JOURNAL OF COMPUTATIONAL BIOLOGY
Volume 16, Number 10, 2009

Pp. 1369–1381
DOI: 10.1089/cmb.2009.0087

Research Article

A Fast and Exact Algorithm for the Median of Three Problem: A Graph Decomposition Approach

ANDREW WEI XU

ABSTRACT

In a previous article, we have shown that adequate subgraphs can be used to decompose multiple breakpoint graphs, achieving a dramatic speedup in solving the median problem. In this article, focusing on the median of three problem, we prove more important properties about adequate subgraphs with rank 3 and discuss the algorithms inventorying simple adequate subgraphs. After finding simple adequate subgraphs of small sizes, we incorporate them into ASMedian, an algorithm to solve the median of three problem. Results on simulated data show dramatic speedup so that many instances can be solved very quickly, even ones containing hundreds or thousands of genes.

Key words: algorithms, combinatorics, computational molecular biology, genomic rearrangements, statistics.

1. INTRODUCTION

THE MEDIAN PROBLEM (Bryant, 1998; Pe'er and Shamir, 1998; Sankoff and Blanchette, 1998; Moret et al., 2002; Bourque and Pevzner, 2002; Adam and Sankoff, 2008; Eriksen, 2007) for genomic rearrangement distances is NP-hard (Caprara, 2003; Tannier et al., 2008). Algorithms have been developed to find exact solutions for small instances (Caprara, 2003; Moret et al., 2002), and there are rapid heuristics of varying degrees of efficiency and accuracy (Bourque and Pevzner, 2002; Adam and Sankoff, 2008, Lenne et al., 2008). In a previous article (Xu and Sankoff, 2008) with the aim of finding a decomposition method to reduce the size of the problem, we introduced the notion of adequate subgraph and showed how they lead to such a decomposition. By applying this method recursively, the size of the problem is effectively reduced. In this article, we focus on the median of three problem, which is to find a genome q with smallest total distance $\sum_{1\leq i\leq 3} d(q, g_i)$ for any given three genomes g_1,g_2,g_3.

Because of its simple structure, we choose to work with DCJ distance (Yancopoulos et al., 2005) $d = n - c$ as most likely to yield non-trivial mathematical results, where n is the number of genes in each genome (assuming that they have the same gene content) and c is the number of cycles in the breakpoint graph. We require genomes to consist of one or more circular chromosomes, but our results could be extended to genomes with multiple linear chromosomes.

In Section 2, several related concepts are defined, such as breakpoint graph and adequate subgraph. In Section 3, some important properties about adequate subgraphs of rank 3 are proved. We discuss the

Department of Mathematics and Statistics, University of Ottawa, Ottawa, Canada. School of Computer and Communication Sciences, Swiss Federal Institute of Technology, Lausanne, Switzerland.

problem of inventorying simple adequate subgraphs in Section 4. Then in Section 5, we give an algorithm ASMedian to solve the median problem. Results on simulated data are given and discussed in Section 6.

2. GRAPHS, SUBGRAPHS, AND MORE

2.1. Breakpoint graph

We construct the breakpoint graph of two genomes by representing each gene by an ordered pair of vertices, adding colored edges to represent the adjacencies between two genes, red edges for one genome and black for the other.

In a genome, every gene has two adjacencies, one incident to each of its two endpoints, since it appears exactly once in that genome. Then in the breakpoint graph, every vertex is incident to one red edge and one black one. Thus, the breakpoint graph is a 2-regular graph which automatically decomposes into a set of alternating-color cycles (Fig. 1).

The edges of one color form a perfect matching of the breakpoint graph, which we will simply refer to as a **matching**, unless otherwise specified. By the red matching, we mean the matching consisting of all the red edges.

The **size** of the breakpoint graph is defined as half the number of vertices it contains, which equals to the size of its matchings and the number of gens in each genome.

2.2. Multiple breakpoint graph and median graph

The breakpoint graph extends naturally to a multiple breakpoint graph (MBG)(Caprara, 2003), representing a set $\mathcal{G}$ of three or more genomes, (Fig. 2). The number of genomes[1] $N_{\mathcal{G}} \geq 3$ in $\mathcal{G}$ is called the **rank** of the MBG, which is also its edge chromatic number. The colors assigned to the genomes are labeled by the integers from 1 to $N_{\mathcal{G}}$. The **size** of an MBG or its subgraph is also defined as half the number of vertices it contains.

For a candidate median genome, we use a different color for its matching E, namely color 0. Adding E to the MBG results in the **median graph**. The set of all possible candidate matchings is denoted by $\mathcal{E}$.

The 0-i **cycles** in a median graph with matching E, numbering $c(0, i)$ in all, are the cycles where 0-edges and i edges alternate. Let $c_E = \sum_{1 \leq i \leq 3} c(0, i)$. Then $c_{\max} = \max\{c_E : E \in \mathcal{E}\}$ is the maximum number of cycles that can be formed from the MBG. **Minimizing the total DCJ distance in the median problem is equivalent to finding an optimal 0-matching E, i.e., with $c_E = c_{\max}$.**

2.3. Subgraphs

Let $\mathbf{V}(G)$ and $\mathbf{E}(G)$ be the sets of vertices and edges of a regular graph G. A **proper subgraph** H of G is one where $\mathbf{V}(H) = \mathbf{V}(G)$ and $\mathbf{E}(H) = \mathbf{E}(G)$ do not both hold. An **induced subgraph** H of G is the subgraph which satisfies the property that if $x, y \in \mathbf{V}(H)$ and $(x, y) \in \mathbf{E}(G)$, then $(x, y) \in \mathbf{E}(H)$.

In this article, we focus on the induced proper subgraphs of MBGs, with even numbers of vertices. Through this article, the size of a subgraph is denoted by m. For a proper induced subgraph H, $\mathcal{E}(H)$ is the set of all its perfect 0-matchings $E(H)$. The number of cycles determined by H and $E(H)$ is $c_{E(H)}(H)$, and

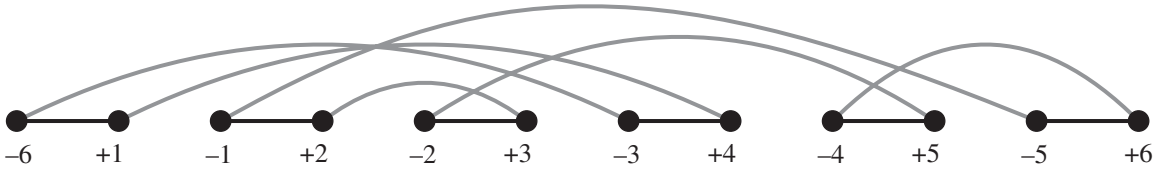

FIG. 1. Breakpoint graph for blue genome 1 −5 −2 3 −6 −4 and red genome 1 2 3 4 5 6.

