Time complexity and relaxation gap for the rank median of three genomes

Victor de Moraes[1 [0009−0001−7317−7515]] and Joao Meidanis[1 [0000−0001−7878−4990]]

Institute of Computing, University of Campinas, Brazil

Abstract. In this paper, we study the genomic median of 3 problem with respect to the rank distance, which looks for a genome that minimizes the sum of the rank distances to three given genomes. We advance the knowledge on the mathematical properties of this problem, and settle its computational complexity, showing it is NP-hard. We also prove that the gap between exact and relaxed solutions of the problem can be arbitrarily large.

1 Introduction

The genome median problem is the problem of finding a genome that minimizes the sum of its distances to three other genomes, given some metric on the genomes. Many metrics have been proposed for genome comparison, with the two most relevant to this paper being the double cut and join (DCJ) distance [10] and the rank distance [11], that focus on large genome rearrangements rather than on point mutations.

Both distances belong to a line of research that aims to achieve a balance between biological significance and computational tractability. Thus, these distances compare multichromosomal genomes, contemplating both linear and cicular chromosomes, and are compatible with biologically observed rearrangements (also called operations in the computational context) such as inversions, transpositions, translocations, fusions, and fissions, among others. The weights assigned to these operations approximate well, for the most part, biological frequency measurements on real genomes [3]. On the other hand, they are efficiently computable given the two genomes.

From the start, a connection between genome rearrangement theory, permutations, and matchings was clear. In 2000, Meidanis and Dias proposed an algebraic formalism for genome rearrangement problems [7], which led to an algebraic distance for circular genomes [3]. Years later, Yancopoulos et al. [10] extended it to encompass linear chromosomes as well, creating the DCJ distance, that quickly became a favorite among researchers. Tannier et al. showed in 2009 that the DCJ median problem is NP-hard. In 2013, Feijão introduced yet another way of extending the algebraic distance to linear chromosomes [5], pointed out that it is similar to but distinct from DCJ, and conjectured that the median problem would be NP-hard for it too [4]. Zanetti et al. later replaced the permutations that represent genomes by permutation matrices, creating the rank

distance, which is just Feijao's algebraic distance in another guise and multiplied by 2 [11, Sec.2.3.2].

In summary, unlike the DCJ median problem, which is known to be NP-hard [8], the complexity of the median problem under the rank distance is not known, to the best of our knowledge. In this paper, we prove that finding genomic rank medians is NP-hard as well.

In the rank model, genomes are represented as binary, symmetric, orthogonal matrices. A relaxation of the genome median problem, where arbitrary, real-valued matrices are allowed, can be solved in polynomial time [1,6]. Although these developments may give rise to heuristics for the genomic version through, for example, maximum weight matchings, little is known about the gap between optimum scores for the genomic and relaxed medians. We show here that this gap can be arbitrarily large, prompting the search for approximation algorithms. Zanetti et al. [11] mention a trivial, 4/3-approximation given by one of the input matrices. To the best of our knowledge, no better approximation bound is known.

This article is structured as follows. In Section 2, we formally define the rank median problem. In Section 3, we introduce some properties of this problem, with the most notable ones being a composition rule for medians of related instances (Lemma 9), and a result on 4-cycles (Theorem 2), which will be used for the two main contributions of this paper: an example of a class of instances where the optimum genomic score is arbitrarily greater than the optimum relaxed score, and a proof that the rank median problem is NP-hard, given in Sections 4 and 5. Finally, our conclusions and plans for future work are in Section 6.

2 Definitions

The rank median model, originally defined by Zanetti et al. [11], works as follows. Genomes are represented by binary, symmetric, orthogonal matrices that encode the adjacencies among gene extemities. For instance, in Figure 1 we have a genome A consisting of one circular chromosome with genes a and b, another circular chromosome with genes c and d, and a linear chromosome with genes e and f.

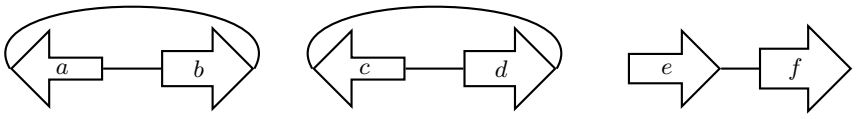

Fig. 1. Genome A.

In Figure 2, the same genome is depicted as a matching, showing just the adjacencies between gene extremities, that are marked with subscripts h for *head* and t for *tail*.

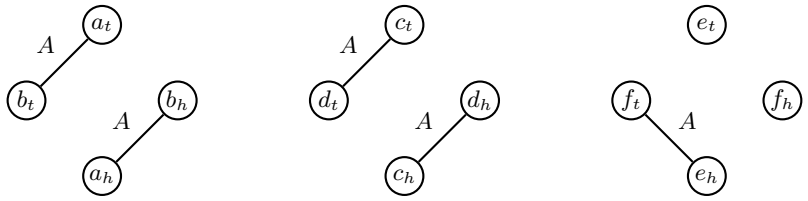

Fig. 2. Genome A as a matching.

The same genome A can be represented as a symmetric, orthogonal, 0-1 matrix as follows, with 1's outside the main diagonal indicating the adjacencies between gene extremities and 1's in the main diagonal indicating free genome extremities in the end of linear chromosomes.

$$\begin{array}{c} \\ a_t \\ a_h \\ b_t \\ b_h \\ c_t \\ c_h \\ d_t \\ d_h \\ e_t \\ e_h \\ f_t \\ f_h \end{array} \begin{array}{c} \begin{array}{cccccccccccc} a_t & a_h & b_t & b_h & c_t & c_h & d_t & d_h & e_t & e_h & f_t & f_h \end{array} \\ \left[\begin{array}{cccccccccccc} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{array}\right] \end{array}$$

In this setting, the **distance** between two genomes A and B is given by the rank of their difference as matrices:

$$d(A, B) = r(B - A).$$

In the rank median problem, the input consists of a triple of genomes $T = (A, B, C)$ and we are interested in finding genomes M that minimize the **score**

$$d(M; T) = d(M; A, B, C) = d(M, A) + d(M, B) + d(M, C). \tag{1}$$

We say that a genome M is a **median** of T, when it minimizes the score given by Equation 1. A **relaxed median** is any real-valued matrix minimizing the score. Sometimes we use the term **genomic median** to refer to a median, to stress the fact that we are dealing with a genomic matrix.

Because d satisfies the triangle inequality, for any matrix M we have

$$d(M; A, B, C) \geq \frac{d(A, B) + d(B, C) + d(C, A)}{2}, \tag{2}$$

and so the right hand side provides a lower bound for the score of a median. It is known that relaxed medians always meet this lower bound [2]. However, this is far from true for genomic medians, as we show in Section 4.

2.1 Relationship with Algebraic and DCJ Distances

We recall the basic details of algebraic and DCJ distances here. Algebraic distance, as mentioned, is exactly rank distance divided by two. As for DCJ distance, its original formulation involved capping the ends of linear chromosomes, thus creating fictitious genes, and then equalizing the number of genes in both genomes by adding null chromosomes (chromosomes with just two caps each — more fictitious genes), and closing the paths to cycles. After that, we find a minimum series of double-cut-and-join operations (see Figure 3) that transforms one genome into the other. The number of operations in this series is the DCJ distance.

It turns out that the same effect can be achieved without adding caps or null chromosomes, as long as the repertoire of operations is enlarged to include single-cut-and-join (Figure 4), as well as cuts and joins. These two last operations are just the destruction and creation of adjacencies, respectively (see Figure 5).

