the results for the several heuristics

implemented earlier by Bourque and Pevzner and the running time for the implementation of the Short Cycles heuristic proposed in this paper. The values in the table are the sum user + sys in the output of **time**, that is the total direct CPU cost of executing the program, not including CPU costs due to kernel processes. The experiments were performed in an AMD Athlon 1.3GHz, 256Kb (cache memory), 512Mb (main memory) workstation.

The Table 3 shows that the heuristic H5 is the most efficient when compared to all other heuristics, however the Short Cycles heuristic has the second fastest performance in the data sets human X chromosome(b); campanulaceae (d); and human, sea urchin and fruit fly (f). The bad performance of the Short Cycles heuristic (H10) mainly in the metazoan and campanulaceae data sets is a result of the lack of good events in the set of genomes of these data sets. These data sets demand that the program spends most of its time doing the k-depth search. In order to improve the performance in the metazoan and campanulaceae data sets, we modified the heuristics to start a search for good events in all the reversals if no good event was found in the events acting on gray edges only, however since some time is spent in the the previous search, then the running-time is slightly longer in H10 heuristic than in the H5 heuristic.

The quality of the solution of the Short Cycles heuristic (H10) was not as good as the quality of the solutions found by the heuristic H5. However, the score difference is less than 10% of the solution score of H5 heuristic for all data sets.

Randomly generated data sets We performed an experiment to compare the performances of the Short Cycles heuristic and the heuristic H5

The Table 5 lists the running time values for the heuristics H5 and H10 in a simulation involving genomes with 100 genes and 10 chromosomes. The heuristic H5 is roughly four times faster than the heuristic H10 (Short Cycles heuristic), even though the Short Cycles heuristic searches through a lower number of events than the heuristic H5. An explanation for this bad performance of the Short Cycles heuristic may be the result of a characteristic of the heuristics: the heuristic does not try to guarantee that the chosen event to be carried out is the one that maximizes the number of good events in the new set of genomes. Checking whether a event maximizes the number of good events is time-consuming since it involves finding all the good events in a set of genomes, however it is possible that, in a worst case scenario such that there is no good event in the set of genomes, the Short Cycles heuristic performs badly because in this case a search through all events in each genome and a k-depth search is performed besides the search in the events acting on gray edges of short cycles.

The Figure 1 shows a comparison between *score*, that is the number of rearrangement

Data set	**H3**	**H4**	**H5**	**H6**	**H7**	**H8**	**H10**
a	0.01s	0.01s	0.01s	0.01s	0.01s	0.05s	0.01s
b	0.01s	0.01s	0.01s	0.01s	0.01s	0.02s	0.01s
c	4m41.3s	68m21.6s	3m5.3s	4m45.5s	2m37.4s	23m16.1s	5m30s
d	19.84s	1m43.6s	3.16s	12.80s	8.97s	10m42.8s	8.2s
e	8.74s	8.34s	7.2s	13.31s	6.68s	58m35.2s	22.1s
f	4.1s	58.6s	2.3s	4.25s	3.0s	4m34.5s	2.8s

Table 3: Running time results from biological data sets for different heuristics.

Data set	**a**	**b**	**c**	**d**	**e**	**f**
H5	7	5	162	66	48	43
H10	7	5	172	69	48	45

Table 4: Number of rearrangement events (scores) for biological data sets in heuristics H5 and H10.

k	Time(s) **H5**	Time(s) **H10**
15	2.1s	15.5s
20	4.7s	22.6s
25	12.3s	44.9s
30	15.3s	1m20.6s
35	26.3s	1m31.87s
40	3m14.2s	9m2.1s
45	4m5.5s	17m12.2s
50	11m9.0	48m31.2s

Table 5: Running time results from randomly generated data sets for heuristics H5 and H10 are summarized in the table. The data sets consist of the randomly generated data with the parameters: $m = 3$ (number of species), $n = 100$ (number of genes), $b = 9($ 10 chromosomes). Similarly to the experiment whit biological data the running time was calculated by the gnu-linux **time** application and the values in the table are the sum user + sys in the output of **time**.