[1]For the median of three problem, this number is just 3.

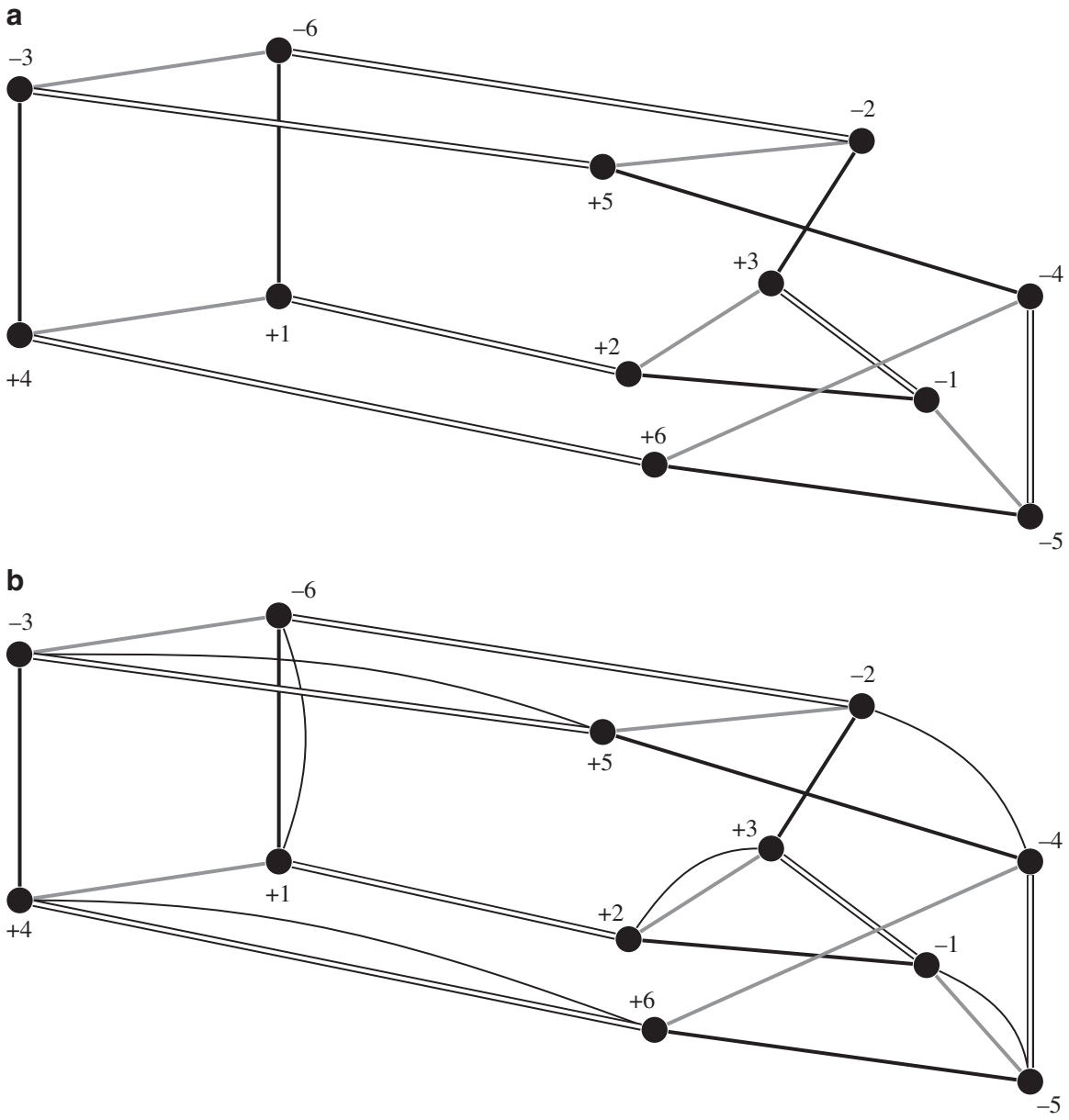

FIG. 2. MBG and median graph. Thick, gray, double, and thin edges denote the edges with colors 1, 2, 3, and 0 correspondingly. (**a**) An MBG based on three genomes: (1 2 3 4 5 6), (1 −5 −2 3 −6 −4), and (1 3 5 −4 6 −2). (**b**) A median graph with the set of thin edges denoting the adjacency edges of a candidate of the median genome.

$c_{\max}(H)$ is the maximum number of cycles that can be formed from H. A 0-matching E^* (H) with $c_{E^*(H)}(H) = c_{\max}(H)$ is called an optimal partial 0-matching, and $\mathcal{E}^*(H)$ is the set of such 0-matchings.

2.4. Non-crossing 0-matchings and decomposers

For a subgraph H of an MBG G, a potential 0-edge would be ***H*-crossing** if it connected a vertex in $\mathbf{V}(H)$ to a vertex in $\mathbf{V}(G) - \mathbf{V}(H)$. A candidate matching containing one or more H-crossing 0-edges is an H-crossing.

An MBG subgraph H is called a **decomposer** if for any MBG containing it, there is an optimal matching that is not H-crossing. It is a **strong decomposer** if for any MBG containing it, all the optimal matchings are not H-crossing.

2.5. Adequate and strongly adequate subgraphs

A connected subgraph H of size m in any MBG is an **adequate subgraph** if $c_{\max}(H) > \frac{1}{2}mN_{\mathcal{G}}$; it is **strongly adequate** if $c_{\max}(H) > \frac{1}{2}mN_{\mathcal{G}}$. For the median of three problem, an adequate subgraph of rank 3 is a subgraph with $c_{\max}(H) \geq \frac{3m}{2}$ and a strongly adequate subgraph of rank 3 is one with $c_{\max}(H) > \frac{3m}{2}$.

A (strongly) adequate subgraph H is **simple** if it does not contain another (strongly) adequate subgraph as an induced subgraph; deleting any vertex from H will destroy its adequacy.

Adequate subgraphs enable us to decompose the MBG into a set of smaller ones, as in the next theorem.

Theorem 1. *(Xu and Sankoff, 2008). Any adequate subgraph is a decomposer. Any strongly adequate subgraph is a strong decomposer.*

3. THE PROPERTIES OF SIMPLE ADEQUATE SUBGRAPHS OF RANK 3

In this section, we prove other important properties about simple adequate subgraphs of rank 3. Multiple edges (together with their end vertices) in MBGs are the simple adequate subgraphs of size one, which are the only exceptions to many of the properties stated below.

