Block Interchange and Reversal Distance on Unbalanced Genomes

Alexsandro Oliveira Alexandrino[1(✉)], Gabriel Siqueira[1], Klairton Lima Brito[1], Andre Rodrigues Oliveira[2], Ulisses Dias[3], and Zanoni Dias[1]

[1] Institute of Computing, University of Campinas, Campinas, Brazil
{alexsandro,gabriel.siqueira,klairton,zanoni}@ic.unicamp.br
[2] Computing and Informatics Department, Mackenzie Presbyterian University, São Paulo, Brazil
andre.rodrigues@mackenzie.br
[3] School of Technology, University of Campinas, Campinas, Brazil
ulisses@ft.unicamp.br

Abstract. One method for inferring the evolutionary distance between two organisms is to find the *rearrangement distance*, which is defined as the minimum number of genome rearrangements required to transform one genome into the other. Rearrangements that do not alter the genome content are known as conservative. Examples of such rearrangements include: *reversal*, which reverts a segment of the genome; *transposition*, which exchanges two consecutive blocks; *block interchange (BI)*, which exchanges two blocks at any position in the genome; and *double cut and join (DCJ)*, which cuts two different pairs of adjacent blocks and joins them in a different manner. Initially, works in this area involved comparing genomes that shared the same set of conserved blocks. Nowadays, researchers are investigating unbalanced genomes (genomes with a distinct set of genes), which requires the use of non-conservative rearrangements such as *insertions* and *deletions* (*indels*). In cases where there are no repeated blocks and the genomes have the same set of blocks, the BI Distance and the Reversal Distance have polynomial-time algorithms, while the complexity of the BI and Reversal Distance problem remains unknown. In this study, we investigate the BI and Indel Distance and the BI, Reversal, and Indel Distance on genomes with different gene content and no repeated genes. We present 2-approximation algorithms for each problem using a variant of the breakpoint graph structure.

Keywords: Block Interchange · Reversal · Unbalanced Genomes

This work was supported by the National Council of Technological and Scientific Development, CNPq (grants 140272/2020-8, 202292/2020-7 and 425340/2016-3), the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001, and the São Paulo Research Foundation, FAPESP (grants 2013/08293-7, 2015/11937-9, 2019/27331-3, 2021/13824-8, and 2022/13555-0).

M. S. Reis and R. C. de Melo-Minardi (Eds.): BSB 2023, LNBI 13954, pp. 1–13, 2023.
https://doi.org/10.1007/978-3-031-42715-2_1

1 Introduction

Mutations play a significant role during the evolutionary process. When these mutations affect large stretches of a genome, they are called *genome rearrangements*. By analyzing the relative order of genes in genomes of related species, we can compute a sequence of rearrangements that transforms one genome into another. Based on the principle of parsimony, the scenario with the least number of rearrangements is assumed to be the most likely to have occurred.

The problem of finding the minimum number of rearrangements required to transform one genome into another, known as the rearrangement distance, is addressed using a model that defines which rearrangements should be considered. There are several genome rearrangement models, including conservative and non-conservative events. Conservative events, such as reversal, block interchange (BI), transposition, and double cut and join (DCJ), do not alter the amount of genetic material. In contrast, non-conservative events, such as insertion and deletion, add or remove genetic material at specific positions in the genome.

The computation of the rearrangement distance between two genomes can be accomplished in polynomial time for certain models, while for others, it is NP-hard. This depends on the level of information available, such as the orientation of genes in each genome. When gene orientations are considered, both the Reversal Distance and the DCJ Distance can be solved in polynomial time [12,16]. However, when orientations are not known, these distances become NP-hard, as demonstrated by previous studies [7,8,13].

Since block interchanges and transpositions change only the relative position of elements but not their orientations, they do not consider gene orientation [11]. The Block Interchange Distance has an exact polynomial time algorithm [9], while the Transposition Distance is NP-hard [6].

The literature on genome rearrangements started the study of the distance between unbalanced genomes (genomes with a distinct set of genes) in 2000 [10], and most of the models use *indels*, which refers to both insertions and deletions. Considering gene orientation, the DCJ and Indel Distance [5] and the Reversal and Indel Distance [15] are both solvable in polynomial time, while the Transposition and Indel Distance is NP-hard [1,2].

Here we study the Block Interchange and Indel Distance and the Block Interchange, Reversal, and Indel Distance, considering that genomes have a distinct set of genes, but there are no occurrences of repeated genes in a genome. We present lower bounds and 2-approximation algorithms for these problems.

2 Definitions

An instance for a rearrangement distance problem has a source genome $\mathcal{G}_1$ and a target genome $\mathcal{G}_2$. We represent the target genome $\mathcal{G}_2$ with the identity string $\iota^n = (+1\ +2\ \ldots\ +n)$, where each element ι_i^n maps a gene or a maximal continuous sequence of genes without correspondence in $\mathcal{G}_1$. We say that $1, 2, \ldots, n$

(without signs) are *labels*. We represent the source genome $\mathcal{G}_1$ with a string $A = (A_1\ A_2\ \ldots\ A_m)$, where A_i maps a gene, using the same mapping of labels and genes used for the target genome, or it represents a maximal continuous sequence of genes without correspondence in $\mathcal{G}_2$. If A_i maps a gene of $\mathcal{G}_1$, then it has a "+" sign if the gene with same label in $\mathcal{G}_2$ has the same orientation, and it has a "−" sign otherwise. For any element A_i that maps a continuous sequence of genes without correspondence in $\mathcal{G}_2$, we set $A_i = \alpha$ without any sign, since this element will be removed regardless of its content.