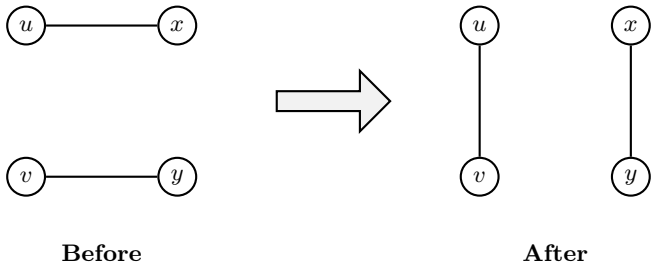

Fig. 3. Double-cut-and-join.

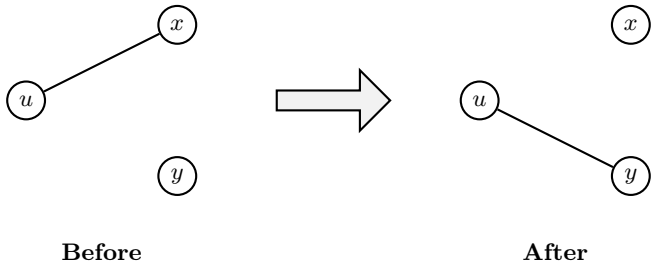

Fig. 4. Single-cut-and-join.

Biologically, this is appealing, since these basic operations implement real, observable genome large mutations such as inversions, translocations, fusions, fission, and others. The same operations can be used to define DCJ, algebraic,

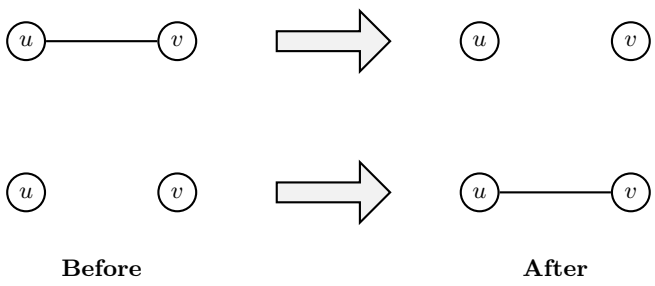

Fig. 5. Above: cut. Below: join.

and rank distance, only the weights change, and we always look for a minimum weight series of operations transforming one genome into the other. Table 1 shows the weights given by each one of the distances mentioned to the basic operations.

	DCJ	Algebraic	Rank
Cut / Join	1	0.5	1
Single-cut-and-join	1	1	2
Double-cut-and-join	1	1	2

Table 1. Weights given to basic operations by DCJ, algebraic, and rank distances.

Some aspects of the relationship among these distances are important in the NP-hardness proof, so we summarize them in the following theorem. Let d_{alg} and d_{DCJ} denote the algebraic and DCJ distances, respectively.

Theorem 1. *For any two genomes (matchings) A and B in G, we have*

$$\frac{d(A,B)}{2} = d_{alg}(A,B) \leq d_{DCJ}(A,B), \tag{3}$$

with equality between d_{alg} and d_{DCJ} when both A and B are circular genomes (perfect matchings).

Proof. The equality in Equation 3 comes from Zanetti et al. [11, Sec.2.3.2]. The inequality comes from Feijao and Meidanis [4, Eq.7], which also shows why we have equality for circular genomes. □

2.2 Matchings and Partial Medians

In this paper, we take a graph oriented approach. It is known that genomes, as defined previously, correspond to matchings of gene extremities [2]. Therefore,

from now on, we will equate genomes to matchings of a complete graph G whose vertices are the gene extremities, and proceed to study the problem from this viewpoint. This is the same approach taken by Tannier et al. [8].

We follow definitions from West [9]. We say that two edges are **independent** when they do not share endpoints. A **matching** is a set of mutually independent edges. A matching A **saturates** a vertex v, and v is called A**-saturated**, when v is incident to an edge in A. Otherwise, we call v A**-unsaturated**. An edge e is **compatible** with a matching L when e is independent from every edge in L. Two matchings L and N are **compatible** when every edge of L is compatible with N. Notice that this happens if an only if $L \cup N$ is also a matching.

Given two matchings A and B in G, we denote by $A + B$ the multigraph with the vertices of G and all edges from A and B, where edges that belong to both A and B are duplicated in $A + B$. Therefore, the connected components of $A + B$ are paths and cycles, including the 2-cycles of duplicated edges.

We translate the notion of distance to matchings. Chindelevitch et al. [2] point out that, when A and B are matchings corresponding to genomes, we have:

$$d(A, B) = |A| + |B| - 2\mathcal{C}(A, B), \tag{4}$$

where $|X|$ denotes the size of set X (in the case of matchings, their number of edges) and $\mathcal{C}(A, B)$ is the number of cycles in $A + B$, including 2-cycles.

We say that a matching L is a **partial median** with respect to an instance $T = (A, B, C)$ of the rank median problem when there exists a matching $M \supseteq L$ such that M is a median of T.

3 Preliminary Results

3.1 Modifying Matchings I

Throughout this document we will often have to inspect how the score of a matching is affected as we change its edges. For this, we need to know how distances are affected when we change edges.

We will now examine how $d(A, L)$ changes when we add a compatible edge e to an arbitrary matching L, given a fixed genome A. A first observation is that e will belong to at most one cycle in $A + (L \cup \{e\})$. Thus, e can either belong to exactly one cycle in $A + (L \cup \{e\})$ or to no cycle at all. The next result tells us how $d(A, L \cup \{e\})$ is related to $d(A, L)$ in each case.

Lemma 1. *Let A and L be matchings, and e an edge compatible with L. Then:*

$$d(A, L \cup \{e\}) = \begin{cases} d(A, L) - 1 \text{ if } \mathcal{C}(A, L \cup \{e\}) = \mathcal{C}(A, L) + 1, \\ d(A, L) + 1 \text{ otherwise.} \end{cases}$$

Proof. In the conditions of the lemma, if $\mathcal{C}(A, L \cup \{e\}) = \mathcal{C}(A, L) + 1$ we have, according to Equation 4,

$$d(A, L \cup \{e\}) = |A| + |L \cup \{e\}| - 2\mathcal{C}(A, L \cup \{e\})$$

$$
\begin{aligned}
&= |A| + |L| + 1 - 2(\mathcal{C}(A, L) + 1) \\
&= |A| + |L| - 2\mathcal{C}(A, L) - 1 \\
&= d(A, L) - 1.
\end{aligned}
$$

Otherwise, the number of cycles does not change, that is, $\mathcal{C}(A, L \cup \{e\}) = \mathcal{C}(A, L)$, and a similar reasoning leads to $d(A, L \cup \{e\}) = d(A, L) + 1$. □

This impacts the calculation of scores. Since the possible change in distance to a specific genome when adding a compatible edge e to L can be either -1 or $+1$, determined by whether e belongs to a cycle or not, we have the following corollary.

Corollary 1. *Let A, B, C, and L be matchings, and e an edge compatible with L. Then:*

$$d(L \cup \{e\}; A, B, C) = d(L; A, B, C) + s,$$

where $s = -3, -1, +1, +3$ in the cases where e belongs to a total of 3, 2, 1, or 0 cycles, respectively, in $A + L$, $B + L$, and $C + L$.