3.1. More properties about adequate subgraphs of rank 3

Lemma 1. *The vertices of simple adequate subgraphs of rank 3 have degrees either 2 or 3.*

Proof. Since the MBG for the median of three problem is 3-regular, the vertex degrees of its induced subgraphs can only be 1, 2, or 3.

The lemma is true for parallel edges (the smallest simple adequate subgraphs), where the vertex degrees are 2 or 3. For simple adequate subgraphs of size two or more, we prove by contradiction that they can not contain vertices of degree 1.

Assume there is a simple adequate subgraph H of size m containing a vertex x of degree 1. In one of the optimal 0-matchings of H, x is connected to vertex y by a 0-edge e, and e appears only in one of the color-alternating cycles. By deleting edge e and vertices x, y, only that cycle is destroyed. Because of its adequacy, the maximum number of cycles formed with H is at least $\frac{3m}{2}$. So for the resultant subgraph F of size $m-1$, the maximum number of cycles can be formed is at least $\frac{3m}{2} - 1 = \frac{3(m-1)}{2} + \frac{1}{2} > \frac{3(m-1)}{2}$. Therefore, F, as a subgraph of H, is also an adequate subgraph, which contradicts the assumption that H is simple.

So the vertex degrees in a simple adequate subgraph can only be 2 or 3. ■

Lemma 2. *Except for the adequate subgraphs consisting of multiple edges, the size of a simple adequate subgraph of rank 3 is even.*

Proof. Suppose there is an odd-sized simple adequate subgraph H of size $2k+1$. Because of its adequacy, the maximum number of cycles formed with H is at least $\left\lceil \frac{3(2k+1)}{2} \right\rceil = 3k+2$. Since H is an proper subgraph, there exists a vertex x with degree 2. Suppose 0-edge e is incident to x in one of H's optimal 0-matchings. By deleting e and the corresponding vertices, two color-alternating cycles are destroyed. Then for the resultant subgraph F of size $2k$, the maximum number of cycles formed with F is at least $3k = \frac{3}{2} \times 2k$. Hence, F, as a subgraph of H, is also an adequate subgraph, which contradicts the simplicity of H. ■

Lemma 3. *Except for the adequate subgraphs consisting of multiple edges, the maximum number of cycles of a simple adequate subgraph of rank 3 is exactly $\frac{3m}{2}$, where m is its size.*

Proof. Because of Lemma 2, we only need to consider even-sized simple adequate subgraphs. Suppose H is a simple adequate subgraph of size $2k$, with which the maximum number of cycles formed is at least $3k+1$. Then by deleting a 0-edge connecting to a degree 2 vertex, the size of the subgraph decreases by 1 and the number of cycles decreases by 2. So H contains another adequate subgraph of size $2k-1$ whose

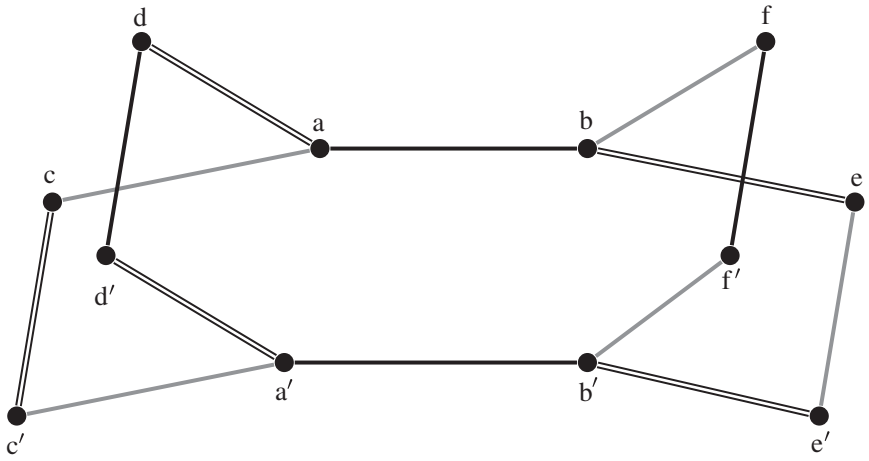

FIG. 3. Illustration of a mirrored-tree graph. Two identical edge-colored binary trees on (a, b, c, d, e, f) and (a', b', c', d', e', f') are connected through the corresponding leave vertices (c, c'), (d, d'), (e, e') and (f, f') to form a mirrored-tree graph.

maximum number of cycles is at least $3k - 1 = \lceil \frac{3}{2}(2k-1) \rceil$, which contradicts the simplicity assumption for H. ■

3.2. There are infinitely many simple adequate subgraphs

In this subsection, we show that there are infinitely many adequate subgraphs, by proving the number of simple adequate subgraphs is infinite, which follows from the infinite size of a special family of simple adequate subgraphs: the mirrored-tree graph (Fig. 3).

Definition 1. ***An mirrored-tree graph****: two identical 3-edge-colored full binary trees*[2] *with corresponding pairs of leaf vertices connected by simple edges (Fig. 3). Being an MBG subgraph, the size of an mirrored-tree graph is defined as half the number of its vertices, which also is the number of vertices contained in each tree.*

Proposition 1.

1. *Any full binary tree containing more than one vertex must have even size;*
2. *for a full binary tree with m vertices, there are* $\frac{m}{2} + 1$ *leaf vertices (with degree 1) and* $\frac{m}{2} - 1$ *inner vertices (with degree 3).*
3. *the total number of edges is* $m - 1$.

Proposition 2. *For a mirrored-tree graph of size m, there are* $\frac{5m}{2} - 1$ *edges in total;* $\frac{m}{2} + 1$ *of them connect the two binary trees and* $2m - 2$ *of them lie in the trees.*

Definition 2. ***Double-Y end****: A mirrored-tree graph of size 4, with one connecting edge missing, as illustrated by Figure 4a. Being a part of an MBG subgraph, it is connected to the remaining graph through the two vertices of degree one.*

Lemma 4. *If a double-Y end appears in an MBG subgraph H, then the 0-edges (a, b), (c, d) and (e, f) connecting corresponding vertices of the two identical trees, must exist in any optimal 0-matching of H, as illustrated by Figure 4c.*

[2]Where vertices degrees can only be 1 or 3.