We use $-A_i$ to denote the element A_i with its orientation reversed. For example, if $A_i = -1$, then $-A_1 = +1$. For the models where gene orientation is not considered, as the Block Interchange and Indel Distance, we can just omit the signs or consider that every element has a "+" sign.

The *alphabet* Σ_σ of a string σ is the set of labels present in σ. Note that $\Sigma_A \setminus \Sigma_{\iota^n} = \{\alpha\}$. Furthermore, there are no adjacent elements in ι^n such that both of them belong to $\Sigma_{\iota^n} \setminus \Sigma_A$, since any maximal continuous segment of genes without correspondence in $\mathcal{G}_1$ are mapped into a single element in ι^n. For the strings $A = (+6\ \alpha\ -3\ +4\ +1\ \alpha)$ and $\iota^6 = (+1\ +2\ +3\ +4\ +5\ +6)$, we have $\Sigma_A \cap \Sigma_{\iota^6} = \{1, 3, 4, 6\}$, $\Sigma_A \setminus \Sigma_{\iota^6} = \{\alpha\}$, $\Sigma_{\iota^6} \setminus \Sigma_A = \{2, 5\}$.

Given a string A with $|A| = m$, a block interchange $\mathcal{BI}(i, j, k, l)$, with $1 \leq i \leq j < k \leq l \leq m$, is a rearrangement that acts on the segments $(A_i\ \ldots\ A_j)$ and $(A_k\ \ldots\ A_l)$ generating the string $A \cdot \mathcal{BI}(i, j, k, l) = (A_1\ \ldots\ A_{i-1}\ \underline{A_k\ \ldots\ A_l}\ A_{j+1}\ \ldots\ A_{k-1}\ \underline{A_i\ \ldots\ A_j}\ A_{l+1}\ \ldots\ A_m)$.

Given a string A with $|A| = m$, a reversal $\rho(i, j)$, with $1 \leq i \leq j \leq m$, inverts the segment $(A_i\ \ldots\ A_j)$ and changes the orientation of the elements in it. It generates the string $A \cdot \rho(i, j) = (A_1\ \ldots\ A_{i-1}\ \underline{-A_j\ \ldots\ -A_i}\ A_{j+1}\ \ldots\ A_m)$.

Given a string A with $|A| = m$, an insertion $\phi(i, S)$, where $0 \leq i \leq m$ and S is a string, is a rearrangement which inserts S in the position $i + 1$ of a string. When applied to A, we have $A \cdot \phi(i, S) = (A_1\ \ldots\ A_i\ \underline{S_1\ \ldots\ S_{|S|}}\ A_{i+1}\ \ldots\ A_m)$.

Given a string A with $|A| = m$, a deletion $\psi(i, j)$, with $1 \leq i \leq j \leq m$, removes the segment $(A_i\ \ldots\ A_j)$ from the string A. When applied to A, we have $A \cdot \psi(i, j) = (A_1\ \ldots\ A_{i-1}\ A_{j+1}\ \ldots\ A_m)$.

A rearrangement model $\mathcal{M}$ defines the set of allowed rearrangements to compute the distance in a rearrangement distance problem. Given an instance (A, ι^n), the distance $d_{\mathcal{M}}(A, \iota^n)$ is the minimum number of operations in $\mathcal{M}$ that transforms A into ι^n. Since both models studied in this paper have indels, we chose not to mention it in the model acronym, so we use $d_{\mathcal{BI}}(A, \iota^n)$ and $d_{\rho,\mathcal{BI}}(A, \iota^n)$ for the Block Interchange and Indel Distance, and the Block Interchange, Reversal, and Indel Distance, respectively.

2.1 Labeled Cycle Graph

The Labeled Cycle Graph [2,14] is an adaptation of the breakpoint graph and the cycle graph created to deal with unbalanced genomes.

Given an instance (A, ι^n), we create the strings $\pi^A = (\pi^A_1\ \ldots\ \pi^A_{n'})$ and $\pi^\iota = (\pi^\iota_1\ \ldots\ \pi^\iota_{n'})$ as copies of A and ι^n, respectively, but removing elements

that do not belong to the set $\Sigma_A \cap \Sigma_{\iota^n}$. We extend both strings by adding the elements $\pi^A_0 = 0$, $\pi^\iota_0 = 0$, $\pi^A_{n'+1} = n+1$, and $\pi^\iota_{n'+1} = n+1$. We use $|\pi^A| = |\Sigma_A \cap \Sigma_{\iota^n}| = n'$ to denote the size of these strings without considering the extended elements 0 and $n+1$.

