Sorting by Reversals, Transpositions, and Indels on Both Gene Order and Intergenic Sizes

Klairton Lima Brito[1(✉)], Géraldine Jean[2], Guillaume Fertin[2], Andre Rodrigues Oliveira[1], Ulisses Dias[3], and Zanoni Dias[1]

[1] Institute of Computing, University of Campinas, Campinas, Brazil
{klairton,andrero,zanoni}@ic.unicamp.br
[2] LS2N, UMR CNRS 6004, University of Nantes, Nantes, France
{geraldine.jean,guillaume.fertin}@univ-nantes.fr
[3] School of Technology, University of Campinas, Limeira, Brazil
ulisses@ft.unicamp.br

Abstract. During the evolutionary process, the genome is affected by various genome rearrangements, which are events that modify large stretches of the genetic material. In the literature, several models were designed to estimate the number of events that occurred during the evolution, but these models represent genomes as a sequence of genes, overlooking the genetic material between consecutive genes. However, recent studies show that taking into account the genetic material present between consecutive genes can be more realistic. Reversal and transposition are genome rearrangements widely studied in the literature. A reversal inverts a segment of the genome while a transposition swaps the positions of two consecutive segments. Genomes also undergo non-conservative events (events that alter the amount of genetic material) such as insertion and deletion, which insert and remove genetic material from intergenic regions of the genome, respectively. We study problems considering both gene order and intergenic regions size. We investigate the reversal distance between two genomes in two scenarios: with and without non-conservative events. For both problems, we show that they belong to NP-hard problems class and we present a 4-approximation algorithm. We also study the reversal and transposition distance between two genomes (and the variation with non-conservative events) and we present a 6-approximation algorithm.

Keywords: Genome rearrangements · Intergenic regions · Approximation algorithms

1 Introduction

Genome rearrangements are events that insert, remove or change the position and/or orientation of large stretches of the genetic material. When we compare

Z. Cai et al. (Eds.): ISBRA 2019, LNBI 11490, pp. 28–39, 2019.
https://doi.org/10.1007/978-3-030-20242-2_3

the genomes of two individuals, one of the main goals is to estimate the sequence of rearrangement events that transformed one genome into the other.

A model $\mathcal{M}$ determines the set of rearrangement events allowed to modify the genome. The size of the smallest sequence of rearrangement events in a model $\mathcal{M}$ capable of transforming a genome into another is called *rearrangement distance*. When we assume no duplicated gene, we can map every gene to a unique number to represent genomes as permutations of integer elements. Usually, rearrangement problems are treated as sorting problems and the goal is to turn any genome π into a specific genome $\iota = (1, \ldots, n)$, which is called *identity permutation*. If the orientation of the genes is known, a positive or negative sign is assigned to each element. Otherwise, signs are omitted.

When the gene orientations are unknown, Caprara [6] proved that the Sorting Permutations by Reversals problem belongs to NP-hard problems class. The best algorithm has an approximation factor of 1.375 and was presented by Berman *et al.* [1]. Sorting Permutations by Transpositions problem also belongs to NP-hard problems class [4] and the best algorithm has an approximation factor of 1.375 [7]. When we consider a model allowing reversals and transpositions we have the Sorting Permutations by Reversals and Transpositions problem. The best algorithm for this problem has an approximation factor of $2.8334 + \epsilon$, for any $\epsilon > 0$ [10] and its complexity is unknown.

The representation of a genome only as a gene sequence is a technique very useful, but all the information that is not present on the genes are discarded that implies in loss of information. In particular, the intergenic regions between consecutive genes are not taken into account by these representations. Recently, authors have suggested that incorporating this information may improve the distance estimate from an evolutionary point of view [2,3]. Thus, it is justified to perform investigations considering the order of the genes and the size of the intergenic regions. Works considering gene order and intergenic sizes have already been presented. A model allowing Double-Cut and Join (DCJ) operation was presented together with the NP-hard proof and a 4/3-approximation algorithm [8]. The DCJ [11] is a rearrangement event that cuts the genome into two points and the extremities of the resulting segments are reassembled following specific criteria. A model allowing DCJs, insertions, and deletions also was investigated and an exact polynomial time algorithm was designed [5] when insertions and deletions act only on intergenic regions. Besides, Oliveira *et al.* [9] presented a model that allows the use of only super short reversals (reversals that affect at most two genes) also considering intergenic regions size.

In this paper, we consider that gene orientations are unknown. We investigate four problems to estimate the distance between genomes also taking into account intergenic regions, two of them allowing just conservative events of reversal and transposition: *Sorting by Intergenic Reversals (SbIR)* and *Sorting by Intergenic Reversals and Transpositions (SbIRT)*. We also investigate two other problems that allow non-conservative events of insertion and deletion: *Sorting by Intergenic Reversals, Insertions, and Deletions (SbIRID)* and *Sorting by Intergenic Reversals, Transpositions, Insertions, and Deletions (SbIRTID)*.

This manuscript is organized as follows. Section 2 provides definitions that are used throughout the paper. Section 3 presents the complexity proofs for SbIR and SbIRID problems and an approximation algorithm for each problem addressed in this manuscript. Section 4 concludes the paper.