3.2 Collapsed Instances

Motivated by Lemma 1, we define the **collapse** of a matching A by a matching L, denoted by $K(A, L)$, as the set of edges that, when added to L, get it closer to A:

$$K(A, L) = \{e \text{ compatible with } L \mid \mathcal{C}(A, L \cup \{e\}) = \mathcal{C}(A, L) + 1\}.$$

The term "collapse" comes from the fact that edges of A adjacent to edges in L will disappear in $K(A, L)$. The other edges will remain, and edges not in A can be present in $K(A, L)$ as well, although every vertex saturated by $K(A, L)$ is also saturated by A. Below we list a few properties of the collapse.

Lemma 2. *If A and L are matchings, the collapse $K(A, L)$ satisfies:*

1. *$K(A, L)$ is a matching.*
2. *$K(A, L)$ is compatible with L.*
3. *$K(A, S) = A \setminus S$ when $S \subseteq A$.*
4. *$K(A, L) = A$ when L is compatible with A.*
5. *$L \subseteq M$ implies $K(A, M) \subseteq K(A, L)$.*
6. *M compatible with A implies M compatible with $K(A, L)$.*

Proof. We will prove only Item 3, which will be needed later. By definition:

$$K(A, S) = \{e \text{ compatible with } S \mid \mathcal{C}(A, S \cup \{e\}) = \mathcal{C}(A, S) + 1\}.$$

Notice that $\mathcal{C}(A, S) = |S|$, since every edge of S belongs to A and will form a 2-cycle. An edge e compatible with S will only form an extra cycle if it also belongs to A. Hence, $K(A, S) = A \setminus S$, because edges in S are not compatible with it. □

We also define the collapse of an instance $T = (A, B, C)$ by a matching L as the instance $K(T, L) = (K(A, L), K(B, L), K(C, L))$.

3.3 Modifying Matchings II

Sometimes we want to force the presence of an edge e into a matching L, even when e is not compatible with L. The straightforward way to do this would be to just remove the edges adjacent to e first. However, there is a smarter way to go about it, using collapses.

Definition 1 (Smart edge addition). *Let L be a matching and e an edge such that $e \notin L$ and incompatible with L. The* smart edge addition *of e to L is defined as*

$$L \oplus e = K(L, \{e\}) \cup \{e\}.$$

The next result parallels Lemma 1 when e is not compatible with L.

Lemma 3. *Let A and L be matchings, and e an edge not in L and incompatible with L. Then:*

$$d(A, L \oplus e) = \begin{cases} d(A, L) - 2 \text{ if } e \in A, \\ d(A, L) + s \text{ otherwise,} \end{cases}$$

where $s \leq 2$.

Proof. See Appendix A. □

We state a first consequence of these observations.

Lemma 4. *Let A, B, and C be three matchings. If e belongs to at least two of these matchings, then $\{e\}$ is a partial median of $T = (A, B, C)$.*

Proof. Let e be as in the statement, and let M be a median of T. Suppose $e \notin M$. Then, e can be either compatible with M or not.

If e is compatible with M, then $M \cup \{e\}$ has a smaller score than M, due to Lemma 1. If e is not compatible with M, then $M \oplus e$ will have a smaller score than M, due to Lemma 3. In both cases, the new score will be decreased by the contributions of the two genomes it belongs to, and the possible increase from the third genome will not be enough to offset this.

As a result, since M is a median, we rule out these possibilites and the only option left is $e \in M$, proving the claim. □

3.4 Medians in Collapsed Instances

The following result relates distances in a genome and its collapse.

Lemma 5. *If A and L are matchings, and e is an edge compatible with L, we have:*

$$d(L \cup \{e\}, A) - d(L, A) = d(\{e\}, K(A, L)) - d(\emptyset, K(A, L)).$$

Proof. According to Lemma 1 and the definition of collapse, the left hand side is equal to -1 if $e \in K(A, L)$ or to $+1$ otherwise. Likewise, the right hand side is equal to -1 if $e \in K(K(A, L), \emptyset)$ and $+1$ otherwise. However, $K(K(A, L), \emptyset) = K(A, L)$, since collapsing by $\emptyset$ does not modify a matching, as per Lemma 2, Item 3. □

What this means is that for any matching L and any compatible e, distances to a genome A are related to simpler distances in the collapsed genome $K(A, L)$. This immediately translates to rank median instances. In other words, we have the following corollary.

Corollary 2. *For any instance $T = (A, B, C)$ of the rank median problem, any matching L, and any edge e compatible with L, we have:*

$$d(L \cup \{e\}; T) - d(L; T) = d(\{e\}; K(T, L)) - d(\emptyset; K(T, L)). \tag{5}$$

Proof. This is just a question of using the definitions of $d(X; T)$ and $K(T, L)$ and applying Lemma 5 to each of the genomes A, B, and C. □

The relevant fact is that Lemma 5 can be extended to arbitrary, compatible matchings. However, we need auxiliary result to prove the extension. We begin with a few results on composite collapses.

Lemma 6. *If A and L are matchings, and e is an edge compatible with L, we have:*

$$K(A, L \cup \{e\}) = K(K(A, L), \{e\}).$$

Proof. See Appendix B.

The previous result can be extended to arbitrary, compatible matchings.

Lemma 7. *If A, L, and M are arbitrary matchings, with L and M compatible, we have:*

$$K(A, L \cup M) = K(K(A, L), M).$$

Proof. By induction on $|M|$. If $|M| = 0$, we have $M = \emptyset$ and the claim is true because $K(A, \emptyset) = A$.

If $|M| \geq 1$, let e be any edge in M, and consider $M' = M \setminus \{e\}$. Since M' is also compatible with L, the claim is true for M' by the induction hypothesis:

$$K(A, L \cup M') = K(K(A, L), M').$$

Then using Lemma 6 twice, plus the induction hypothesis, we end up with:

$$\begin{aligned} K(A, L \cup M) &= K(A, L \cup M' \cup \{e\}) \\ &= K(K(A, L \cup M'), \{e\}) \\ &= K(K(K(A, L), M'), \{e\}) \\ &= K(K(A, L), M' \cup \{e\}) \\ &= K(K(A, L), M). \end{aligned}$$

□

We are now ready for the extension of Lemma 5 to arbitrary matchings.

Lemma 8. *If A, L, and M are arbitrary matchings, with L and M compatible, then*

$$d(L \cup M, A) - d(L, A) = d(M, K(A, L)) - d(\emptyset, K(A, L)).$$

Proof. By induction on $|M|$. If $|M| = 0$ then $M = \emptyset$, and both sides are equal to zero. If $|M| \geq 1$, take any edge $e \in M$ and let M' be $M \setminus \{e\}$. This matching M' is also compatible with L, so the induction hypothesis must hold for it, leading to:

$$d(L \cup M', A) - d(L, A) = d(M', K(L, A)) - d(\emptyset, K(A, L)). \tag{6}$$

Since M is a matching, we have that $\{e\}$ is compatible with M'. Therefore, we can apply Lemma 5, obtaining:

$$d(L \cup M' \cup \{e\}, A) - d(L \cup M', A) = d(\{e\}, K(A, L \cup M')) - d(\emptyset, K(A, L \cup M')). \tag{7}$$

Also, by Lemma 5 again, we have:

$$d(M' \cup \{e\}, A') - d(M', A') = d(\{e\}, K(A', M')) - d(\emptyset, K(A', M')), \tag{8}$$

where $A' = K(A, L)$. Now adding side by side Equations 6 and 7, and subtracting Equation 8, we get:

$$d(L \cup M, A) - d(L, A) = d(M, K(A, L) - d(\emptyset, K(A, L)),$$

since $K(A, L \cup M') = K(A', M')$ by Lemma 7. □

Corollary 3. *For any instance $T = (A, B, C)$ of the rank median problem, and any matchings L, M, with M compatible with L, we have:*

$$d(L \cup M; T) - d(L; T) = d(M; K(T, L)) - d(\emptyset; K(T, L)).$$

Proof. Again, this is just a question of using the definition of $d(M; T)$ and applying Lemma 8 to each of A, B, and C. □

We are now ready for a result that relates medians in a multi-graph and their collapses.