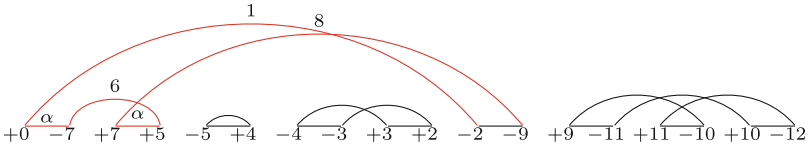

Fig. 1. Labeled Cycle Graph for the strings $A = (\alpha\ {+7}\ \alpha\ {-5}\ {-4}\ {+3}\ {-2}\ {+9}\ {+11}\ {+10})$ and ι^n, with $n = 11$. There are four cycles in this graph. The cycle $C_1 = (6, 1, 2)$ is a divergent cycle with $\Lambda(C_1) = 4$. All the other cycles have 0 runs. The cycle $C_2 = (3)$ is a trivial cycle. The cycle $C_3 = (5, 4)$ is a divergent cycle. The cycle $C_4 = (9, 7, 8)$ is an oriented cycle.

The Labeled Cycle Graph for (A, ι^n) is the undirected graph $G(A, \iota^n) = (V, E, \ell)$, where $V = \{+\pi^A_0, -\pi^A_1, +\pi^A_1, -\pi^A_2, +\pi^A_2, \ldots, -\pi^A_{n'}, +\pi^A_{n'}, -\pi^A_{n'+1}\}$ is the set of vertices; $E = E_s \cup E_t$ is the set of edges, which is divided into source (E_s) and target (E_t) edges; and ℓ is an edge labeling function.

Source edges connect vertices that are adjacent in π^A, while target edges connect vertices that are adjacent in π^ι. The set of source edges $E_s = \{e_i = (+\pi^A_{i-1}, -\pi^A_i) : 1 \le i \le n'+1\}$. A source edge $e_i = (+\pi^A_{i-1}, -\pi^A_i)$ has *index* i. The label $\ell(e_i) = \emptyset$ if π^A_{i-1} and π^A_i are consecutive in A. Otherwise, we have $\ell(e_i) = \alpha$. The set of target edges $E_t = \{e^t_i = (+\pi^\iota_{i-1}, -\pi^\iota_i) : 1 \le i \le n'+1\}$. A target edge $e^t_i = (+\pi^\iota_{i-1}, -\pi^\iota_i)$ has *index* i. The label $\ell(e^t_i) = \emptyset$ if π^ι_{i-1} and π^ι_i are consecutive. Otherwise, the label $\ell(e^t_i) = \pi^\iota_{i-1} + 1$.

We say that an edge is clean if it has empty label; otherwise, we say that the edge is labeled.

Since there are exactly one source and one target edge incident to each vertex, there exists a unique decomposition of the graph into a collection of edge alternating cycles.

We draw the graph by arranging the vertices horizontally, following the order in which they appear in π^A. The source edges are displayed as horizontal lines while the target edges are shown as arcs. Edges that have a label are marked in red, and the label is placed above the edge. Figure 1 provides an illustration of this representation.

Each cycle C in $G(A, \iota^n)$ is denoted by the list of source edges indices that belong to C. For a cycle $C = (c_1, c_2, \ldots, c_k)$, we construct the list of indices starting with the rightmost source edge (i.e., $c_1 > c_i$, for all $1 < i \le k$) and traversing it from right to left.

A cycle with k source edges is called a k-cycle. A 1-cycle is called trivial. The number of cycles in $G(A, \iota^n)$ is denoted by $c(A, \iota^n)$. For a rearrangement β, we define $\Delta c(A, \iota^n, \beta) = (|\pi^A| + 1 - c(A, \iota^n)) - (|\pi^A \cdot \beta| + 1 - c(A \cdot \beta, \iota^n))$.

A cycle $C = (c_1, c_2, \ldots, c_k)$ is *oriented* if the values $(c_1, c_2, \ldots, c_k)$ do not form a decreasing sequence. Given a cycle $C = (c_1, c_2, \ldots, c_k)$, a source edge e_{c_i} is *convergent* if it is traversed from right to left, and it is *divergent* otherwise. Note that e_{c_1} is always convergent by our convention of how the cycle is traversed when listing indices. A pair of edges (c_i, c_j) is divergent if one of the source edges is divergent and the other is convergent. A cycle is divergent if it has at least one divergent source edge, and it is convergent otherwise.

A graph $G(A, \iota^n)$ has divergent cycles if, and only if, A has at least one element with "$-$" sign. Therefore, for the Block Interchange and Indel Distance there are only convergent cycles in the graph.

An *insertion run* is a maximal path that starts and ends with labeled target edges and has no labeled source edge. Similarly, a *deletion run* is a maximal path that starts and ends with labeled source edges and has no labeled target edge. The number of runs in a cycle C is given by $\Lambda(C)$.