2 Basic Definitions

Given a genome $\mathcal{G}$ as a sequence of n genes $g_1, \ldots, g_n$ and a sequence of $n+1$ intergenic regions $r_1, \ldots, r_{n+1}$, each gene is surrounded by two intergenic regions so that: $\mathcal{G} = (r_1, g_1, r_2, g_2, \ldots, r_n, g_n, r_{n+1})$. Our model assumes that (i) genes orientation is unknown, (ii) there are no duplicate genes, and (iii) the considered genomes share the same set of genes. In this way, we can assign each gene a unique value and map the gene sequence as a permutation $\boldsymbol{\pi = (\pi_1\ \pi_2\ \ldots\ \pi_n)}$, $\pi_i \in \mathbb{N}$, $1 \leq \pi_i \leq n$, and $\pi_i \neq \pi_j$ for all $i \neq j$.

Rearrangement events can split intergenic regions, and each intergenic region has a well-defined amount of nucleotides. Thus, we represent it by the size (amount of nucleotides). The sequence of intergenic regions $\boldsymbol{\breve{\pi} = (\breve{\pi}_1\ \breve{\pi}_2\ \ldots\ \breve{\pi}_{n+1})}$, $\breve{\pi}_i \in \mathbb{N}$, represents the respective intergenic region sizes, such that $\breve{\pi}_i$ is on the left side of π_i, whereas $\breve{\pi}_{i+1}$ is on the right side.

Since we represent genes as a permutation π, we can treat the problem as a sorting problem in which the target permutation is the identity ι. This approach is widely used and it means that if we are able to transform $(\pi, \breve{\pi})$ into $(\iota, \breve{\iota})$ we can also transform $(\alpha, \breve{\alpha})$ into $(\sigma, \breve{\sigma})$. Therefore, an **instance** of our problem is composed by three elements $(\pi, \breve{\pi}, \breve{\iota})$ and the **rearrangement distance** considering a model $\mathcal{M}$ is represented as $d_{\mathcal{M}}(\pi, \breve{\pi}, \breve{\iota})$.

Definition 1. *An intergenic reversal* $\rho_{(x,y)}^{(i,j)}$*, with* $1 \leq i \leq j \leq n$*,* $0 \leq x \leq \breve{\pi}_i$*,* $0 \leq y \leq \breve{\pi}_{j+1}$*, and* $\{x, y\} \subset \mathbb{N}$*, splits the intergenic regions* $\breve{\pi}_i$ *and* $\breve{\pi}_{j+1}$ *into pieces with sizes* $(x, x' = \breve{\pi}_i - x)$ *and* $(y, y' = \breve{\pi}_{j+1} - y)$*, respectively. The sequence* $(x', \pi_i, \ldots, \pi_j, y)$ *is inverted and the pieces* (x, y) *and* (x', y') *form new intergenic regions* $\breve{\pi}_i$ *and* $\breve{\pi}_{j+1}$*, respectively.*

Definition 2. *An intergenic transposition* $\tau_{(x,y,z)}^{(i,j,k)}$*, with* $1 \leq i < j < k \leq n+1$*,* $0 \leq x \leq \breve{\pi}_i$*,* $0 \leq y \leq \breve{\pi}_j$*,* $0 \leq z \leq \breve{\pi}_k$*, and* $\{x, y, z\} \subset \mathbb{N}$*, splits the intergenic regions* $\breve{\pi}_i$*,* $\breve{\pi}_j$*, and* $\breve{\pi}_k$ *into pieces with sizes* $(x, x' = \breve{\pi}_i - x)$*,* $(y, y' = \breve{\pi}_j - y)$*, and* $(z, z' = \breve{\pi}_k - z)$*, respectively. The sequences* $(x', \pi_i, \ldots, y)$ *and* $(y', \pi_j, \ldots, z)$ *swap positions without changing orientation, and pieces* (x, y')*,* (z, x')*, and* (y, z') *form new intergenic regions* $\breve{\pi}_i$*,* $\breve{\pi}_{k+i-j}$*, and* $\breve{\pi}_k$*, respectively.*

Figures 1(a) and (b) show a generic example of intergenic reversal and intergenic transposition, respectively.

Definition 3. *An intergenic insertion* ϕ_x^i*, such that* $1 \leq i \leq (n+1)$*,* $x > 0$*, and* $x \in \mathbb{N}$ *acts on the intergenic region* $\breve{\pi}_i$ *inserting an amount* x *of nucleotides.*

Definition 4. *An intergenic deletion* ψ_x^i*, such that* $1 \leq i \leq (n+1)$*,* $0 < x \leq \breve{\pi}_i$*, and* $x \in \mathbb{N}$ *acts on the intergenic region* $\breve{\pi}_i$ *removing an amount* x *of nucleotides.*