Lemma 9. *Consider an instance $T = (A, B, C)$ of the rank median problem and let L be a partial median of T. Then, M is a median of $K(T, L)$ if and only if $L \cup M$ is a median of T.*

Proof. We start with the "only if" part. Assume $L \cup M$ is a median of T. Medians are matchings, so L and M are compatible. By definition, M is a median of $K(T, L)$ if it minimizes $d(M; K(T, L))$. By Corollary 3,

$$d(M; K(T, L)) = d(L \cup M; T) - d(L; T) + d(\emptyset; K(T, L)).$$

The only quantity dependent on M in the right hand side is the first term, and thus M is a median of $K(T, L)$ if it minimizes $d(L \cup M; T)$, which is trivially true since $L \cup M$ is a median of T.

For the "if" part, because L is a partial median, there is a matching M' compatible with L such that $L \cup M'$ is a median of T.

By Corollary 3, we have

$$d(L \cup M; T) - d(L; T) = d(M; K(T, L)) - d(\emptyset; K(T, L))$$

and

$$d(L \cup M'; T) - d(L; T) = d(M'; K(T, L)) - d(\emptyset; K(T, L)).$$

Now M is a median of $K(T, L)$, so $d(M; K(T, L)) \leq d(M'; K(T, L))$, and thus

$$\begin{aligned} d(L \cup M; T) &= d(L \cup M; T) - d(L; T) + d(L; T) \\ &= d(M; K(T, L) - d(\emptyset; K(T, L)) + d(L; T) \\ &= d(M; K(T, L)) + d(L \cup M'; T) - d(M'; K(T, L) \\ &\leq d(M'; K(T, L)) + d(L \cup M'; T) - d(M'; K(T, L)) \\ &= d(L \cup M'; T). \end{aligned}$$

But, since $L \cup M'$ is a median of T, this should be an equality, and so $L \cup M$ is also a median of T. □

The result also partly applies to partial medians.

Corollary 4. *Consider an instance $T = (A, B, C)$ of the rank median problem and let L be a partial median of T. If M is a partial median of $K(T, L)$ then $L \cup M$ is a partial median of T.*

Proof. If M is a partial median of $K(T, L)$, then there exists a median M' in $K(G, L)$ with $M \subseteq M'$. By Lemma 9, $L \cup M'$ is a median of T and it contains $L \cup M$. □

4 Relaxation Gap

As far as we know, the literature is lacking instances with a specific difference in score between relaxed medians and genomic medians. It is not even known if there is a limit on how large this difference can be. In this section, we show two classes of graphs witnessing arbitrary differences in score.

Define the *score gap* for an instance $T = (A, B, C)$ of the rank median problem as

$$g(T) = \min_{M \text{genomic}} d(M; T) - \min_{M \text{any matrix}} d(M; T).$$

Lemma 10. *If there exists a median M of $T = (A, B, C)$ such that $M \cap (A \cup B \cup C) = \emptyset$, then $\emptyset$ is a median of T.*

Proof. Let M be a matching with $M \cap (A \cup B \cup C) = \emptyset$. Denote by $\mathcal{Z}(X, Y)$ the set of cycles formed by matchings X and Y, so that $|\mathcal{Z}(X, Y)| = \mathcal{C}(X, Y)$, according to our previous definition of $\mathcal{C}(X, Y)$. Consider now the set of cycles $Z = \mathcal{Z}(M, A) \uplus \mathcal{Z}(M, B) \uplus \mathcal{Z}(M, C)$. We are using disjoint union because we want to consider an eventual cycle that appears, say, in $\mathcal{Z}(M, A)$ and also in $\mathcal{Z}(M, B)$ as two distinct entities. Now build a bipartite graph $\mathcal{B}$ with vertices in Z and M and edges linking a cycle $c \in Z$ to $e \in M$ exactly when $e \in c$. Let x_c be the degree of cycle $c \in Z$ in $\mathcal{B}$, and y_e the degree of $e \in M$ in $\mathcal{B}$.

Since M does not contain any edge in $A \cup B \cup C$, any cycle $c \in Z$ has degree at least two in $\mathcal{B}$, and thus

$$2|Z| \leq \sum_{c \in Z} x_c.$$

We also know that each edge $e \in M$ can belong to at most three cycles in Z (at most one in each of $Z(A, M)$, $Z(B, M)$, and $Z(C, M)$) and thus

$$\sum_{e \in M} y_e \leq 3|M|.$$

Since both sums equal the number of edges in $\mathcal{B}$, we have $2|Z| \leq 3|M|$. With that we can calculate a lower bound on M's score:

$$\begin{aligned} d(M; A, B, C) &= d(M, A) + d(M, B) + d(M, C) \\ &= 3|M| + |A| + |B| + |C| - 2(\mathcal{C}(A, M) + \mathcal{C}(B, M) + \mathcal{C}(C, M)) \\ &= 3|M| + |A| + |B| + |C| - 2|Z| \\ &\geq 3|M| + |A| + |B| + |C| - 3|M| \\ &= |A| + |B| + |C| \\ &= d(\emptyset; A, B, C). \end{aligned}$$

Thus, if M is a median, so is $\emptyset$. □

Lemma 11. *Let $G = G_1 \uplus G_2$, where $\uplus$ denotes disjoint union. Let A_1, B_1, and C_1 be matchings in G_1, and A_2, B_2, and C_2 be matchings in G_2. Finally, let M_1 be a median of A_1, B_1, and C_1, and M_2 be a median of A_2, B_2, and C_2. Then $M_1 \cup M_2$ is a median of $A = A_1 \cup A_2$, $B = B_1 \cup B_2$, and $C = C_1 \cup C_2$ in G.*

Proof. Let M be a median of A, B, and C. For $i = 1, 2$, take N_i as the set of edges of M that are incident to vertices in G_i. Since M is a median, and given that N_1 and N_2 are compatible, we have that $N = N_1 \cup N_2$ is a partial median, and by collapsing with it we get another instance $(K(A, N), K(B, N), K(C, N))$ of the rank median problem, with a solution $M' = M \setminus N$ by Lemma 9.

Now, since N contains all edges in M with both endpoints in the same subgraph, the matching M' can only contain edges with one end in G_1 and another in G_2. Because G_1 and G_2 are disconnected, and recalling that a collapse never augments the set of saturated vertices, we conclude that $M' \cap (E[K(A, N)] \cup E[K(B, N)] \cup E[K(C, N)]) = \emptyset$, and then we can apply Lemma 10 to get $\emptyset$ as a

median of $K(A, N)$, $K(B, N)$, and $K(C, N)$. By Lemma 9 again, N is a median of A, B, and C.