The *indel potential* of a cycle C is a value of how much indels are necessary to turn $\Lambda(C) = 0$ without merging cycles or creating new cycles with runs in it. We define the indel potential of C as follows:

$$\lambda(C) = \begin{cases} \left\lceil \frac{\Lambda(C)+1}{2} \right\rceil, & \text{if } \Lambda(C) > 0 \\ 0, & \text{otherwise.} \end{cases}$$

We also denote by $\lambda(A, \iota^n)$ the sum of indel potentials of all cycles in $G(A, \iota^n)$, that is, $\lambda(A, \iota^n) = \sum_{C \in G(A,\iota^n)} \lambda(C)$. We also have $\Delta\lambda(A, \iota^n, \beta) = \lambda(A, \iota^n) - \lambda(A \cdot \beta, \iota^n)$, which denotes the change in the indel potential of the graph caused by a rearrangement.

Lemma 1 (Alexandrino *et al.* [2]). *For any deletion ψ and strings A and ι^n, we have that $\Delta c(A, \iota^n, \psi) + \Delta\lambda(A, \iota^n, \psi) \leq 1$.*

Lemma 2 (Alexandrino *et al.* [2]). *For any insertion ϕ and strings A and ι^n, we have that $\Delta c(A, \iota^n, \phi) + \Delta\lambda(A, \iota^n, \phi) \leq 1$.*

Lemma 3. *For any block interchange $\mathcal{BI}$ and strings A and ι^n, we have that $\Delta c(A, \iota^n, \mathcal{BI}) + \Delta\lambda(A, \iota^n, \mathcal{BI}) \leq 2$.*

Proof. We divide this proof according to the number of cycles affected by $\mathcal{BI}$ [9].

If $\mathcal{BI}$ affects four cycles C_1, C_2, C_3, and C_4, then it merges these cycles into two new cycles C_1' and C_2'. In the best scenario, two deletion runs and two insertion runs from C_1 and C_3 are merged in C_1'. Similarly, two deletion runs and two insertion runs from C_2 and C_4 are merged in C_2'. In this case, $\Lambda(C_1') = \Lambda(C_1) + \Lambda(C_3) - 2$ and $\Lambda(C_2') = \Lambda(C_2) + \Lambda(C_4) - 2$. Therefore, $\Delta\lambda(A, \iota^n, \mathcal{BI}) = 4$ and $\Delta c(A, \iota^n, \mathcal{BI}) + \Delta\lambda(A, \iota^n, \mathcal{BI}) = 2$. An example is presented in Fig. 2.

If $\mathcal{BI}$ affects three cycles C_1, C_2 and C_3, then it merges these cycles into a new cycle C'. Similarly to the previous case, in the best scenario, the number of runs decreases in four and $\Lambda(C') = \Lambda(C_1) + \Lambda(C_2) + \Lambda(C_3) - 4$. Therefore, $\Delta\lambda(A, \iota^n, \mathcal{BI}) = 4$ and $\Delta c(A, \iota^n, \mathcal{BI}) + \Delta\lambda(A, \iota^n, \mathcal{BI}) = 2$.

If $\mathcal{BI}$ affects two cycles C_1 and C_2, then it turns these cycles into two new cycles or into four new cycles. If it turns C_1 and C_2 into two new cycles C_1' and C_2', then, in the best scenario, the number of runs decreases in four and $\Lambda(C_1') = \Lambda(C_1) - 2$ and $\Lambda(C_2') = \Lambda(C_1) - 2$. Therefore, the indel potential decreases by one for each cycle, so $\Delta\lambda(A, \iota^n, \mathcal{BI}) = 2$, and $\Delta c(A, \iota^n, \mathcal{BI}) = 0$.

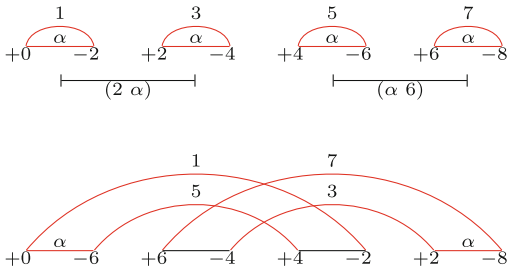

Fig. 2. Example of a block interchange that acts on four cycles. In this example, we have $A = (0\ \alpha\ 2\ \alpha\ 4\ \alpha\ 6\ \alpha\ 8)$ and $n = 7$. The indel potential of the original graph is equal to $4 \times \lceil (2+1)/2 \rceil = 8$ and the indel potential of the new graph is equal to $\lceil (2+1)/2 \rceil + \lceil (2+1)/2 \rceil = 4$.

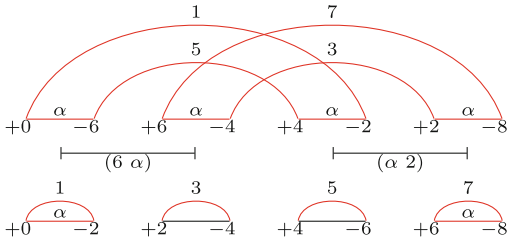

Fig. 3. Example of a block interchange that acts on two cycles creating four new cycles. In this example, we have $A = (0\ \alpha\ 6\ \alpha\ 4\ \alpha\ 2\ \alpha\ 8)$ and $n = 7$. The indel potential of the original graph is equal to $\lceil (4+1)/2 \rceil + \lceil (4+1)/2 \rceil = 6$ and the indel potential of the new graph is equal to $\lceil (2+1)/2 \rceil + \lceil (1+1)/2 \rceil + \lceil (1+1)/2 \rceil + \lceil (2+1)/2 \rceil = 6$.