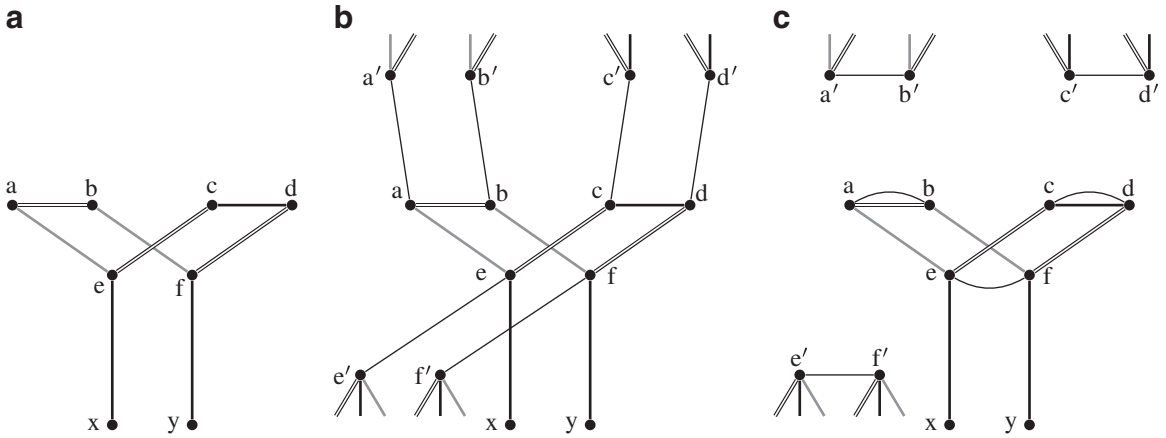

FIG. 4. (**a**) Illustration of a double-Y end and it is connected to the remaining subgraph only through vertices x and y. (**b**) 0-matching not containing 0-edges (a, b),(c, d),(e, f). (**c**) Another 0-matching obtained by applying 3 DCJ operations to the 0-matching in (b), that does contain those three 0-edges and forms more color-alternating cycles.

Proof. In an optimal 0-matching of H, if any of the three 0-edges (a, b), (c, d) and (e, f) appears, the other two 0-edges must also exist. Then only one case is left to disprove–-that none of these 0-edges appears in some optimal 0-matching, as illustrated by Figure 4b.

Figure 4c is obtained by three DCJ operations on the 0-edges of Figure 4b, creating 0-edges (a, b), (c, d) and (e, f). By comparison we can see that: a′, b′ are connected by a double/thin line pattern alternating path and c′, d′ are connected by a thick/thin line pattern alternating path in both figures. So they are involved in the same number of cycles in both figures.

Apart from these, Figure 4b contains another 6 paths, which can form at most 6 color-alternating cycles; Figure 4c contains 4 cycles as well as another 6 paths of 3 different colors which will form at least 3 cycles, summing up to a total of 7 cycles or more. So Figure 4c forms more cycles than Figure 4b.

For the cases where vertices a', b', c', d', e', f' are incident to different set of edges, the same result still holds.

Since 0-matchings of H containing 0-edges (a, b), (c, d), and (e, f) have more cycles than 0-matchings not containing them, these three 0-edges must exist in any optimal 0-matchings. ■

Theorem 2. *With a mirrored-tree graph of size m, we can form a maximum of* $\frac{3m}{2}$ *color alternating cycles, hence it is an adequate subgraph; Furthermore, it does not contain any smaller adequate subgraphs, so it is a simple adequate subgraph.*

Proof. We first prove that there is a 0-matching of the mirrored-tree graph, forming $\frac{3m}{2}$ color alternating cycles. This is just the set of 0-edges connecting the corresponding vertices of the two trees. With this 0-matching, each non-0 edge connecting two trees makes a cycle by itself; and the edges on the tree form cycles of size 2 with the corresponding edges on the other tree. From Proposition 2, there are $\frac{m}{2} + 1$ edges connecting trees and $2m - 2$ edges on the trees, the total number of cycles is $\frac{3m}{2}$.

Next we show that is the only optimal 0-matching. For any binary tree, since the number of leaf vertices is larger than the number of inner vertices by 2, there is always an inner vertex being connected to two leaf vertices. In the corresponding mirrored-tree graph, this gives a double-Y end.

For a mirrored-tree graph H, we add two 0-edges parallel to the connecting edges of its double-Y end as 0-edges (a, b) and (c, d) in Figure 4c. Then by shrinking them, H becomes a quasi mirrored-tree graph[3] of smaller size, containing double-Y ends or quasi double-Y ends.[4] By applying this procedure of adding and

[3]In which, there may be multiple edges connecting the two identical trees.

[4]The ones whose connecting edges might be multiple edges. Obviously the conclusion in Lemma 4 also applies to quasi double-Y ends.

shrinking 0-edges to the (quasi) double-Y ends recursively, H finally becomes a three-parallel edge. Since in each step the new added 0-edges must appear in all optimal 0-matchings, the resultant perfect 0-matching is the only optimal 0-matching of H.

The symmetrical structure of mirrored-tree graphs leads to color-alternating cycles of smallest sizes–-1 and 2 only. In detecting whether a mirrored-tree graph H contains any smaller adequate subgraphs, it is sufficient to only consider its subgraphs with symmetrical structures. Using reasoning similar to the above paragraphs, it can be shown that the optimal 0-matchings for these symmetrical subgraphs of H are the subsets of the 0-edges in the optimal 0-matching of H. However none of these symmetrical subgraphs of H can form cycles of $\frac{3}{2}$ times their sizes. So the mirrored-tree graphs are simple adequate subgraphs. ■

Theorem 3. *There are infinitely many simple adequate subgraphs.*

Proof. Since there are full binary trees of arbitrary large size, which give mirrored-tree graphs with arbitrary large (even) size. Also because mirrored-tree graphs are simple adequate subgraphs, there exist simple adequate subgraphs with arbitrary large (even) size. ■

4. INVENTORYING SIMPLE ADEQUATE SUBGRAPHS

4.1. It is practical to use simple adequate subgraphs of small sizes

Before using the adequate subgraphs to reduce the search space for finding an optimal 0-matching, we need to inventory the adequate subgraphs. Theorem 3 states that there are infinitely many simple adequate subgraphs, hence infinitely many adequate subgraphs, so it is impossible to inventory all of them and use them to decompose the median problems. However, it is practical to work on simple adequate subgraphs of small sizes, as justified by the following:

1. There are much fewer simple adequate subgraphs. And many non-simple adequate subgraphs can be decomposed into several simple adequate subgraphs embedded in each other. Hence many non-simple ones can be detected through the constituent simple ones.
2. The algorithms to inventory simple adequate subgraphs for a given size require more than exponential time in their size.
3. The total number of simple adequate subgraphs increases dramatically as the size increases. The complexity of the algorithm to detect the existence of a given simple adequate subgraph also increases accordingly. Combining these two factors, we conclude that it is prohibitively expensive to detect the existence of simple adequate subgraphs of large sizes.
4. Simple adequate subgraphs of small sizes exist with much higher probability than subgraphs of greater size on random MBGs.