Note that, since both G and N have no edges between the subgraphs G_1 and G_2, then there are also no cycles between them, and thus N minimizes the score for G_1 and G_2 separately. This implies that N_i is a median of A_i, B_i, and C_i, for $i = 1, 2$. Since M_1 and M_2 are also medians, we have

$$d(N_1; A_1, B_1, C_1) = d(M_1; A_1, B_1, C_1),$$
$$d(N_2; A_2, B_2, C_2) = d(M_2; A_2, B_2, C_2).$$

Now

$$\begin{aligned} d(M_1 \cup M_2; A, B, C) &= d(M_1; A_1, B_1, C_1) + d(M_2; A_2, B_2, C_2) \\ &= d(N_1; A_1, B_1, C_1) + d(N_2; A_2, B_2, C_2) \\ &= d(N; A, B, C), \end{aligned}$$

showing that $M_1 \cup M_2$ is a median of A, B, and C, as it equals the score of another median, namely, N. □

Define an *AB 4-cycle* as a cycle with 4 edges, two of which are in matching A and the other two in matching B. If we have an instance $T = (A, B, C)$ of the rank median problem with just an AB 4-cycle and an empty C, it has a score gap of 1. Lemma 11 implies that, to achieve a score gap of k, it suffices to build an instance with k disjoint AB 4-cycles and empty C.

That result by itself does not say anything about the score gap for connected graphs. For the rest of this section, we will show how to join the AB 4-cycles with C-edges into a connected graph, without decreasing the score gap. We start with a result on 4-cycles.

Lemma 12. *If there is an AB 4-cycle in an instance $T = (A, B, C)$, then there is at least one edge e of the 4-cycle such that $\{e\}$ is a partial median.*

Proof. Let M be a median and let $N \subseteq M$ be the set of edges in M incident to the vertices of the cycle. Since M is a median, $M \setminus N$ is a partial median, and thus $K(G, M \setminus N)$ is a graph where N is a median.

Note that, because the cycle is a 4-cycle, there exists a subset of at most 8 vertices that contains both the vertices of the 4-cycle and the vertices incident to N. We can verify that at least one edge of the 4-cycle is a partial median by brute-forcing through all possible 8-graph vertices that contain an AB 4-cycle. See Appendix C for a piece of code that performs this verification. It runs in about 10 minutes in most modern computers or laptops.

Choosing a median N' that contains one of those edges, we conclude that $M \setminus N \cup N'$ is a median of G by Lemma 9. □

Corollary 5. *If there is an AB 4-cycle in G, then at least one pair of edges in that cycle is a partial median.*

Proof. By collapsing the partial median edge e given by Lemma 12, we get a 2-cycle on the opposite edge f of the cycle. By Lemma 4, that edge f is a partial median in $K(G, \{e\})$. We conclude that $\{e, f\}$ is a partial median in G by Corollary 4. □

Theorem 2. *If the genomes A, B of $T = (A, B, C)$ are perfect matchings, with AB cycles being either 2-cycles or 4-cycles, then there exists a median given by choosing either the edges in A or B for each AB cycle.*

Proof. This can be done through a proof by induction on the number n of AB cycles. If $n = 1$, Lemma 4 and Corollary 5 show that either A or B is a partial median, which is in fact a median since both A and B are perfect matchings.

For $n \geq 2$, let M be a partial median in one of the AB cycles. Now $K(T, M)$ is a graph with $K(A, M)$ and $K(B, M)$ as perfect matchings, again with each AB-cycle being of size 2 or 4, so the induction hypothesis applies, yielding a median M' in $K(T, M)$. By Lemma 9, the matching $M \cup M'$ satisfies the theorem. □

Now, let a **path of cycles** be an instance $T = (A, B, C)$ constructed by joining k disjoint AB 4-cycles, with C being composed of $k-1$ edges joining the ith cycle with the $(i+1)$th cycle, for $1 \leq i < k$.

By Theorem 2, for any path of cycles there exists a median M in this graph that is given by choosing either A or B for each 4-cycle. Now, independently of whether the A or B edges are chosen, we have that the number of AM-cycles plus the number of BM-cycles is equal to 3 for each AB cycle, and we can also verify that there are no cycles in $C + M$, so

$$\begin{aligned} \mathcal{C}(A, M) + \mathcal{C}(B, M) &= 3k, \\ \mathcal{C}(C, M) &= 0. \end{aligned}$$

Then, the score of the genomic median is

$$\begin{aligned} d(M; A, B, C) &= d(A, M) + d(B, M) + d(C, M) \\ &= 3|M| + |A| + |B| + |C| - 2\mathcal{C}(A, M) - 2\mathcal{C}(B, M) - 2\mathcal{C}(C, M) \\ &= 6k + 2k + 2k + (k-1) - 6k - 0 \\ &= 5k - 1. \end{aligned}$$

The relaxed median R always reaches the lower bound, so

$$\begin{aligned} d(R; A, B, C) &= \frac{1}{2}\left(d(A, B) + d(B, C) + d(C, A)\right) \\ &= \frac{1}{2}\left(2|A| + 2|B| + 2|C| - 2\mathcal{C}(A, B) - 2\mathcal{C}(B, C) - 2\mathcal{C}(C, A)\right) \\ &= \frac{1}{2}(4k + 4k + (2k-2) - 2k - 0 - 0) \\ &= 4k - 1. \end{aligned}$$

Thus their difference is

$$d(M; A, B, C) - d(R; A, B, C) = k$$

that is, we have a class of connected graphs in which the score gap grows with 1/4 of the number of vertices in the graph.

5 Complexity of the Rank Median Problem

Although relaxed medians can be calculated in polynomial time, the fact that a connected graph can have a genomic median with a score arbitrarily distant from the score of a relaxed median suggests that the rank median problem does not have a polynomial time algorithm.

As it turns out, the rank median problem can be proven to be NP-hard building upon arguments that worked for the DCJ median problem. Tannier et al. proved that the DCJ median problem is NP-hard [8, Thm 5], even for circular genomes, which correspond to perfect matchings. The proof was done through a reduction from the **Breakpoint Graph Decomposition** (BGD) problem. Looking at the reduction to the DCJ problem, a stronger statement can be made.

Theorem 3. *The DCJ median problem is NP-hard even for instances $T = (A, B, C)$ where A, B, and C are perfect matchings and $A + B$ is a disjoint union of 4-cycles.*

Proof. In the reduction performed by Tannier et al. [8] only genomes of this kind appear when an instance of BGD is transformed into an instance of DCJ median. □

Now, we are ready for the result on rank medians.

Theorem 4. *The rank median problem for three genomes is NP-hard.*

Proof. We use a reduction from DCJ median, where A and B are perfect matchings and $A + B$ is a disjoint union of 4-cycles. If there was a polynomial time program P that solved rank medians, we could feed such an instance T to it and, by Theorem 2, we would get back a rank median M that is also a perfect matching. Since DCJ and rank distances differ by a factor of 2 for circlular genomes (by Theorem 1), we have:

$$d_{DCJ}(M; T) = \frac{d(M; T)}{2}.$$

We claim that M is also a DCJ median. Indeed, if there was a genome M' with smaller DCJ score, it would also have smaller rank score, because

$$\frac{d(M'; T)}{2} \leq d_{DCJ}(M'; T) < d_{DCJ}(M; T) = \frac{d(M; T)}{2},$$

by Theorem 1. But this is impossible, given that M is a rank median. We conclude that M is indeed a DCJ median.

Therefore, a solver for rank medians would be a DCJ median solver as well, for instances of the kind specified above. Since DCJ median is NP-hard for these instances, so is the rank median problem for three genomes. □

6 Conclusions

We have given an example of a class of graphs such that the error of their medians differs from the lower bound given by Equation 2 by one fourth of the number of vertices.

We have also proven that the rank median problem is NP-hard, and given some tools that can help with the problem of finding a median as well as proving statements about medians on a certain class of instances. More specifically, Lemma 9 shows how, if we know that some edge set belongs to at least one median, to find a median for the entire instance by finding the median on a smaller instance.