If it affects two cycles C_1 and C_2, and it turns these cycles into four new cycles C_1', C_2', C_3', and C_4', then, in the best scenario, two pairs of deletion runs are merged, but note that each cycle has at least one insertion run, as shown in Fig. 3. So, $\Lambda(C_1') = X$, such that $1 \leq X < \Lambda(C_1)$, $\Lambda(C_2') = min(\Lambda(C_1) - X - 2, 1)$, $\Lambda(C_3') = Y$, such that $1 \leq Y < \Lambda(C_2)$, and $\Lambda(C_4') = min(\Lambda(C_2) - Y - 2, 1)$. Therefore, the indel potential of the graph remains the same.

If $\mathcal{BI}$ affects one cycle C_1, then it turns this cycle into a new cycle or into three new cycles. If it does not change the number of cycles, then, in the best scenario,

it can decrease the number of runs in the cycle by four and $\Delta\lambda(A, \iota^n, \mathcal{BI}) = 2$. If it turns this cycle into three new cycles C'_1, C'_2, and C'_3, then, in the best scenario, two pairs of deletion runs are merged, but note that each cycle has at least one insertion run. Similarly to the previous case, the indel potential of the graph remains the same. □

Lemma 4 (Willing *et al.* [14]). *For any reversal ρ and strings A and ι^n, we have that $\Delta c(A, \iota^n, \rho) + \Delta\lambda(A, \iota^n, \rho) \leq 1$.*

The graph $G(A, \iota^n)$ has only trivial cycles and indel potential of zero if, and only if, the strings $A = \iota^n$. Note that, when $A = \iota^n$, we have $|\pi^A| + 1 - c(A, \iota^n) + \lambda(A, \iota^n) = 0$.

Lemma 5. *For any strings A and ι^n, we have*

$$d_{\mathcal{BI}}(A, \iota^n) \geq \left\lceil \frac{|\pi^A| + 1 - c(A, \iota^n) + \lambda(A, \iota^n)}{2} \right\rceil.$$

Proof. Since $|\pi^{A'}| + 1 - c(A', \iota^n) + \lambda(A', \iota^n) = 0$ only if $A' = \iota^n$, a sequence of rearrangements that transform A into ι^n must decrease the value of $|\pi^A| + 1 - c(A, \iota^n) + \lambda(A, \iota^n)$ to zero. From Lemmas 1 to 3, a rearrangement can decrease this value by at most two and, therefore, the bound follows. □

Lemma 6. *For any strings A and ι^n, we have*

$$d_{\rho,\mathcal{BI}}(A, \iota^n) \geq \left\lceil \frac{|\pi^A| + 1 - c(A, \iota^n) + \lambda(A, \iota^n)}{2} \right\rceil.$$

Proof. Similar to the proof of Lemma 5 considering Lemmas 1 to 4. □

3 2-Approximation Algorithms for the Distance Problems

In this section, we introduce algorithms with approximation factors of 2 that use the graph structure presented in the previous section. Alexandrino *et al.* [2] presented a result on how to remove insertion runs from the graph and decrease the indel potential, but only considering unsigned strings. We present how this can be done for signed strings as well.

Lemma 7. *For any strings A and ι^n, if $G(A, \iota^n)$ has insertion runs, then there exists an insertion ϕ with $\Delta c(A, \iota^n, \phi) + \Delta\lambda(c, \iota^n, \phi) = 1$.*

Proof. Consider the insertion run $(v_1, v_2, \ldots, v_j)$ of a cycle C, such that v_1 has the same sign as the element of A that corresponds to v_1 and (v_1, v_2) is a labeled target edge. Let $o_1, o_2, \ldots, o_k$ be indices such that (v_{o_i}, v_{o_i+1}) is the i-th labeled target edge of this run.

We construct $S = (x_1, x_2, \ldots, x_k)$ as follows: for $1 \leq i \leq k$, if v_{o_i+1} has a "$-$" sign, then $x_i = \ell((v_{o_i}, v_{o_i+1}))$; otherwise, $x_i = -\ell((v_{o_i}, v_{o_i+1}))$. The insertion of S after the element of A corresponding to v_1 removes the run and adds k cycles in

the graph. A trivial cycle is created with the vertices $(v_1, -x_1)$. For each element x_i, with $1 \leq i < k$, there is a cycle $(+x_i, v_{o_i+1}, v_{o_i+2}, \ldots, v_{o_{i+1}}, -x_{i+1}, +x_i)$. The last vertex $+x_k$ belongs to what is left of the cycle C or to a trivial cycle, in the case where all target edges of C belong to the removed run. An example of this operation is shown in Fig. 4.