4.2. Algorithms to inventory simple adequate subgraphs

To enumerate the simple adequate subgraphs, we need to search among all the MBG subgraphs, which consist of (perfect or non-perfect) matchings of three colors. In order to count the number of cycles, the perfect 0-matchings must also be enumerated. So the algorithms need to work on graphs consisting of 4 matchings, hence the problem is computationally costly.

Our simple adequate subgraph inventorying algorithm uses a depth-first branch-and-bounce method in enumerating all possible subgraphs for a given size. The graph grows by adding an edge at each step. It is backtracked whenever the current graph contains a smaller simple adequate subgraph and then restrained on another path to grow the graph until all subgraphs have been searched.

To speed up the algorithm, we adopt several useful methods and techniques:

1. Only inventory simple adequate subgraphs of even sizes, as a result of Lemma 2.
2. Fix the 0-matching. Any median subgraph is isomorphic to $\frac{(2m)!}{2^m m!} - 1$ other median subgraphs by permuting the 0-edges.
3. Only allow the graphs whose number of 1-edges is no less than the number of 2-edges and the number of 2-edges is no less than the number of 3-edges, because of the isomorphism associated with the permutation of colors.
4. Every vertex must be incident to 2 or 3 non-0-edges, because of Lemma 1.

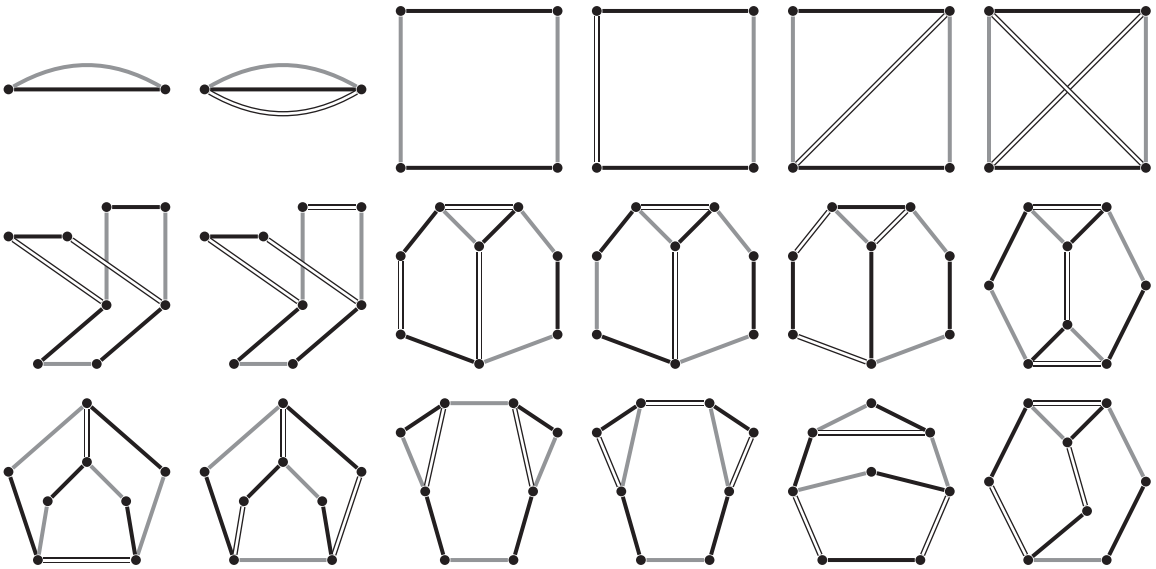

FIG. 5. Simple adequate subgraphs of size 1, 2, and 4 for MBGs on three genomes.

4.3. Simple adequate subgraph enumerated

In Figure 5, the simple adequate subgraphs of size 1, 2, and 4 are listed. Each subgraph represent a class of subgraphs isomorphic under the permutations of vertices and colors.

5. SOLVING THE MEDIAN OF THREE PROBLEM BY RECURSIVELY DETECTING SIMPLE ADEQUATE SUBGRAPHS

Our algorithm using adequate subgraphs to decompose the median problems is called **ASMedian**. It adopts a branch-and-bound method to find an optimal 0-matching for any given MBG. At any intermediate step during the branch-and-bound search, an intermediate configuration (IC for short) is constructed,

Algorithm 1 ASMedian

Input: three genomes containing any number of circular chromosomes
Output: the median genome and the maximum number of cycles c
construct the MBG, assign its upper bound u to U and its lower bound to c^* and push it into the unexamined list $\mathcal{L}$;
while $U > c^*$ **and** $\mathcal{L}$ **is not empty do**
  pop out an IC with $u = U$ from $\mathcal{L}$;
  if an adequate subgraph H **is found in the iMBG of that IC then**
    set the major set as the one of H;
  else
    select the vertex with smallest label and set the major set as the set containing all 0-edges incident to that vertex;
  generate a set of new ICs with their partial 0-matchings are expanded to include a 0-matching in the major set and their iMBGs as the resultant graphs of shrinking these partial 0-matchings;
  update U and c^*;
  if c^* **gets updated then** Remove all the ICs with $u \leq c^*$ in $\mathcal{L}$;
  push the new generated ICs with $u > c^*$ into $\mathcal{L}$;
// the maximum cycle number has been found;
set c as c^* and construct the median genome from the optimal 0-matching obtained;
return c and the median genome;

containing a partial 0-matching and an intermediate MBG (iMBG for short) resulted by a process of edge-shrinking (Xu and Sankoff, 2008) of that partial 0-matching from the original MBG. The algorithm keeps a list of unexamined ICs $\mathcal{L}$, initially just consisting of the original MBG.

At each step, from $\mathcal{L}$ an unexamined IC with the largest upper bound is selected to examine. According to whether an inventoried simple adequate subgraph exists in that iMBG and what simple adequate subgraph it is, a number of new ICs are generated, containing smaller and non-empty iMBGs and expanded partial 0-matchings. Then we update U the largest upper bound of all unexamined ICs and c^* the largest cycle number encountered so far. We prune the ICs whose upper bounds are no larger than c^*. The algorithm stops when $c^* \geq U$ or no unexamined ICs remain. Then c^* is the maximum cycle number for the original MBG, and the corresponding 0-matching is an optimal 0-matching.

5.1. Examining the intermediate MBGs

Definition 3. ***The inquiry set*** *is the set of simple adequate subgraphs (of small sizes) for which the ASMedian algorithm looks on the intermediate MBGs. For a specific algorithm, the inquiry set is given as a parameter.*

The iMBG of the selected IC is examined to see the existence of any simple adequate subgraph in the inquiry set. If one of such subgraphs H exists, then we know there is an optimal 0-matchings of iMBG which is non-H-crossing. This 0-matching can be divided into two parts: a 0-matching of H and a partial 0-matching of the remaining intermediate MBG.