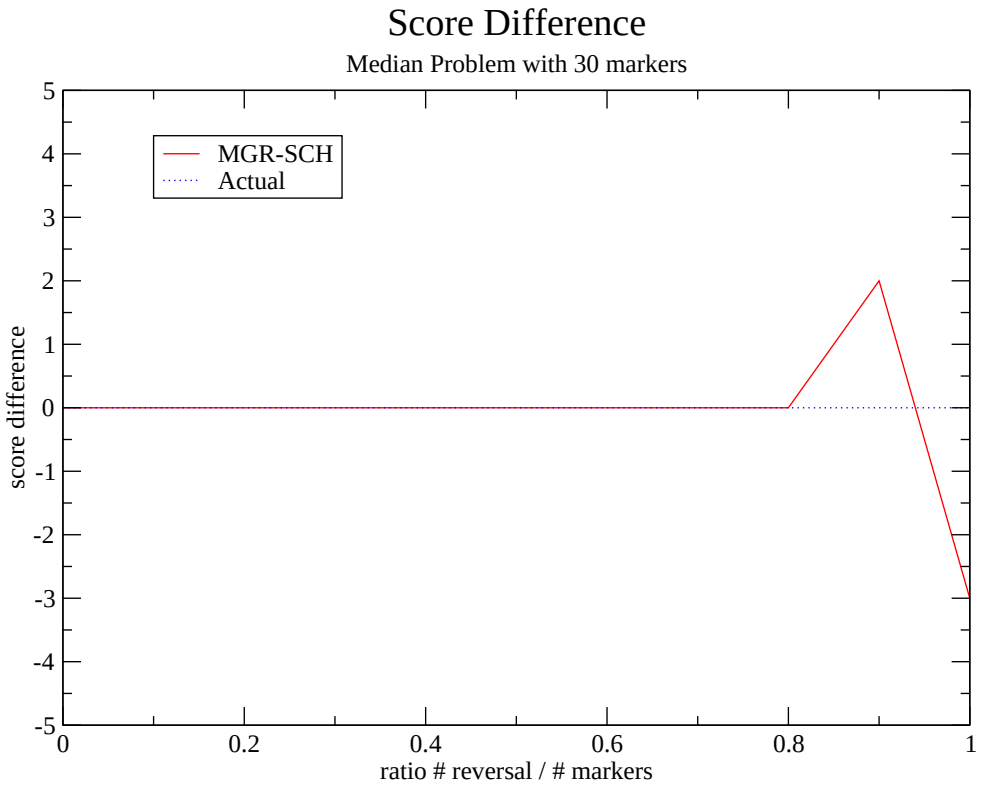

Figure 1: Comparison between the number of rearrangement events (score) obtained by Short Cycles Heuristic scenario and the number of rearrangement events of the actual randomly generated scenario. The data set is composed by three genomes ($m = 3$) equidistant by k reversals to the ancestral identity permutation (1 2 ... 30). Each simulation was repeated ten times for each ratio $\#reversals/\#genes = 3k/n$ ($n = 30$). The graph shows the average difference in the number of reversals in the tree (score) obtained by the heuristic and the actual number of reversals used to generate the data set.

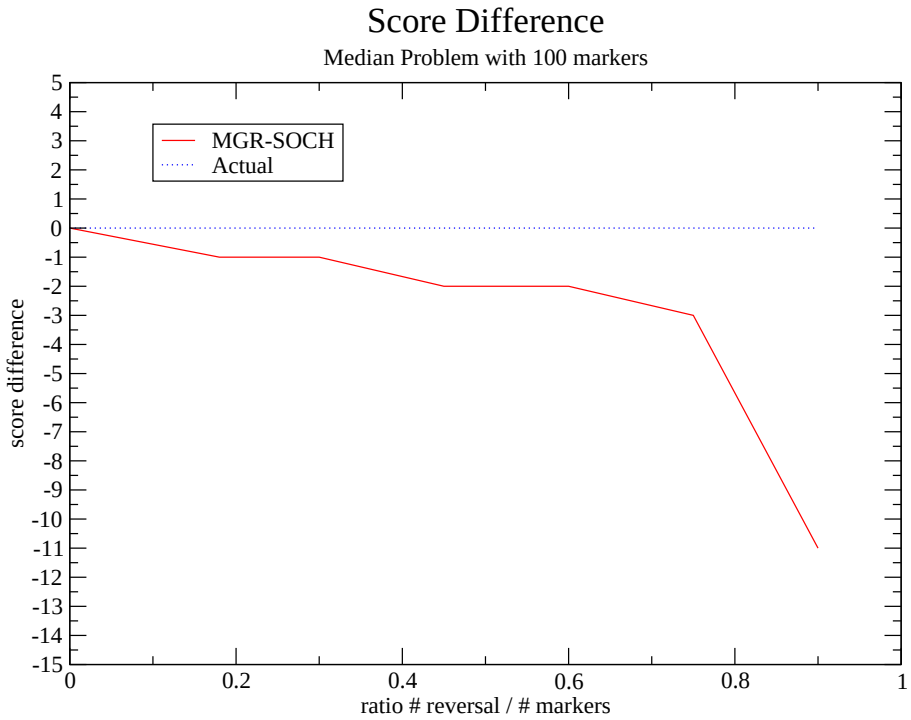

Figure 2: Comparison between the number of rearrangement events (score) obtained by Short Cycles Heuristic scenario and the number of rearrangement events of the actual randomly generated scenario. The data set is composed by three genomes ($m = 3$) equidistant by k reversals to the ancestral identity permutation (1 2 ... 100). Similarly to the case of genomes with 30 genes, each simulation was repeated ten times for each ratio $r = 3k/n$ ($n = 100$). The graph shows the average difference in the number of reversals in the tree (score) obtained by the heuristic and the actual number of reversals used to generate the data set.

events, in the tree obtained by the Short Cycles heuristic and the actual number of reversals used to generate the data set. The data set consists of three genomes that are equally distant from the identity permutation by k reversals. Each genome contains 30 genes and three chromosomes. For each ratio $r = \#reversal/\#genes$ the curve shows the average difference between the scores for ten repetitions of each experiment. The heuristics accurately finds the score used in the data set for $r \leq 0.8$, but it fails for $r > 0.8$. There is a slightly improvement in this curve if compared to the curve built using the same parameters by Bourque and Pevzner [3] that starts to find different scores for $r \geq 0.7$.