If $\Lambda(C) \leq 2$, then removing a run of C reduces both the number of runs and the indel potential of the graph by one. Otherwise, removing an insertion run leads to the merging of two deletion runs. In this case, the number of runs of C decreases by two and the indel potential of the graph decreases by one. As the insertion adds k elements in A and k cycles in the graph with no runs, we have $\Delta c(A, \iota^n, \phi) = 0$. Therefore, $\Delta c(A, \iota^n, \phi) + \Delta\lambda(c, \iota^n, \phi) = 1$. □

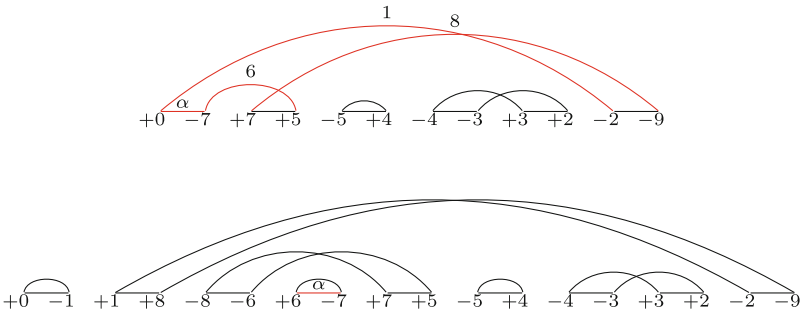

Fig. 4. Example of a insertion that removes a run from a cycle. In this example, we have the insertion run $(+0, -2, -9, +7, +5, -7)$. The insertion of $(+1\ -8\ +6)$ at the start of A removes this run and creates three new cycles.

Now, we show how block interchange operations can be used to increase the number of cycles in the graph without increasing the indel potential.

Lemma 8. *For any strings A and ι^n, such that $|\pi^A| + 1 - c(A, \iota^n) > 0$ and $G(A, \iota^n)$ has no labeled target edges, there exists a block interchange $\mathcal{BI}$ such that $\Delta c(A, \iota^n, \mathcal{BI}) + \Delta\lambda(c, \iota^n, \mathcal{BI}) = 2$.*

Proof. Consider that $G(A, \iota^n)$ has an oriented cycle $C = (c_1, \ldots, c_\ell)$, and let c_i, c_j, c_k be a triple such that $i < j < k$ and $c_i > c_k > c_j$. Such triple always exists in an oriented cycle and it is called an oriented triple [4]. A block interchange applied on these three source edges creates three cycles C', C'', and C''' [4]. Let S_1 and S_2 be the two segments changed by the block interchange. If the source edges are labeled, we can move the elements α in such a way that they end up in the same cycle. To do this, we include the segment to be removed from the first source edge in S_1 and the segment to be removed from the third source edge in S_2. In this way, the segments to be removed are merged in a single source edge, as shown in Fig. 5. An analogous operation is used if only two of these source edges are labeled. Therefore, this block interchange does not affect the indel potential of the graph and increases the number of cycles by two.

Consider that $G(A, \iota^n)$ has only non-oriented cycles and let $C = (c_1, \ldots, c_\ell)$ be a cycle of $G(A, \iota^n)$. Bafna and Pevzner [4] showed that for every e_{c_i} and e_{c_j} from C, with $c_i > c_j$, there exists a cycle $D = (d_1, \ldots, d_{\ell'})$ with source edges e_{d_x} and e_{d_y} such that either $c_i > d_x > c_j > d_y$ or $d_x > c_i > d_y > c_j$. Assume, without loss of generality, that $c_i > d_x > c_j > d_y$. A block interchange that acts on these four source edges creates four new cycles C', C'', D', and D'': C' is formed by the path that goes from e_{c_i} to e_{c_j} with a source edge that joins the first and last vertices of this path; C'' is formed by the path that goes from e_{c_j} to e_{c_i} with a source edge that joins the first and last vertices of this path; D' and D'' are analogous. The first segment of the block interchange starts at the source edge d_y, including the segment to be removed from e_{d_y} if the edge e_{d_y} is labeled, and ends at the source edge c_j. The second segment starts at the source edge d_x and ends at the source edge c_i, including the segment to be removed from e_{c_i} if the edge e_{c_i} is labeled. In this way, the segments to be removed from the same cycle are merged and the number of deletion runs remains the same, as shown in Fig. 6. □

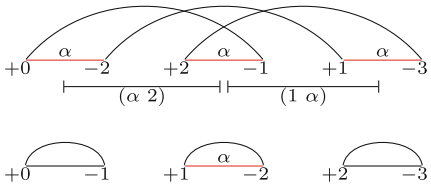

Fig. 5. Example of a block interchange acting on an oriented cycle and creating three new cycles. In this example, we have $A = (0\ \alpha\ 2\ \alpha\ 1\ \alpha\ 3)$ and $n = 2$. The block interchange moves the elements α in such a way that only one source edge remains labeled.