We have also given a class of instances, in Theorem 2, that contains both the class of instances mentioned above as well as the instances used in the reductions for the NP-hardness proof. These instances have the special property of having a guaranteed median that belongs to a relatively smaller search space, and with the score of the matchings in that search space changing only with the number of cycles between the matching and one of the genomes.

6.1 Future Work

Although we have shown ways to simplify the problem of finding a median in an instance if it contains a 2-cycle or an alternate 4-cycle, the NP-hardness result implies that, in general, we cannot do much better than brute-forcing over all possible matchings in the corresponding complete graph to find a median.

The most straightforward way of finding all possible matchings is through a recursive function that chooses an edge in a graph, and for all matchings in the subgraph with that edge removed, create one matching that contains the edge and another that does not, as in Algorthm 1. This algorithm finds all possible matchings, and by keeping track of all their scores, it can be used to find all medians of an input instance.

By Lemma 10, to find a median in $T = (A, B, C)$, it suffices to be able to look for matchings that are subsets of $A \cup B \cup C$; together with Lemma 9, this would give a brute-force algorithm for finding a median: we iterate over all edges in $A \cup B \cup C$, and for all medians found in the instance resulting from the collapse of the current edge, construct a matching with the current edge and the current median of the collapsed instance, and choose the ones with the lowest score (comparing it with the score of the empty matching too). Algorithm 2 implements this idea.

Algorithm 1 Straightforward function that returns all matchings in a graph

```
1: function MATCHINGS(G)
2:     m ← ∅
3:     e ← an arbitrary edge of G
4:     for each matching M in MATCHINGS(G \ {e}) do
5:         m ← m ∪ {M}
6:         m ← m ∪ {M ∪ {e}}
7:     end for
8:     return m
9: end function
```

Algorithm 2 Median finding by collapse

```
1: function MEDIANS(T = (A, B, C))
2:     m ← {∅}
3:     s ← |A| + |B| + |C|
4:     for e ∈ A ∪ B ∪ C do
5:         m′ ← MEDIANS(K(T, e))
6:         n ← min{d(M ∪ {e}; T) : M ∈ m′}
7:         if n = s then
8:             m ← m ∪ {M ∈ m′ : d(M ∪ {e}; T) = n}
9:         end if
10:        if n < s then
11:            m ← {M ∈ m′ : d(M ∪ {e}; T) = n}
12:            s ← n
13:        end if
14:    end for
15:    return m
16: end function
```

However, the number of matchings checked by this algorithm, without any optimizations, may be greater than the number of matchings of G in some situations, for example, when A, B, and C are perfect matchings, and so it is not necessarily more efficient. This shows that there is room for improvement in this algorithm.

One possible direction for future work would be to optimize this algorithm by avoiding the inclusion of the same edge in different recursive branches, and investigate whether or not that would result in a better algorithm for finding medians. Another direction would be to prove or find a counterexample for the hypothesis that all medians that do not intersect A, B, or C are necessarily empty. If that hypothesis is true, it would imply Algorithm 2 is capable of finding all possible medians, rather than a non-empty subset of them.

Acknowledgments. This study was financed, in part, by the São Paulo Research Foundation (FAPESP), Brazil. Process Numbers #2024/01200-8 and #2025/05185-6.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

A Proof of Lemma 3

In this section, we need the following auxiliary lemma, that specifies exactly what a collapse by a single edge looks like.

Lemma 13. *If A is any matching and e is any edge, we have:*

$$K(A,\{e\}) = \begin{cases} A \setminus \{e\} & \text{if } e \in A, \\ A & \text{if } e \text{ is compatible with } A, \\ A \setminus \{f\} & \text{if } e \notin A \text{ is incident to exactly one edge } f \in A, \\ A \setminus \{f, f'\} \cup \{e'\} & \text{if } e \notin A \text{ is incident to two edges } f, f' \in A. \end{cases}$$

where e' is the edge that forms a 4-cycle with e, f, and f'.

Proof. By definition:

$$K(A,\{e\}) = \{g \text{ compatible with } \{e\} \mid \mathcal{C}(A,\{e,g\}) = \mathcal{C}(A,\{e\}) + 1\}.$$

But:

$$\mathcal{C}(A,\{e\}) = \begin{cases} 1 \text{ if } e \in A, \\ 0 \text{ if } e \notin A. \end{cases}$$

Consider the case when $e \in A$. Then $\mathcal{C}(A,\{e\}) = 1$. To belong to $K(A,\{e\})$, an edge g needs to be part of another cycle in $A + \{e,g\}$. This happens if and only if g is an edge of A different from e. Hence, $K(A,\{e\}) = A \setminus \{e\}$ in this case.

Now assume that $e \notin A$. Then $\mathcal{C}(A,\{e\}) = 0$. To belong to $K(A,\{e\})$, an edge g needs to be independent from e and be part of cycle in $A + \{e,g\}$. Edges that form cycles in $A + \{e,g\}$ are the ones that connect nodes in the same

connected component in $A + \{e\}$. All these connected components are paths. If e is compatible with A, all these components are single edges, and therefore $K(A, \{e\}) = A$, since we have to exclude e for not being compatible with itself. If there is only one edge $f \in A$ incident to e, its component is a 3-path containing both e and f, and there is no way to form a cycle with an edge compatible with e there. Then $K(A, \{e\}) = A \setminus \{f\}$ in this case, since the other edges of A form cycles.

Finally, if e is incident to two edges $f, f' \in A$, then the edge e' that forms a 4-cycle with e, f, and f' is compatible with e and forms a cycle, as do the edges of A except f and f'. Therefore, $K(A, \{e\}) = A \setminus \{f, f'\} \cup \{e'\}$ in this case. □

Lemma 3. *Let A and L be matchings, and e an edge not in L and incompatible with L. Then:*

$$d(A, L \oplus e) = \begin{cases} d(A, L) - 2 \text{ if } e \in A, \\ d(A, L) + s \text{ otherwise,} \end{cases}$$

where $s \leq 2$.

Proof. By case analysis. We have two possibilities for edge e: either $e \in A$ or $e \notin A$.

1. **Edge $e \in A$**: Since $e \notin L$ and e is not compatible with L, we have only two subcases here: either e is incident to a single edge $f \in L$ or it is incident to two, $f, f' \in L$.
   (a) **Edge e is incident to a single edge $f \in L$**: In this first subcase, $L \oplus e = L \setminus \{f\} \cup \{e\}$, so $|L \oplus e| = |L|$. Then $A + (L \oplus e)$ will have a 2-cycle in e while $A + L$ does not have any cycles involving e or f, so $\mathcal{C}(A, L \oplus e) = \mathcal{C}(A, L) + 1$. Therefore:

$$\begin{aligned} d(A, L \oplus e) &= |A| + |L \oplus e| - 2\mathcal{C}(A, L \oplus e) \\ &= |A| + |L| - 2\mathcal{C}(A, L) - 2 \\ &= d(A, L) - 2, \end{aligned}$$