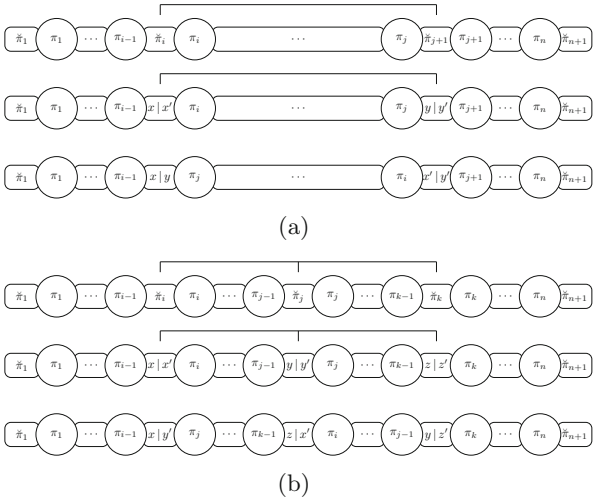

Fig. 1. The figure illustrates two intergenic genome rearrangement operations, the reversal (a) and transposition (b).

From now on, we will refer to the operations simply as a reversal, transposition, insertion, and deletion. Since reversal and transposition do not insert or remove genetic material, when dealing solely with conservative events the sum of intergenic regions size in both genomes is the same.

Definition 5. *Given a permutation π with n elements, the extended permutation π' is a permutation with two new elements $\pi'_0 = 0$ and $\pi'_{n+1} = n + 1$.*

From now on, we use the term permutation to denote the extended permutation. We say that an intergenic region is of (in)correct size if it occurs between two elements consecutive both in π' and ι' and whose size is (not) the same in π' and ι'.

Definition 6. *An intergenic breakpoint is a pair of elements π'_i and π'_{i+1} of the permutation π', such that either they are not consecutive in the identity permutation ι', or they are consecutive and the intergenic region between them has an incorrect size.*

An *intergenic breakpoint* represents a region that at some point has to be affected by some operation either to fix the gene order or the size of the intergenic region. Figure 2 shows examples of intergenic breakpoints.

Definition 7. *The number of intergenic breakpoints is denoted as $ib(\pi', \breve{\pi}, \breve{\iota})$.*

Remark 1. The instance $(\iota', \breve{\iota}, \breve{\iota})$ has a property that does not occur in any other instance: $ib(\iota', \breve{\iota}, \breve{\iota}) = 0$.

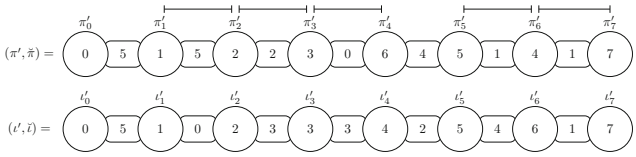

Fig. 2. Given the permutation $\pi' = (0, 1, 2, 3, 6, 5, 4, 7)$ and the size of the intergenic regions $\breve{\pi} = (5, 5, 2, 0, 4, 1, 1)$ and $\breve{\iota} = (5, 0, 3, 3, 2, 4, 1)$ we have the following intergenic breakpoints: (π'_1, π'_2), (π'_2, π'_3), (π'_3, π'_4), (π'_5, π'_6), and (π'_6, π'_7). Note that the elements of the intergenic breakpoint (π'_3, π'_4) are not consecutive on the identity permutation while the elements of the intergenic breakpoint (π'_2, π'_3) are, but the intergenic region between the elements has an incorrect size.

Definition 8. *The variation in the number of intergenic breakpoints after applying a sequence of operations S is denoted as $\Delta_{ib}(\pi', \breve{\pi}, \breve{\iota}, S)$, such that:*

$$\Delta_{ib}(\pi', \breve{\pi}, \breve{\iota}, S) = ib((\pi', \breve{\pi}, \breve{\iota}) \cdot S) - ib(\pi', \breve{\pi}, \breve{\iota}).$$

Definition 9. *A pair (a, b) is a block in a permutation π' if $|a - b| = 1$, the elements a and b are consecutive in the permutation π', and the intergenic region between them has a correct size.*

Definition 10. *Two intergenic breakpoints (π'_i, π'_{i+1}) and (π'_j, π'_{j+1}), such that $i < j$, are connected if the following conditions are fulfilled:*

i. At least one of the pairs (π'_i, π'_{i+1}), (π'_j, π'_{j+1}), (π'_i, π'_j), (π'_i, π'_{j+1}), (π'_{i+1}, π'_j), or (π'_{i+1}, π'_{j+1}) corresponds to two consecutive elements in the identity permutation ι' that do not form a block in the permutation π'.

ii. $\breve{\pi}_{i+1} + \breve{\pi}_{j+1} \geq \breve{\iota}_k$, such that $\breve{\iota}_k$ is the size of the intergenic region between the pair of consecutive elements in the identity permutation ι'.

Connected intergenic breakpoints represent regions that have the potential to remove at least one intergenic breakpoint by placing two consecutive elements and fixing the size of the intergenic region between them. For example, in Fig. 2, the intergenic breakpoints (π'_3, π'_4) and (π'_6, π'_7) are connected while the intergenic breakpoints (π'_2, π'_3) and (π'_3, π'_4) are not.

3 Results

We start this section by showing that problems SbIR and SbIRID belong to the problem class NP-hard. For this, we used a reduction of Sorting by Reversals problem, which does not consider intergenic regions. Then, we present lower bounds (Subsect. 3.1) and approximation algorithms (Subsect. 3.2) for each of the variations of the problems addressed in this work.