A **major set** of H is the minimal set of 0-matchings of H, which guarantees that at least one of them must appear in an optimal 0-matchings of the MBG, without the knowledge of the remaining part of the MBG (as will be shown else where). The size of the major set is denoted by μ. Since the inquiry set is given in advance, the major sets for the simple adequate subgraphs are also known in advance. Then according to this major set, μ new ICs will be generated with smaller iMBGs, each resulting from the shrinking of a 0-matching of H in the major set from the iMBG of the currently selected IC.

When no simple adequate subgraph in the inquiry set exists, the vertex v with the smallest remaining label is selected. The nominal major set of size $2\tilde{n} - 1$ is constructed, where $\tilde{n}$ is the size of the iMBG--just the set of 0-edges incident to v (here each partial 0-matching is just a 0-edge). Then $2\tilde{n} - 1$ new ICs are created accordingly.

If the inquiry set is chosen as all simple adequate subgraphs of sizes 1, 2 and 4, then the sizes of their major sets are just one, except for one case where it is 2. It can be seen that whenever a simple adequate subgraph is detected, the search space is roughly reduced by a factor of $(2\tilde{n})^2$, $(2\tilde{n})^4$, or $(2\tilde{n})^8$.

5.2. The lower bound and the upper bound

For each intermediate configuration, the ASMedian algorithm calculates its upper bound and prunes it if the value is no larger than c^*—the maximum number of cycles encountered so far.

Because of the search schema we use (see next subsection), it takes a while for the algorithm to reach any perfect 0-matching. Due to the fact that the number of cycles formed by partial 0-matchings are small, to calculate c^* from them will make the pruning procedure very inefficient. Instead, for each intermediate configuration, a tight lower bound is calculated and c^* takes the maximum of these lower bounds and the encountered cycle numbers.

Since the DCJ distance is a metric measure, for any median of three problem, there is an associated lower bound for the total distance. Assume the three known genomes are labeled as 1, 2, 3, and $d_{1,2}$, $d_{1,3}$, $d_{2,3}$ denote the pairwise distances and $c_{1,2}$, $c_{1,3}$, $c_{2,3}$ denote the cycle numbers between any two pairs. The lower bound for the total distance is $d \geq \frac{d_{1,2}+d_{1,3}+d_{2,3}}{2}$. Because $d_{i,j} = n - c_{i,j}$, then we get an upper bound for the total cycle number,

$$c \leq \frac{3n}{2} + \frac{c_{1,2}+c_{1,3}+c_{2,3}}{2}. \tag{1}$$

To find a lower bound for the total cycle number, we can set the 0-matching to any of the matchings representing the three known genomes and take largest total cycle number of the three as the lower bound, so that

$$c \geq c_{1,2} + c_{1,3} + c_{2,3} - \min\{c_{1,2}, c_{1,3}, c_{2,3}\}. \quad (2)$$

For any IC, by adding $\tilde{c}$, the number of cycles formed by its partial 0-matching, to the lower bound and upper bound of its intermediate MBG, we get the upper bound and lower bound of this IC, denoted by u and l correspondingly.

$$u = \tilde{c} + \frac{3\tilde{n}}{2} + \frac{\tilde{c}_{1,2} + \tilde{c}_{1,3} + \tilde{c}_{2,3}}{2} \quad (3)$$

$$l = \tilde{c} + \tilde{c}_{1,2} + \tilde{c}_{1,3} + \tilde{c}_{2,3} - \min\{\tilde{c}_{1,2}, \tilde{c}_{1,3}, \tilde{c}_{2,3}\}. \quad (4)$$

The IC and all ICs derived from it, are referred as the **parent** IC and the **child** ICs. A non-increasing property holds between the upper bounds of the parent IC and the child ICs.

Lemma 5. *The upper bounds of the child ICs are never larger than the upper bound of their parent IC.*

Proof. Suppose a child IC is obtained from the parent IC by adding a 0-edge e. We first inspect the possible effects on $\tilde{c}$ and $\tilde{c}_{1,2}$ of adding e to the iMBG of the parent IC.

a. If e connects two 1-2 cycles, then the two cycles will be merged into one. Then $\tilde{c}$ remains the same and $\tilde{c}_{1,2}$ decreases by 1;
b. if e parallels a 1-edge (or a 2-edge), then one 0-1 cycle of size 1 is formed and the 1-2 cycle containing that edge becomes a shorter one. So that $\tilde{c}$ increases by 1 and $\tilde{c}_{1,2}$ remains the same;
c. if e connects two vertices of the same 1-2 cycle, not paralleling any edges, then this 1-2 cycle may be split into two or remain with a smaller size. Therefore $\tilde{c}$ remains the same and $\tilde{c}_{1,2}$ increases by 0 or 1.

Since the size of the iMBG in the child IC decreases by 1, $\frac{3\tilde{n}}{2}$ decreases by $\frac{3}{2}$. As long as $\tilde{c} + \frac{\tilde{c}_{1,2} + \tilde{c}_{1,3} + \tilde{c}_{2,3}}{2}$ does not increase more than $\frac{3}{2}$, u never increases.

1. If e does not parallel any edge, then $\tilde{c}$ remains the same, and each of $\tilde{c}_{1,2}, \tilde{c}_{1,3}, \tilde{c}_{2,3}$ increases at most by 1. So u does not increase;
2. if e parallels one edge, $\tilde{c}$ increases by 1 and only one of $\tilde{c}_{1,2}, \tilde{c}_{1,3}, \tilde{c}_{2,3}$ increases at most by 1. So u does not increase;
3. if e parallels two edges, $\tilde{c}$ increases by 2 and the cycle formed by the parallel edges is destroyed and the other two terms of $\tilde{c}_{1,2}, \tilde{c}_{1,3}, \tilde{c}_{2,3}$ remain the same. So u does not change;
4. e parallels three edges, $\tilde{c}$ increases by 3 and the three cycles formed by the parallel edges are destroyed. So u remains the same.

So the upper bound of the child ICs are never larger than the upper bound of their parent IC. ■

The algorithm maintains an overall upper bound U which is the maximum upper bound of all unexamined ICs. Another global variable, as mentioned before, is the largest total cycle number or lower bound c^* found so far. Obviously, the maximum total cycle number c of the original MBG lies between c^* and U.