   as claimed.
   (b) **Edge e is incident to two edges $f, f' \in L$**: In this second subcase for $e \in A$, we have e incident to two edges $f, f' \in L$. Then $L \oplus e = L \setminus \{f, f'\} \cup \{e, e'\}$, so $|L \oplus e| = |L|$ again and $A + (L \oplus e)$ will have a 2-cycle in e that does not exist in $A + L$. If the connected component of e, f, and f' in $A + L$ is a path (see Figure 6, left), then e' will not be in a cycle in $A + (L \oplus e)$. But if this component is a cycle $ef'Pf$ in $A + L$ (Figure 6, right), this cycle will be shortened to $e'P$ when f and f' are replaced by e and e'. In any case, $\mathcal{C}(A, L \oplus e) = \mathcal{C}(A, L) + 1$ and the same derivation as above shows that $d(A, L \oplus e) = d(A, L) - 2$.
2. **Edge $e \notin A$**: Here again we have that e is incident to either one or two edges of L. As in the previous block, in any case we have $|L \oplus e| = |L|$. All we have to do here is to show that $\mathcal{C}(A, L \oplus e) \geq \mathcal{C}(A, L) - 1$, because this leads to:

$$s = d(A, L \oplus e) - d(A, L)$$

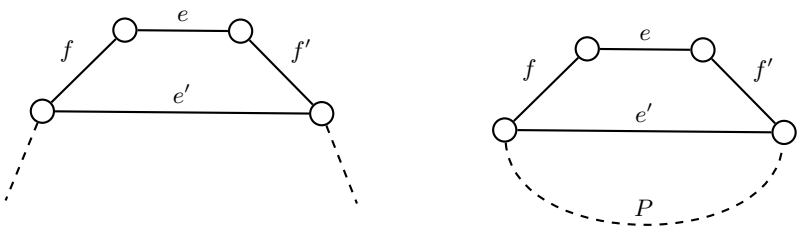

Fig. 6. Two subcases where $e \in A$. In the left picture, e, f, and f' belong to a path in $A + L$. In the picture to the right, e, f, and f' belong to a cycle in $A + L$. The dots in the pictures represent arbitrarily long, alternating A-L paths.

$$
\begin{aligned}
&= |A| + |L \oplus e| - 2\mathcal{C}(A, L \oplus e) - |A| - |L| + 2\mathcal{C}(A, L) \\
&= -2\mathcal{C}(A, L \oplus e) + 2\mathcal{C}(A, L) \\
&\leq -2(\mathcal{C}(A, L) - 1) + 2\mathcal{C}(A, L) \\
&= 2.
\end{aligned}
$$

In other words, all we have to do is convince ourselves that the multigraph $A + (L \oplus e)$ has at most one less cycle than $A + L$. However, edge addition does not lead to cycle loss, and edge removal accounts for at most one cycle lost per edge removed. Consequently, we do not have to worry about the case where e is incident to a single edge $f \in L$, since in this case a single edge is removed, causing the loss of at most one cycle.
The real concern is the case where e is incident to two edges $f, f' \in L$, because then there is the potential of losing two cycles as we remove f and f' from L and add e and e'. But, for this potential to materialize, both f and f' must belong to cycles in $A + L$, and distinct ones for that matter, as depicted in Figure 7. However, in this situation, notice that we also create a new cycle $ePe'P'$ in $A + (L \oplus e)$. Therefore, the net loss is at most one cycle, since we lose two but create one cycle.

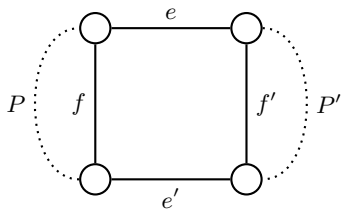

Fig. 7. Case 2: The removal of f and f' leads to the loss of two cycles. However, the inclusion of e and e' causes a new cycle to appear: $ePe'P'$. The dotted lines P and P' represent arbitrarily long, alternating A-L paths.

□

B Proof of Lemma 6

Lemma 6. *If A and L are matchings, and e is an edge compatible with L, we have:*

$$K(A, L \cup \{e\}) = K(K(A, L), \{e\}).$$

Proof. Consider a connected component X of $A + (L \cup \{e\})$ that does not contain e. This component will be present in $A + L$ as well. This component will contribute one edge to $K(A, L \cup \{e\})$ if it is a path with both vertices of degree one A-saturated, and no edge at all otherwise. The component's contribution to $K(A, L)$ will be exactly the same, since $e \notin X$. Therefore, $K(A, L)$ and $K(A, L \cup \{e\})$ receive identical contributions from all connected components not containing e.

Therefore, we focus on the connected component of e in the multigraph $A + (L \cup \{e\})$. This component can be a cycle or a path. If it is a path, we focus on the A-saturation status of its extreme points, that is, the degree one nodes of this path. They may be both A-unsaturated, one A-unsaturated an the other A-saturated, or both A-saturated. We have therefore 4 cases, analyzed below.

1. **Edge e belongs to a cycle in $A + (L \cup \{e\})$**: Then $e \in K(A, L)$ by definition. In this case, $K(A, L \cup \{e\}) = K(A, L) \setminus \{e\}$, because e is removed from the collapse, no other edge in this cycle belong to either set, and the sets agree on the contributions from other connected components. On the other hand, given that $e \in K(A, L)$, Lemma 13 results in $K(K(A, L), \{e\}) = K(A, L) \setminus \{e\}$ as well.
2. **Edge e belongs to a path in $A + (L \cup \{e\})$**: Here we have three subcases, according to the A-saturation status of the path extremities.
   (a) **Both extremities A-unsaturated**: In this case, $e \notin K(A, L)$ and the entire path is absent from this set as well. Adding e to L will not change this, so $K(A, L \cup \{e\}) = K(A, L)$, as they agree in the contributions from other components. On the other hand, e is compatible with $K(A, L)$, because the entire path is absent, and Lemma 13 leads to $K(K(A, L), \{e\}) = K(A, L)$ as well.
   (b) **One extremity A-unsaturated and one A-saturated**: In this case, $e \notin K(A, L)$, the entire path is absent from this set as well, but there is an extra edge f in $K(A, L)$ linking an endpoint of e with the A-saturated extremity of the path (see Figure 8). Adding e to L will remove this edge, so $K(A, L \cup \{e\}) = K(A, L) \setminus \{f\}$, as they agree in the contributions from other components. On the other hand, e is now not compatible with $K(A, L)$ and is incident to a single edge f in this set. Lemma 13 will imply that $K(K(A, L), \{e\}) = K(A, L) \setminus \{f\}$ as well.
   (c) **Both extremities A-staturated**: In this case, $e \notin K(A, L)$, the entire path is absent from this set as well, but there are two extra edges f and f' in $K(A, L)$ linking the endpoints of e with the A-saturated extremities

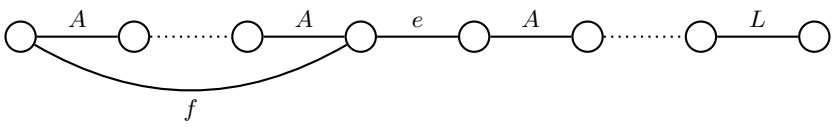

Fig. 8. Case 2b: One extremity A-unsaturated and one A-saturated in the e-path. The dots in the picture represent arbitrarily long, alternating A-L paths.

of the path (see Figure 9). Adding e to L will remove these edges, but will also include a third edge e' linking both extremities of the path, so $K(A, L \cup \{e\}) = K(A, L) \setminus \{f, f'\} \cup \{e'\}$, as they agree in the contributions from other components. On the other hand, e is again not compatible with $K(A, L)$ and is indicent to f and f' in this set. Lemma 13 then implies that $K(K(A, L), \{e\}) = K(A, L) \setminus \{f, f'\} \cup \{e'\}$ as well.