The Sorting by Reversals problem (SbR) has already been proven NP-hard [6]. An instance of this problem consists of a permutation δ and a natural number d. The goal is to determine if its possible to transform δ into ι applying at most d reversals.

Lemma 1. *SbIR problem is NP-hard.*

Proof. We can reduce all instances of SbR to instances of SbIR by setting $\pi = \delta$ and $\breve{\pi} = \breve{\iota} = (0\ 0\ ...\ 0)$. Note that it is possible to transform δ into ι applying at most d reversals if and only if $d_{SbIR}(\pi, \breve{\pi}, \breve{\iota}) \leq d$. □

Lemma 2. *SbIRID problem is NP-hard.*

Proof. We can reduce all instances of SbR to instances of SbIRID by setting $\pi = \delta$ and $\breve{\pi} = \breve{\iota} = (0\ 0\ ...\ 0)$. Note that for these instances of the SbIRID problem no insertion and deletion will be applied, otherwise we could get a smaller sequence of reversals just by ignoring the insertions and deletions. That way, it is possible to transform δ into ι applying at most d reversals if and only if $d_{SbIRID}(\pi, \breve{\pi}, \breve{\iota}) \leq d$. □

3.1 Lower Bounds

Following lemmas present lower bounds for each problem we consider.

Lemma 3. $\Delta_{ib}(\pi', \breve{\pi}, \breve{\iota}, \rho) \geq -2$ *for any reversal* ρ.

Proof. Suppose that (π'_{i-1}, π'_i) and (π'_j, π'_{j+1}) are intergenic breakpoints. In this case, the best scenario after applying reversal $\rho^{(i,j)}_{(x,y)}$ removes the intergenic breakpoints (π'_{i-1}, π'_i) and (π'_j, π'_{j+1}), reducing the number of intergenic breakpoints by two. Since any reversal only affects the neighborhood of two pairs of genes and two intergenic regions it is impossible to remove more than two intergenic breakpoints. □

Lemma 4. $\Delta_{ib}(\pi', \breve{\pi}, \breve{\iota}, \tau) \geq -3$ *for any transposition* τ.

Proof. Suppose that (π'_{i-1}, π'_i), (π'_{j-1}, π'_j), and (π'_{k-1}, π'_k) are intergenic breakpoints. In this case, the best scenario is a transposition $\tau^{(i,j,k)}_{(x,y,z)}$ that removes the intergenic breakpoints (π'_{i-1}, π'_i), (π'_{j-1}, π'_j), and (π'_{k-1}, π'_k), reducing the number of intergenic breakpoints by three. Since any transposition only affects the neighborhood of three pairs of genes and three intergenic regions it is impossible to remove more than three intergenic breakpoints. □

Lemma 5. $\Delta_{ib}(\pi', \breve{\pi}, \breve{\iota}, \phi) \geq -1$ *for any insertion* ϕ.

Proof. As an insertion acts in just one intergenic region this means that the best scenario is to remove the intergenic breakpoint (π'_{i-1}, π'_i) after applying an insertion ϕ^i_x, reducing by one the number of intergenic breakpoints. □

Lemma 6. $\Delta_{ib}(\pi', \breve{\pi}, \breve{\iota}, \psi) \geq -1$ *for any deletion* ψ.

Proof. As a deletion acts in just one intergenic region this means that the best scenario is to remove the intergenic breakpoint (π'_{i-1}, π'_i) after applying a deletion ψ^i_x, reducing by one the number of intergenic breakpoints. □

Theorem 1. $d_{SbIR}(\pi, \breve{\pi}, \breve{\iota}) \geq \frac{ib(\pi', \breve{\pi}, \breve{\iota})}{2}$.

Proof. By the Remark 1, we know that $(\iota', \breve{\iota}, \breve{\iota})$ is the only instance with no intergenic breakpoints. To achieve the identity permutation and to fix the intergenic region sizes we need to remove $ib(\pi', \breve{\pi}, \breve{\iota})$ intergenic breakpoints. Also, by Lemma 3, a reversal removes at most two intergenic breakpoints and lemma follows. □

Theorem 2. $d_{SbIRID}(\pi, \breve{\pi}, \breve{\iota}) \geq \frac{ib(\pi', \breve{\pi}, \breve{\iota})}{2}$.

Proof. Directly by Lemmas 5 and 6, and Theorem 1. □

Theorem 3. $d_{SbIRT}(\pi, \breve{\pi}, \breve{\iota}) \geq \frac{ib(\pi', \breve{\pi}, \breve{\iota})}{3}$.

Proof. Directly by Remark 1 and Lemmas 3 and 4. □

Theorem 4. $d_{SbIRTID}(\pi, \breve{\pi}, \breve{\iota}) \geq \frac{ib(\pi', \breve{\pi}, \breve{\iota})}{3}$.

Proof. Directly by Lemmas 5 and 6, and Theorem 3. □

3.2 Approximation Algorithms

In this subsection, we will present four approximation algorithms. Initially, we will show an algorithm with approximation factor 4 for the SbIR and SbIRID problems. Then, we will present an algorithm with approximation factor 6 for the SbIRT and SbIRTID problems.