The Figure 2 shows the results for a experiment involving genomes with 100 genes similar to the previous experiment. However, besides the difference in the number of genes, the number of chromosomes in each genome is ten. In this experiment, the solutions found by the Short Cycles heuristic were different from the actual number of reversals for all ratios r. However, the difference in the scores is small (less than 10% of k) for $r \leq 0.6$.

5 Conclusion

Finding new heuristics that improve the performance or resolution quality of the MGR algorithm may be an important step for the improvement of algorithms for the Multiple Genome Rearrangement Problem. We propose an efficient version of the MGR heuristics based on looking rearrangement events acting on gray edges of short cycles. Most of these rearrangement events are proper reversals and empirical data evidences that the probability of a good event being a proper reversal is not low. We argue that such reversals are common in breakpoint graphs from both simulated and biological data, but we show that the heuristics performs badly for some kinds of data sets. Since the performance of the algorithm is $O(m^3n^5 + mn^6)$, we show that, although the running time is adequate for data sets with few genomes, the running time for data sets with many genomes is higher than the Bourque and Pevzner heuristic. The quality of the solution given by the Short Cycles heuristic is only marginally affected when compared to the quality of solution of the heuristic designed by Bourque and Pevzner.

Searching for new relations among kinds of rearrangement events and randomized approximation algorithms are promising directions for future work.

References

[1] M. Blanchette, G. Bourque, and D. Sankoff. Breakpoint phylogenies. In S. Miyano and T. Takagi, editors, *Genome Informatics*, pages 25–34, Tokyo, Japan, 1997. Universal Academy.

[2] G. Bourque, P. Pevzner, and G. Tesler. Reconstructing the genomic architecture of ancestral mammals: Lessons from human, mouse, and rat genomes. *Genome Research*, 4(14):507–516, 2004.

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

[4] A. Caprara. Formulations and hardness of multiple sorting by reversals. In S. Istrail, P. A. Pevzner, and M. Waterman, editors, *Proceedings of the 3rd Annual International Conference on Computational Molecular Biology (RECOMB'99)*, pages 84–93, Lyon, France, 1999. ACM Press.

[5] T. G. I. S. Consortium. Initial sequencing and analysis of the human genome. *Nature*, 409:860–921, 2001.

[6] D. Graur and W.-H. Li. *Fundamentals of Molecular Evolution*. Sinauer Associates, Sunderland, Massachusetts, 2000.

[7] S. Hannenhalli and P. A. Pevzner. Transforming men into mice (polynomial algorithm for genomic distance problem). In *Proceedings of the 36th Annual Symposium on Foundations of Computer Science (FOCS'95)*, pages 581–592, Los Alamitos, USA, Oct. 1995. IEEE Computer Society Press.

[8] S. Hannenhalli and P. A. Pevzner. Transforming cabbage into turnip: Polynomial algorithm for sorting signed permutations by reversals. *Journal of the ACM*, 46(1):1–27, Jan. 1999.

[9] J. Kececioglu and D. Sankoff. Exact and approximation algorithms for sorting by reversals. Technical Report 1824, Centre de Recherches mathématiques, Université de Montréal, July 1992.

[10] J. D. Palmer and L. A. Herbon. Plant mitochondrial DNA evolves rapidly in structure, but slowly in sequence. *Journal of Molecular Evolution*, 27:87–97, 1988.

[11] D. Sankof and M. Blanchette. Probability models for genome rearrangement and linear invariants for phylogenetic inference. In S. Istrail, P. A. Pevzner, and M. Waterman, editors, *Proceedings of the 3rd Annual International Conference on Com-*

putational Molecular Biology (RECOMB-99), pages 302–309, Lyon, France, Apr. 1999. ACM Press.

[12] D. Sankoff and M. Blanchette. Multiple genome rearrangements. In S. Istrail, P. A. Pevzner, and M. Waterman, editors, *Proceedings of the 2nd Annual International Conference on Computational Molecular Biology (RECOMB-98)*, pages 243–247, New York, USA, Mar. 1998. ACM Press.

[13] J. C. Setubal and J. Meidanis. *Introduction to Computional Molecular Biology.* PWS Publishing Company, 1997.