5.3. The optimistic search schema

Our algorithm is neither a strict depth-first nor a strict breadth-first search schema, but follows an "optimistic" search strategy. From the list of all unexamined ICs, we select the one with the largest upper bound. The intuition behind this is, the ICs with larger upper bounds are more likely to lead to perfect 0-matchings with larger cycle numbers. Beside the intuitive aspect, we can prove that this optimistic search schema has a smallest search space in terms of the number of ICs it examines.

Theorem 4. *The set of ICs the optimistic search schema examines includes all ICs with $u > c$, plus a subset of ICs with $u = c$. Furthermore, since the search space of every branch-and-bound method includes all ICs with $u > c$, the optimistic search schema has the smallest search space possible.*

Proof. Obviously, every IC with $u > c$ should be examined by the algorithm; otherwise, the possibility of having a maximum total cycle number with $c + 1$ or more cannot be eliminated.
Because of Lemma 5, for any IC with $u \geq c$, all the ICs lying on the path from the original of the search to this IC have their upper bounds larger than or equal to c. So the algorithm with optimistic search schema

never needs to examine any IC with $u < c$ to find the ones with $u \geq c$, i.e., this algorithm finds all ICs with $u \geq c$ without examining any ones with smaller upper bounds. By the time that the ones with $u \geq c$ have been examined, an optimal 0-matching with c cycles has been found and the algorithm stops. And the search space for the optimistic schema includes all the ICs with $u > c$ and a subset of the ICs with $u = c$. Hence the optimistic search schema has the smallest search space possible. ■

The exact algorithm in Caprara (2003) consists of cascaded runs of depth-first branch-and-bound search, with the first run seeking a solution whose cycle number is equal to the upper bound of the original MBG and the subsequent runs seeking solutions with one cycle less than the previous ones, until a solution is found. The cascaded branch-band-bound algorithm and our optimistic branch-and-bound algorithm are similar in terms of the search spaces. The intermediate configurations may be examined more than once in the former algorithm. In our optimistic algorithm, some intermediate configurations with smaller upper bounds need to be stored temporarily. Although storing huge amount of these intermediate configurations can be a challenge to physical memories or even hard disks, the problem is dramatically improved with the adequate subgraph decomposition method and it can be further improved by finding better pruning methods, such as finding a better lower bound or running a heuristic before the main exact algorithm starts.

6. RESULTS ON SIMULATED DATA

ASMedian algorithm is implemented in Java and runs on a MacBook, using only one 2.16-GHz CPU. Sets of data are simulated with varying parameters n and $\pi = \rho/n$, where n is the number of gene in each genome and ρ is the number of random reversals applied to the ancestor $I = 1, \cdots, n$ independently to derive each of the three different genomes. n ranges among 10, 20, 30, 40, 50, 60, 80, 100, 200, 300, 500, 1000, 2000, 5000 and π starts from 0.1 and increases by intervals of 0.1. For each data set, 10 instances are generated.

6.1. The running time for simulated data sets with varying n and $\pi = \rho/n$

Table 1 shows the average running time in seconds for all data sets whose 10 instances can all be solved within one hour or the number of solved instances in parenthesis for the remaining data sets. It can be seen that relatively large instances can be solved if ρ/n remains at 0.3 or less. It also shows that for small n, the median is easy to find even if ρ/n is large enough to effectively scramble the genomes.

6.2. The effect of adequate subgraph discovery on speed-up

Table 2 shows how the occurrence of adequate subgraphs of 2 and 4 can dramatically speed up the solution to the median problem, generally from more than half an hour to a fraction of a second. The algorithm using no adequate subgraphs runs too slowly to terminate within a practical time.

Table 1. For Each Data Set, If its Ten Instances All Finish in 1 Hour, Then Their Average Running Time Is Shown in Seconds; Otherwise, the Number of Finished Instances Is Shown with Parentheses

n	ρ/n	*0.1*	*0.2*	*0.3*	*0.4*	*0.5*	*0.6*	*0.7*	*0.8*	*0.9*	*1.0*
10		$4\,10^{-4}$	$1\,10^{-4}$	$2\,10^{-4}$	$8\,10^{-4}$	$4\,10^{-4}$	$2\,10^{-3}$	$2\,10^{-3}$	$8\,10^{-4}$	$4\,10^{-4}$	$5\,10^{-4}$
20		$2\,10^{-4}$	$2\,10^{-4}$	$3\,10^{-4}$	$6\,10^{-4}$	$9\,10^{-4}$	$6\,10^{-4}$	$2\,10^{-2}$	$7\,10^{-3}$	$2\,10^{-2}$	$5\,10^{-3}$
30		$2\,10^{-3}$	$2\,10^{-3}$	$3\,10^{-4}$	$7\,10^{-4}$	$5\,10^{-3}$	$3\,10^{-2}$	$5\,10^{-2}$	$1\,10^{-1}$	$4\,10^{-1}$	1
40		$1\,10^{-4}$	$2\,10^{-4}$	$3\,10^{-4}$	$6\,10^{-4}$	$4\,10^{-2}$	1	6	$6\,10^{1}$	$6\,10^{1}$	$5\,10^{1}$
50		0	$4\,10^{-4}$	$5\,10^{-4}$	$2\,10^{-3}$	$7\,10^{-2}$	$7\,10^{1}$	(9)	(7)	(7)	
60		$2\,10^{-3}$	$1\,10^{-3}$	$5\,10^{-3}$	$3\,10^{-2}$	$5\,10^{1}$	(7)				
80		$3\,10^{-4}$	$4\,10^{-4}$	$7\,10^{-4}$	$8\,10^{-2}$	(9)					
100		$3\,10^{-4}$	$7\,10^{-4}$	$1\,10^{-3}$	$7\,10^{1}$	(1)					
200		$7\,10^{-3}$	$1\,10^{-2}$	$3\,10^{-2}$	(0)						
300		$5\,10^{-3}$	$5\,10^{-3}$	$2\,10^{-2}$	(0)						
500		$2\,10^{-2}$	$2\,10^{-2}$	$1\,10^{-1}$	(0)						
1000		$9\,10^{-2}$	$7\,10^{-2}$	$8\,10^{1}$	(0)						
2000		$9\,10^{-2}$	$3\,10^{-1}$	(3)							
5000		2	2	(0)							

Table 2. Speedup Due to Discoveries of Adequate Subgraphs of Size 2 and 4

		Run time		*Number of edges*			
Run	*Speedup factor*	*With AS of sizes 1, 2, 4*	*With AS of size 1*	*AS1*	*AS2*	*AS4*	*AS0*
1	41,407	4.5×10^{-2}	1.9×10^{3}	53	39	8	0
2	85,702	3.0×10^{-2}	2.9×10^{3}	53	34	12	1
3	2,542	5.4×10^{0}	1.4×10^{4}	56	26	16	2
4	16,588	3.9×10^{-2}	6.5×10^{2}	58	42	0	0
5	$>10^{6}$	5.9×10^{2}	stopped	52	41	4	3
6	199,076	6.0×10^{-3}	1.2×10^{3}	56	44	0	0
7	6,991	2.9×10^{-1}	2.1×10^{3}	54	33	12	1
8	$>10^{6}$	4.2×10^{1}	stopped	57	38	0	5
9	1,734	8.7×10^{0}	1.5×10^{4}	65	22	8	5
10	855	2.1×10^{0}	1.8×10^{3}	52	38	8	2

Three genomes are generated from the identity genome with $n = 100$ by 40 random reversals. Time is measured in seconds. Runs were halted after 10 hours. AS1, AS2, AS4, AS0 are the numbers of edges in the solution median constructed consequent to the detection of adequate subgraphs of sizes 1, 2, 4 and at steps where no adequate subgraphs were found, respectively.