Lemma 7. *Let $(\pi, \breve{\pi}, \breve{\iota})$ be an instance such that $\sum_{i=1}^{n+1} \breve{\pi}_i \geq \sum_{i=1}^{n+1} \breve{\iota}_i$ and the number of intergenic breakpoints is greater than one. It is always possible to find at least one pair of intergenic breakpoints that are connected.*

Proof. Since $ib(\pi', \breve{\pi}, \breve{\iota}) > 1$, we can find at least a pair of intergenic breakpoints. We have to show that at least one of those pairs will be connected. Suppose that exists an instance $(\pi, \breve{\pi}, \breve{\iota})$, such that $\sum_{i=1}^{n+1} \breve{\pi}_i \geq \sum_{i=1}^{n+1} \breve{\iota}_i$, $ib(\pi', \breve{\pi}, \breve{\iota}) > 1$, and there is not a pair of intergenic breakpoints that are connected. The possibilities for not finding such a pair of intergenic breakpoints are:

- For all pairs of intergenic breakpoints (π'_i, π'_{i+1}) and (π'_j, π'_{j+1}) the elements (π'_i, π'_{i+1}), (π'_j, π'_{j+1}), (π'_i, π'_j), (π'_i, π'_{j+1}), (π'_{i+1}, π'_j), and (π'_{i+1}, π'_{j+1}) are not consecutive in the identity permutation, but if it is true π cannot be a permutation.
- For all pairs of intergenic breakpoints (π'_i, π'_{i+1}) and (π'_j, π'_{j+1}) we do not have enough intergenic material to remove any intergenic breakpoint $\breve{\pi}_{i+1} + \breve{\pi}_{j+1} < \breve{\iota}_k$, such that $\breve{\iota}_k$ is the size of the intergenic region between the pair of consecutive elements in the identity permutation. If it is true it implies that $\sum_{i=1}^{n+1} \breve{\pi}_i < \sum_{i=1}^{n+1} \breve{\iota}_i$ and that contradicts the initial assumption. □

Lemma 8. *Let (π'_i, π'_{i+1}) and (π'_j, π'_{j+1}) be intergenic breakpoints that are connected. It is possible to remove at least one intergenic breakpoint after at most two reversals.*

Proof. We analyze the possibilities to remove an intergenic breakpoint based on a pair of connected intergenic breakpoints.

i. (π'_i, π'_j) or (π'_{i+1}, π'_{j+1}) are consecutive in the identity permutation: these cases are symmetric and we need to apply only one reversal $\rho^{(i+1,j)}_{(x,y)}$ to place the element π'_j on the right side of the element π'_i or π'_{i+1} on the left side of π'_{j+1}. As $\breve{\pi}_{i+1} + \breve{\pi}_{j+1} \geq \breve{\iota}_k$, then the x and y parameters can always be chosen properly to fill the intergenic region with the correct size between the consecutive elements generated (Fig. 3(a)).

ii. (π'_i, π'_{j+1}): in this case we apply two consecutive reversals. In this scenario, we need an intergenic breakpoint (π'_k, π'_{k+1}), such that $k < i$ or $k > j$, to apply the sequence of reversals without creating new intergenic breakpoints. We will prove that exists such intergenic breakpoint by contradiction. Suppose that there is no intergenic breakpoint (π'_k, π'_{k+1}) such that $k < i$ or $k > j$. This means that the segments $(\pi'_0, \ldots, \pi'_i)$ and $(\pi'_{j+1}, \ldots, \pi'_{n+1})$ are composed of consecutive elements with no intergenic breakpoint between them; also we know that π'_i and π'_{j+1} are consecutive elements, but if both statements are true it implies that there are no valid values for the elements π_{i+1} and π_j of the permutation π. If $k < i$ we apply a reversal $\rho^{(k+1,i)}_{(0,\breve{\pi}_{i+1})}$ to obtain the case (i) (Fig. 3(b)). If $k > j$ we apply a reversal $\rho^{(j+1,k)}_{(0,\breve{\pi}_{k+1})}$ to obtain the case (i) (Fig. 3(c)). Note that in both scenarios the intergenic regions sizes remains the same and the case (i) can be applied (Fig. 3(b)).

iii. (π'_{i+1}, π'_j): In this case we apply two consecutive reversals. In this scenario, we need an intergenic breakpoint (π'_k, π'_{k+1}), such that $k > i$ and $k < j$, to apply the sequence of reversals without creating new intergenic breakpoints. We will prove that exists such intergenic breakpoint by contradiction. Suppose that there is no intergenic breakpoint (π'_k, π'_{k+1}) such that $k > i$ and $k < j$. This means that the segment $(\pi'_{i+1}, \ldots, \pi'_j)$ is composed of consecutive elements with no intergenic breakpoint between them; also we know that (π'_{i+1}, π'_j) are consecutive elements, but if both statements are true implies that there are no valid values for the elements π_{i+1} and π_j of the permutation π. After identifying the intergenic breakpoint (π'_k, π'_{k+1}) we apply a reversal $\rho^{(i+1,k)}_{(0,\breve{\pi}_{k+1})}$ (Fig. 3(d)) to obtain the case (i).