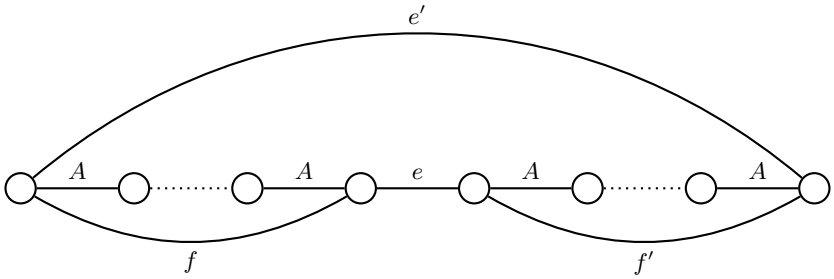

Fig. 9. Case 2c: Both extremities of the e-path are A-saturated. The dots in the picture represent arbitrarily long, alternating A-L paths.

We see that in all cases we have $K(A, L \cup \{e\}) = K(K(A, L), \{e\})$, proving the initial claim. □

C Code

To verify Lemma 12, copy the following code to a file `test.py` and run it with `python test.py`. The word 'problem' should not appear in the output.

```
#!/usr/bin/env python3

########################################################
### Function that returns all matchings for a list
### of vertices.

def matchings(l):
```

```
    """
    Returns all matchings of elements of list l
    """
    if len(l) <= 1:
        return [[]]
    else:
        m = matchings(l[1:])
        for i in l[1:]:
            l1 = l[1:]
            l1.remove(i)
            m += [m1 + [(l[0],i)] for m1 in matchings(l1)]
        return m

#######################################################
### Node class for union-find structure

class Node:
    def __init__(self, value):
        self.value = value
        self.parent = self

#######################################################
### Main class for union-find structure

class UnionFind:
    def __init__(self):
        self.nodes = {}

    def find(self, value):
        if value not in self.nodes:
            self.nodes[value] = Node(value)
            return value

        current_node = self.nodes[value]
        while current_node.parent != current_node:
            current_node = current_node.parent

        return current_node.value

    ### This is a special form of the union operation.
    ### It returns 1 if the arguments already belong
    ### to the same component, and 0 otherwise.
    ### In the context of two matchings, returning 1
    ### means closing a cycle.

    def union(self, value1, value2):
        root1 = self.find(value1)
        root2 = self.find(value2)

        if root1 != root2:
```

```
            self.nodes[root2].parent = self.nodes[root1]
            return 0
        else:
            return 1

######################################################
### Distance computation using union-find structure to
    determine
### the connected components of the union of two matchings.

def dist(x, y):
    """
    Compute distance between matchings.

    Method: use union-find structire to record the connected
    components of the union of two matchings.
    For each edge in either matching, add 1 to distance.
    For each cycle closed (union returns 1), subtract 2
    from distance.
    """
    d = 0
    uf = UnionFind()
    for u, v in x:
        d += 1
        uf.union(u, v)
    for u, v in y:
        d += 1
        d -= 2 * uf.union(u, v)
    return d

def score(m, a, b, c):
    """
    Computes the score d(m; a, b, c).
    """
    return dist(m, a) + dist(m, b) + dist(m, c)

def medians(a, b, c, candidates):
    """
    Returns all medians of the first three parameters
    among all matchings given by the fourth parameter.
    """
    minimum = -1
    for m in candidates:
        sc = score(m, a, b, c)
        if minimum == -1 or sc < minimum:
            minimum = sc
            medians = [m]
        elif sc == minimum:
            medians.append(m)
    return medians
```

```
def occur(edges, lists):
    """
    Returns true when at least one of the edges
    belongs to at least one of the medians.
    """
    for e in edges:
        for l in lists:
            if e in l:
                return True
    return False

#######################################################
### Tests whether there is always an edge that is a
### partial median in a 4-cycle involving A and
### B in a graph with 8 vertices.

def main():
    ## Edges making up the 4-cycle
    ## It is important that all edges are increasing
    ### e.g., write (1,2), not (2,1)
    initial_a = [(0,1), (2,3)]
    initial_b = [(1,2), (0,3)]

    ## Complement: possible matchings on the
    ## vertices not involved in the initial edges
    compl = matchings(list(range(4,8)))

    ## All 8-vertex matchings: to loop over for
    ## the C matching and for median candidates
    all = matchings(list(range(8)))

    for ca in compl:
        ## ca is the complement for A
        a = initial_a + ca
        print("A", a)
        for cb in compl:
            ## cb is the complement for A
            b = initial_b + cb
            print("B", b)
            for c in all:
                print("C", c)
                ## Collects all medians, see if some edge
    belongs
                mds = medians(a, b, c, all)
                print("MS", mds)
                if not occur(initial_a + initial_b, mds):
                    print('problem', a, b, c, mds)

if __name__ == "__main__":
```

```
main()
```

References

1. Chindelevitch, L., La, S., Meidanis, J.: A cubic algorithm for the generalized rank median of three genomes. Algorithms for Molecular Biology **14**(1), 16 (Jul 2019). https://doi.org/10.1186/s13015-019-0150-y
2. Chindelevitch, L., Pereira Zanetti, J.P., Meidanis, J.: On the rank-distance median of 3 permutations. BMC Bioinformatics **19**(6), 142 (May 2018). https://doi.org/10.1186/s12859-018-2131-4
3. Dias, Z., Meidanis, J.: Genome rearrangements distance by fusion, fission, and transposition is easy. In: Proceedings Eighth Symposium on String Processing and Information Retrieval. pp. 250–253 (2001). https://doi.org/10.1109/SPIRE.2001.989776
4. Feijão, P., Meidanis, J.: Extending the algebraic formalism for genome rearrangements to include linear chromosomes. IEEE/ACM Transactions on Computational Biology and Bioinformatics **10**, 819–831 (2012), https://api.semanticscholar.org/CorpusID:2303303
5. Feijão, P.C.: On genome rearrangement models. Ph.D. thesis, University of Campinas (UNICAMP), Campinas, Brazil (2012), https://repositorio.unicamp.br/acervo/detalhe/865187
6. Meidanis, J., Chindelevitch, L.: Fast median computation for symmetric, orthogonal matrices under the rank distance. Linear Algebra and its Applications **614**, 394–414 (Apr 2021). https://doi.org/10.1016/j.laa.2020.10.030
7. Meidanis, J., Dias, Z.: An alternative algebraic formalism for genome rearrangements. In: Sankoff, D., Nadeau, J.H. (eds.) Comparative Genomics: Empirical and Analytical Approaches to Gene Order Dynamics, Map Alignment and the Evolution of Gene Families, pp. 213–223. Springer Netherlands, Dordrecht (2000), https://doi.org/10.1007/978-94-011-4309-7_20
8. Tannier, E., Zheng, C., Sankoff, D.: Multichromosomal median and halving problems under different genomic distances. BMC Bioinformatics **10**(1), 120 (Apr 2009). https://doi.org/10.1186/1471-2105-10-120
9. West, D.: Introduction to Graph Theory. Prentice-Hall (2001)
10. Yancopoulos, S., Attie, O., Friedberg, R.: Efficient sorting of genomic permutations by translocation, inversion and block interchange. Bioinformatics **21**(16), 3340–3346 (Aug 2005). https://doi.org/10.1093/bioinformatics/bti535
11. Zanetti, J.P.P., Biller, P., Meidanis, J.: Median approximations for genomes modeled as matrices. Bulletin of mathematical biology **78**(4), 786–814 (2016). https://doi.org/10.1007/s11538-016-0162-4