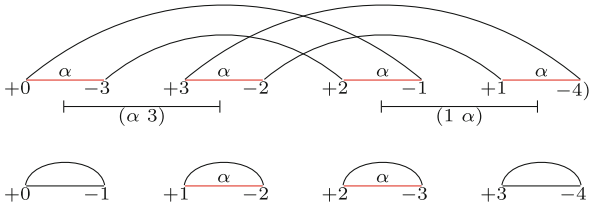

Fig. 6. Example of a block interchange acting on two non-oriented cycles and creating fours new cycles. In this example, we have $A = (0\ \alpha\ 3\ \alpha\ 2\ \alpha\ 1\ \alpha\ 4)$ and $n = 3$. The block interchange moves the elements α in such a way that the segments to be removed from the same cycle are merged.

Lemma 9. *For any strings A and ι^n, such that $|\pi^A| + 1 - c(A, \iota^n) = 0$, there exists a deletion ψ with $\Delta c(A, \iota^n, \psi) + \Delta\lambda(c, \iota^n, \psi) = 1$.*

Proof. Since $|\pi^A| + 1 - c(A, \iota^n) = 0$, each cycle of this graph is trivial. Each cycle has at most one insertion run and one deletion run. A deletion that cleans a source edge of a cycle C decreases the number of runs in C by one. Therefore, $\Delta\lambda(c, \iota^n, \psi) = 1$ and $\Delta c(A, \iota^n, \psi) = 0$. □

Algorithm 1 uses the results of Lemmas 7 to 9.

Theorem 1. *Algorithm 1 is a 2-approximation for the problem of rearrangement distance with block interchanges and indels.*

Proof. By Lemmas 7 to 9, each operation $\beta \in \{\mathcal{BI}, \phi, \psi\}$ applied by the algorithm has $\Delta c(A, \iota^n, \beta) + \Delta\lambda(c, \iota^n, \beta) \geq 1$. In this way, at the end of the algorithm, the resulting string A' satisfies $|\pi^{A'}| + 1 - c(A', \iota^n) + \lambda(A', \iota^n) = 0$ and, consequently, $A' = \iota^n$. Furthermore, the algorithm uses at most $|\pi^A| + 1 - c(A, \iota^n) + \lambda(A, \iota^n)$ operations. By Lemma 5, the algorithm is a 2-approximation. □

Algorithm 1: 2-Approximation algorithm for block interchange and indels distance.

Input: Strings A and ι^n
Output: An upper bound for the rearrangement distance $d_{\mathcal{BI}}(A, \iota^n)$

```
Let d ← 0
while G(A, ι^n) has insertion runs do
    Apply an insertion according to Lemma 7
    d ← d + 1
while |Σ_A ∩ Σ_{ι^n}| + 1 − c(A, ι^n) > 0 do
    Apply a block interchange according to Lemma 8
    d ← d + 1
while G(A, ι^n) has deletion runs do
    Apply a deletion according to Lemma 9
    d ← d + 1
return d
```

For the BI and Reversal Distance, we consider gene orientation. Therefore, it is possible that divergent cycles exist in the labeled cycle graph. For convergent cycles, we can still apply only block-interchanges to create new cycles in the graph. The next lemma shows that it is always possible to find a reversal applied to a divergent cycle and break it into two new cycles, while maintaining the indel potential.

Lemma 10. *For any strings A and ι^n, such that $G(A, \iota^n)$ has no labeled target edges and $G(A, \iota^n)$ has a divergent cycle C, there exists a reversal ρ with $\Delta c(A, \iota^n, \rho) + \Delta\lambda(c, \iota^n, \rho) = 1$.*

Proof. Let $C = (c_1, c_2, \ldots, c_k)$ be a divergent cycle in $G(A, \iota^n)$ and let $(e_{c_x}, e_{c_{x+1}})$ be a pair of divergent edges with minimum x. A reversal applied to these edges breaks C into a trivial cycle C' and another cycle C'' [3]. The reversal can be chosen in a way that any α element is accumulated in the cycle C'', which makes the indel potential of the trivial cycle equals 0 and the indel potential of C'' equals the indel potential of C. In this way, we have $\Delta c(A, \iota^n, \rho) + \Delta\lambda(c, \iota^n, \rho) = 1$. An example of such operation is shown in Fig. 7. □

Theorem 2. *Algorithm 2 is a 2-approximation for the problem of rearrangement distance with block interchanges, reversals, and indels.*

Proof. By Lemmas 7 to 10, each operation $\beta \in \{\mathcal{BI}, \rho, \phi, \psi\}$ applied by Algorithm 2 has $\Delta c(A, \iota^n, \beta) + \Delta\lambda(c, \iota^n, \beta) \geq 1$. After the while loop, the resulting string A' satisfies $|\pi^{A'}| + 1 - c(A', \iota^n) + \lambda(A', \iota^n) = 0$ and $A' = \iota^n$. Therefore, the algorithm uses at most $|\pi^A| + 1 - c(A, \iota^n) + \lambda(A, \iota^n)$ operations. By Lemma 6, the algorithm is a 2-approximation. □

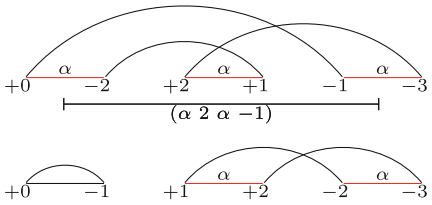

Fig. 7. Example of a reversal acting on a divergent cycle and creating two new cycles. In this example, we have $A = (0\ \alpha\ 2\ \alpha\ -1\ \alpha\ 3)$ and $n = 2$. The reversal moves the α element in such a way that the trivial cycle created has only clean edges.