iv. (π'_i, π'_{i+1}) or (π'_j, π'_{j+1}): these cases are symmetric and we need to apply two consecutive reversals. Initially, we apply a reversal $\rho^{(i+1,j)}_{(0,\breve{\pi}_{j+1})}$ without changing the intergenic regions sizes, as result we obtain the case (i) (Fig. 3(e)). □

Theorem 5. *SbIR problem is 4-approximable.*

Proof. While the permutation is not sorted and while the permutation has intergenic regions with incorrect size, it is always possible to remove at least one intergenic breakpoint after applying at most two reversals (Lemmas 7 and 8). In the worst case, it gives us a total of $2ib(\pi', \breve{\pi}, \breve{\iota})$ reversals to transform $(\pi, \breve{\pi}, \breve{\iota})$ into $(\iota, \breve{\iota}, \breve{\iota})$. By the Theorem 1, we obtained the lower bound $ib(\pi', \breve{\pi}, \breve{\iota})/2$ and the theorem follows. □

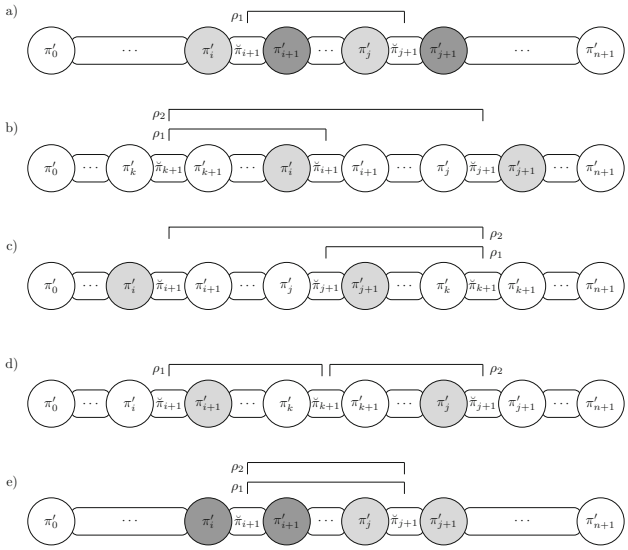

Fig. 3. The possibilities that can be found when we have a pair of connected intergenic breakpoints and the operations of reversal that must be applied to remove at least one intergenic breakpoint. The pair of elements that are consecutive in the identity permutation are represented with a gray scale color.

Lemma 9. *Let* $(\pi, \breve{\pi}, \breve{\iota})$ *be an instance of the Sorting by Intergenic Reversals, Insertions, and Deletions problem, such that* $ib(\pi', \breve{\pi}, \breve{\iota}) > 0$*. It is always possible to find an insertion* ϕ *such that* $\Delta_{ib}(\pi', \breve{\pi}, \breve{\iota}, \phi) \leq 0$*.*

Proof. Since $ib(\pi', \breve{\pi}, \breve{\iota}) > 0$, then it exists at least one intergenic breakpoint that we can apply an insertion in this region. Therefore, in the worst case the amount of intergenic breakpoints remains the same. □

Lemma 10. *Let* $(\pi', \breve{\pi}, \breve{\iota})$ *be an instance of the Sorting by Intergenic Reversals, Insertions, and Deletions problem, such that* $ib(\pi', \breve{\pi}, \breve{\iota}) = 1$ *and* $\sum_{i=1}^{n+1} \breve{\pi}_i > \sum_{i=1}^{n+1} \breve{\iota}_i$*. It is always possible to find a deletion* ψ *such that* $\Delta_{ib}(\pi', \breve{\pi}, \breve{\iota}, \psi) = -1$*.*

Proof. Since $ib(\pi', \breve{\pi}, \breve{\iota}) = 1$, then we know that $\pi' = \iota'$, otherwise the number of intergenic breakpoints should be greater than one. Since all the elements of the permutation π' are consecutive and $\sum_{i=1}^{n+1} \breve{\pi}_i > \sum_{i=1}^{n+1} \breve{\iota}_i$, there is an intergenic region $\breve{\pi}_k$, such as $\breve{\pi}_k > \breve{\iota}_k$. Thus the deletion $\psi^k_{\breve{\iota}_k - \breve{\pi}_k}$ removes the intergenic breakpoint (π'_{k-1}, π'_k) and the lemma follows. □

Theorem 6. *SbIRID problem is 4-approximable.*

Proof. We are going to divide the proof into three cases:

i. $\sum_{i=1}^{n+1} \breve{\pi}_i > \sum_{i=1}^{n+1} \breve{\iota}_i$: Lemmas 7 and 8 remain valid as long as $ib(\pi', \breve{\pi}, \breve{\iota}) > 1$, then we apply only one deletion to remove the last intergenic breakpoint (Lemma 10).