Among the last four columns in Table 2, AS1, AS2, AS4 denote the total numbers of 0-edges constructed due to discovering adequate subgraphs of size 1, 2 and 4; AS0 denotes the number of 0-edges constructed at the stages where no small adequate subgraphs are discovered. Since at each stage where no small adequate subgraphs are discovered, the algorithm searches at most $2n$ intermediate MBGs, while AS0 ranges from 0 to 5, this suggests that the solution space searched by the ASMedian algorithm ranges from 1 to 10^6, which is a dramatic reduction from $(2\times100-1)!! \sim 10^{187}$—i.e., the space of all possible solutions when genomes contain 100 genes.

7. CONCLUSION

In this article, several important properties about the adequate subgraphs of rank 3 are proved. We show that there are infinitely many adequate subgraphs, hence it is not possible to list all these subgraphs. By showing that the simple adequate subgraphs of small sizes have the largest occurrence probability on random MBGs and the algorithms of detecting them are simple and fast, it is practical and efficient to solve the median of three problem by only using simple adequate subgraphs of small sizes. This is confirmed by the dramatic speedup shown in the results on simulated data. Whether it is worth exploring simple adequate subgraphs of size 6 is not clear. It depends on many factors, such as the size of the problem (number of genes genomes contained) and the algorithms for detecting subgraphs and their implementations.

DISCLOSURE STATEMENT

No competing financial interests exist.

REFERENCES

Adam, Z., and Sankoff, D. 2008. The abcs of mgr with dcj. *Evol. Bioinform.* 4, 69–74.

Bourque, G., and Pevzner, P.A. 2002. Genome-scale evolution: reconstructing gene orders in the ancestral species. *Genome Res.* 12.

Bryant, D. 1998. The complexity of the breakpoint median problem [Technical Report CRM-2579]. Centre de recherches mathématiques, Université de Montréal.

Caprara, A. 2003. The reversal median problem. *INFORMS J. Comput.* 15, 93–113.

Eriksen, N. 2007. Reversal and transposition medians. *Theor. Comput. Sci.* 374, 111–126.

Lenne, R., Solnon, C., Stützle, T., et al. 2008. Reactive stochastic local search algorithms for the genomic median problem. *LNCS* 4972, 266–276.

Moret, B.M.E., Siepel, A.C., Tang, J., et al. 2002. Inversion medians outperform breakpoint medians in phylogeny reconstruction from gene-order data. *LNCS*, 2452.

Pe'er, I., and Shamir, R. 1998. The median problems for breakpoints are np-complete. *ECCC*.

Sankoff, D., and Blanchette, M. 1998. Multiple genome rearrangement and breakpoint phylogeny. *J. Comput. Biol.* 5, 555–570.

Tannier, E., Zheng, C., and Sankoff, D. 2008. Multichromosomal genome median and halving problems. *LNCS* 5251, 1–13.

Xu, A.W., and Sankoff, D. 2008. Decompositions of multiple breakpoint graphs and rapid exact solutions to the median problem. *LNBI* 5251.

Yancopoulos, S., Attie, O., and Friedberg, R. 2005. Efficient sorting of genomic permutations by translocation, inversion and block interchange. *Bioinformatics* 21, 3340–3346.

Address correspondence to:
Dr. Andrew Wei Xu
School of Computer and Communication Sciences
Swiss Federal Institute of Technology (EPFL)
EPFL IC LCBB INJ 231
Station 14
CH-1015 Lausanne, Switzerland

E-mail: wei.xu@epfl.ch

This article has been cited by:

1. Caroline Anne Larlee, Alex Brandts, David Sankoff. 2016. Compromise or optimize? The breakpoint anti-median. *BMC Bioinformatics* **17**:S18. . [CrossRef]
2. Zhaoming Yin, Jijun Tang, Stephen W. Schaeffer, David A. Bader. 2016. Exemplar or matching: modeling DCJ problems with unequal content genome data. *Journal of Combinatorial Optimization* **32**:4, 1165-1181. [CrossRef]
3. Joao Paulo Pereira Zanetti, Priscila Biller, Joao Meidanis. 2016. Median Approximations for Genomes Modeled as Matrices. *Bulletin of Mathematical Biology* **78**:4, 786-814. [CrossRef]
4. Nguyen Ngan, Hickey Glenn, Zerbino Daniel R., Raney Brian, Earl Dent, Armstrong Joel, Kent W. James, Haussler David, Paten Benedict. 2015. Building a Pan-Genome Reference for a Population. *Journal of Computational Biology* **22**:5, 387-401. [Abstract] [Full Text HTML] [Full Text PDF] [Full Text PDF with Links] [Supplemental Material]
5. Jun Zhou, Fei Hu, William Hoskins, Jijun TangAssessing ancestral genome reconstruction methods by resampling 25-31. [CrossRef]
6. Zhaoming Yin, Jijun Tang, Stephen W. Schaeffer, David A. Bader. 2013. Streaming Breakpoint Graph Analytics for Accelerating and Parallelizing the Computation of DCJ Median of Three Genomes. *Procedia Computer Science* **18**, 561-570. [CrossRef]
7. Seunghwa Kang, Jijun Tang, Stephen W. Schaeffer, David A. Bader. 2011. Rec-DCM-Eigen: Reconstructing a Less Parsimonious but More Accurate Tree in Shorter Time. *PLoS ONE* **6**:8, e22483. [CrossRef]
8. Martin Bader. 2011. The transposition median problem is NP-complete. *Theoretical Computer Science* **412**:12-14, 1099-1110. [CrossRef]
9. Andrew Wei Xu, Bernard M. E. MoretGASTS: Parsimony Scoring under Rearrangements 351-363. [CrossRef]