Algorithm 2: 2-Approximation algorithm for Block Interchange, Reversal and Indel Distance.

Input: Strings A and ι^n
Output: An upper bound for the rearrangement distance $d_{\rho,\mathcal{BI}}(A, \iota^n)$

```
Let d ← 0
while G(A, ι^n) has insertion runs do
    Apply an insertion according to Lemma 7
    d ← d + 1
while |Σ_A ∩ Σ_{ι^n}| + 1 − c(A, ι^n) > 0 do
    if G(A, ι^n) has a divergent cycle then
        Apply a reversal according to Lemma 10
        d ← d + 1
    else
        Apply a block interchange according to Lemma 8
        d ← d + 1
while G(A, ι^n) has deletion runs do
    Apply a deletion according to Lemma 9
    d ← d + 1
return d
```

The time complexity of both algorithms is $O(n^2)$. Creating the Labeled Cycle Graph and classifying its cycles takes $O(n)$ time. Each while loop runs for $O(n)$ iterations, and each operation can be performed in $O(n)$ time.

4 Conclusion

In this work, our main results are related to a structure called labeled cycle graph. This graph can represent a complete instance of the problems, and we were able to present good bounds for the Block Interchange and Indel Distance and the Block Interchange, Reversal, and Indel Distance. With these results, we developed 2-approximation algorithms for both distance problems.

The present study assumed equal costs for all rearrangements and absence of repeated genes in the genomes. To extend the research, future works can explore variations in the costs of rearrangements and the inclusion of genomes containing repeated genes.

References

1. Alexandrino, A.O., Oliveira, A.R., Dias, U., Dias, Z.: Genome rearrangement distance with reversals, transpositions, and indels. J. Comput. Biol. **28**(3), 235–247 (2021)
2. Alexandrino, A.O., Oliveira, A.R., Dias, U., Dias, Z.: Labeled cycle graph for transposition and indel distance. J. Comput. Biol. **29**(03), 243–256 (2022)
3. Bafna, V., Pevzner, P.A.: Genome rearrangements and sorting by reversals. SIAM J. Comput. **25**(2), 272–289 (1996)
4. Bafna, V., Pevzner, P.A.: Sorting by transpositions. SIAM J. Discret. Math. **11**(2), 224–240 (1998)
5. Braga, M.D., Willing, E., Stoye, J.: Double cut and join with insertions and deletions. J. Comput. Biol. **18**(9), 1167–1184 (2011)
6. Bulteau, L., Fertin, G., Rusu, I.: Sorting by transpositions is difficult. SIAM J. Discret. Math. **26**(3), 1148–1180 (2012)
7. Caprara, A.: Sorting permutations by reversals and eulerian cycle decompositions. SIAM J. Discret. Math. **12**(1), 91–110 (1999)
8. Chen, X.: On sorting permutations by double-cut-and-joins. In: Thai, M.T., Sahni, S. (eds.) COCOON 2010. LNCS, vol. 6196, pp. 439–448. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-14031-0_47
9. Christie, D.A.: Sorting permutations by block-interchanges. Inf. Process. Lett. **60**(4), 165–169 (1996)
10. El-Mabrouk, N.: Genome rearrangement by reversals and insertions/deletions of contiguous segments. In: Giancarlo, R., Sankoff, D. (eds.) CPM 2000. LNCS, vol. 1848, pp. 222–234. Springer, Heidelberg (2000). https://doi.org/10.1007/3-540-45123-4_20
11. Fertin, G., Labarre, A., Rusu, I., Tannier, É., Vialette, S.: Combinatorics of Genome Rearrangements. Computational Molecular Biology. The MIT Press, London (2009)
12. Hannenhalli, S., Pevzner, P.A.: Transforming cabbage into turnip: polynomial algorithm for sorting signed permutations by reversals. J. ACM **46**(1), 1–27 (1999)

13. Kececioglu, J.D., Sankoff, D.: Exact and approximation algorithms for sorting by reversals, with application to genome rearrangement. Algorithmica **13**, 180–210 (1995)
14. Willing, E., Stoye, J., Braga, M.: Computing the inversion-indel distance. IEEE/ACM Trans. Comput. Biol. Bioinf. **18**(6), 2314–2326 (2021)
15. Willing, E., Stoye, J., Braga, M.D.: Computing the inversion-indel distance. IEEE/ACM Trans. Comput. Biol. Bioinf. **18**(6), 2314–2326 (2020)
16. Yancopoulos, S., Attie, O., Friedberg, R.: Efficient sorting of genomic permutations by translocation, inversion and block interchange. Bioinformatics **21**(16), 3340–3346 (2005)