ii. $\sum_{i=1}^{n+1} \breve{\pi}_i = \sum_{i=1}^{n+1} \breve{\iota}_i$: Lemmas 7 and 8 are sufficient to sort the permutation and to fix the sizes of the intergenic regions by applying only reversals.
iii. $\sum_{i=1}^{n+1} \breve{\pi}_i < \sum_{i=1}^{n+1} \breve{\iota}_i$: Initially we apply an insertion to make $\sum_{i=1}^{n+1} \breve{\pi}_i = \sum_{i=1}^{n+1} \breve{\iota}_i$ (Lemma 9). Sequentially, Lemmas 7 and 8 guarantee that the permutation will be sorted and the sizes of the intergenic regions will be fixed applying only reversals. Note that it is not guaranteed that the initial insertion removes any intergenic breakpoint, but then only reversals are applied and the last reversal must remove two intergenic breakpoints. Considering the insertion and the last reversal, on average, we were able to remove one intergenic breakpoint after applying one operation.

Considering the three cases, in the worst scenario, it gives us a total of $2ib(\pi', \breve{\pi}, \breve{\iota})$ operations to transform $(\pi, \breve{\pi}, \breve{\iota})$ into $(\iota, \breve{\iota}, \breve{\iota})$. By Theorem 2, we obtained the lower bound $ib(\pi', \breve{\pi}, \breve{\iota})/2$ and the theorem follows. □

Lemma 11. *Let (π'_i, π'_{i+1}) and (π'_j, π'_{j+1}) be intergenic breakpoints that are connected. It is possible to remove at least one intergenic breakpoint after at most two operations of reversal or transposition.*

Proof. Similar to Lemma 8, we will analyze the possibilities to remove an intergenic breakpoint based on a pair of connected intergenic breakpoints.

i. (π'_i, π'_j) or (π'_{i+1}, π'_{j+1}): We need to apply only one reversal, which is exactly as the procedure shown in case (i) of Lemma 8.
ii. (π'_i, π'_{j+1}): In this case we need to apply one transposition. As shown previously in case (ii) of Lemma 8, we know that it must exist another intergenic breakpoint (π'_k, π'_{k+1}) such that $k < i$ or $k > j$. If $k < i$ we apply a transposition $\tau_{(x,y,z)}^{(k+1,i+1,j+1)}$ to place the element π'_i on the left side of the element π'_{j+1} (Fig. 4(a)). If $k > j$ we apply a transposition $\tau_{(x,y,z)}^{(i+1,j+1,k+1)}$ to place the element π'_{j+1} on the right side of the element π'_i (Fig. 4(b)). In both scenarios, we have that $\breve{\pi}_{i+1} + \breve{\pi}_{j+1} \geq \breve{\iota}_k$, then the x, y, and z parameters always can be chosen properly to fill the intergenic region with the correct size between the consecutive elements generated.
iii. (π'_{i+1}, π'_j): In this case we need to apply one transposition. As shown previously in case (iii) of Lemma 8, we know that it must exist another intergenic breakpoint (π'_k, π'_{k+1}) such that $k > i$ and $k < j$. After identifying the intergenic breakpoint (π'_k, π'_{k+1}) we apply a transposition $\tau_{(x,y,z)}^{(i+1,k+1,j+1)}$ to place the element π'_j on the left side of the element π'_{i+1} (Fig. 4(c)). Since $\breve{\pi}_{i+1} + \breve{\pi}_{j+1} \geq \breve{\iota}_k$, then the x, y and z parameters always can be chosen properly to fill the intergenic region with the correct size between the consecutive elements generated.
iv (π'_i, π'_{i+1}) or (π'_j, π'_{j+1}): We need to apply two reversals exactly as the procedure shown on Lemma 8 case (iv). □

Theorem 7. *SbIRT problem is 6-approximable.*

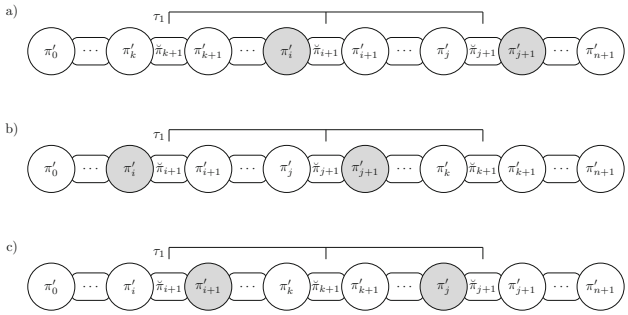

Fig. 4. The possibilities to remove at least one intergenic breakpoint by applying only one transposition. The pair of elements that are consecutive in the identity permutation are represented with the color gray.

Proof. While the permutation is not sorted and with all the intergenic regions with the correct size it is always possible to remove at least one intergenic breakpoint after applying at most two operations (Lemmas 7 and 11). In the worst case, it gives us a total of $2ib(\pi', \breve{\pi}, \breve{\iota})$ operations to transform $(\pi, \breve{\pi}, \breve{\iota})$ into $(\iota, \breve{\iota}, \breve{\iota})$. By Theorem 3, we obtained the lower bound $ib(\pi', \breve{\pi}, \breve{\iota})/3$ and the theorem follows. □

Theorem 8. *SbIRTID problem is 6-approximable.*

Proof. Similar to Theorem 6, we are going to divide the analysis into three cases:

i. $\sum_{i=1}^{n+1} \breve{\pi}_i > \sum_{i=1}^{n+1} \breve{\iota}_i$: Lemmas 7 and 11 remain valid as long as $ib(\pi', \breve{\pi}, \breve{\iota}) > 1$, then we apply only one deletion to remove the last intergenic breakpoint (Lemma 10).
ii. $\sum_{i=1}^{n+1} \breve{\pi}_i = \sum_{i=1}^{n+1} \breve{\iota}_i$: Lemmas 7 and 11 are sufficient to sort the permutation and to fix the sizes of the intergenic regions by applying only reversals and transpositions.
iii. $\sum_{i=1}^{n+1} \breve{\pi}_i < \sum_{i=1}^{n+1} \breve{\iota}_i$: Initially we apply an insertion to make $\sum_{i=1}^{n+1} \breve{\pi}_i = \sum_{i=1}^{n+1} \breve{\iota}_i$ (Lemma 9). Sequentially, Lemmas 7 and 11 guarantee that the permutation will be sorted and the sizes of the intergenic regions will be fixed applying only reversals and transpositions. Note that it is not guaranteed that the initial insertion removes any intergenic breakpoint, but then only reversals and transpositions are applied and the last operation (reversal or transposition) must remove at least two intergenic breakpoints. Considering the insertion and the last operation, on average, we were able to remove one intergenic breakpoint after applying one operation.

Considering the three cases, in the worst scenario, it gives us a total of $2ib(\pi', \breve{\pi}, \breve{\iota})$ operations to transform $(\pi, \breve{\pi}, \breve{\iota})$ into $(\iota, \breve{\iota}, \breve{\iota})$. By Theorem 4, we obtained the lower bound $ib(\pi', \breve{\pi}, \breve{\iota})/3$, and the theorem follows. □

4 Conclusion

We proved that the problems of Sorting by Intergenic Reversals and Sorting by Intergenic Reversals, Insertions, and Deletions belong to NP-hard problems class. Besides, we presented for both problems an algorithms with approximation factor 4. We also investigate the Sorting by Intergenic Reversals and Transpositions problem and the variation with non-conservative events of insertion and deletion. For both problems, we designed approximation algorithms of factor 6.

As future works, we intend to improve the approximation factors of the algorithms and develop cost functions that consider the likelihood of each operation.

Acknowledgments. This work was supported by the National Council for Scientific and Technological Development - CNPq (grants 400487/2016-0, 425340/ 2016-3, and 140466/2018-5), the São Paulo Research Foundation - FAPESP (grants 2013/08293-7, 2015/ 11937-9, 2017/12646-3, and 2017/16246-0), the Brazilian Federal Agency for the Support and Evaluation of Graduate Education - CAPES, and the CAPES/COFECUB program (grant 831/15).

References

1. Berman, P., Hannenhalli, S., Karpinski, M.: 1.375-approximation algorithm for sorting by reversals. In: Möhring, R., Raman, R. (eds.) ESA 2002. LNCS, vol. 2461, pp. 200–210. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45749-6_21
2. Biller, P., Guéguen, L., Knibbe, C., Tannier, E.: Breaking good: accounting for fragility of genomic regions in rearrangement distance estimation. Genome Biol. Evol. **8**(5), 1427–1439 (2016)
3. Biller, P., Knibbe, C., Beslon, G., Tannier, E.: Comparative genomics on artificial life. In: Beckmann, A., Bienvenu, L., Jonoska, N. (eds.) CiE 2016. LNCS, vol. 9709, pp. 35–44. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-40189-8_4
4. Bulteau, L., Fertin, G., Rusu, I.: Sorting by transpositions is difficult. SIAM J. Comput. **26**(3), 1148–1180 (2012)
5. Bulteau, L., Fertin, G., Tannier, E.: Genome rearrangements with indels in intergenes restrict the scenario space. BMC Bioinform. **17**(14), 426 (2016)
6. Caprara, A.: Sorting permutations by reversals and eulerian cycle decompositions. SIAM J. Discrete Math. **12**(1), 91–110 (1999)
7. Elias, I., Hartman, T.: A 1.375-approximation algorithm for sorting by transpositions. IEEE/ACM Trans. Comput. Biol. Bioinform. **3**(4), 369–379 (2006)
8. Fertin, G., Jean, G., Tannier, E.: Algorithms for computing the double cut and join distance on both gene order and intergenic sizes. Algorithms Mol. Biol. **12**(1), 16 (2017)
9. Oliveira, A.R., Jean, G., Fertin, G., Dias, U., Dias, Z.: Super short reversals on both gene order and intergenic sizes. In: Alves, R. (ed.) BSB 2018. LNCS, vol. 11228, pp. 14–25. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01722-4_2
10. Rahman, A., Shatabda, S., Hasan, M.: An approximation algorithm for sorting by reversals and transpositions. J. Discrete Algorithms **6**(3), 449–457 (2008)
11. Yancopoulos, S., Attie, O., Friedberg, R.: Efficient sorting of genomic permutations by translocation, inversion and block interchange. Bioinformatics **21**(16), 3340–3346 (2005)