A Brief Introduction to the Sorting by Transposition Problem

Ulisses Martins Dias
Zanoni Dias

1 Introduction

A genome can be abstracted as an ordered set of oriented genes (or group of genes). Genome rearrangements are a class of mutation affecting the order and, sometimes, the orientation, of group of genes. Genome rearrangements occur when chromosomes break in specific locations, and the pieces are joined in a different way. The result is a sequence with the same set of genes (and so the same features), but in a different order.

We assume that the order of block of genes in a genome is represented by a permutation \( \pi = (\pi_1 \pi_2 \ldots \pi_n) \), for \( \pi_i \in \mathbb{I}, 0 < |\pi_i| \leq n \) and \( i \neq j \leftrightarrow |\pi_i| \neq |\pi_j| \). Each element of the permutation has a plus or minus sign. This sign represents the orientation of the genes in a given genome.

There are many different rearrangement operations, such as transpositions, reversals and translocations, that transform a given genome into another. These events generate interesting optimization problems for computer science. We describe each of the above events as follow:

- **Reversals** involve only one chromosome. These events break the chromosome in two points. The pieces are reassembled, but the segment between these two points is reversed.

- **Transpositions** also involve only one chromosome. These events break the chromosome in three points. The segment between the first two breakpoints is removed and it is inserted again in the third point.

- **Translocations** involve two different chromosomes. These chromosomes exchange contiguous groups of genes in their ends.

Figure 1 shows the three types of genome rearrangements described above using our model of genome. We called \( P1, P2 \) and \( P3 \) the break points in
the chromosomes. We also colored in blue and red the regions affected by the events. Finally, we called CR1 and CR2 the chromosomes affected by the translocation events.

For each genome rearrangement, we can define the distance between two genomes $\pi$ and $\sigma$ as the minimum number of events to transform one genome into the other. Note that different events lead to different distance problems. For example, reversals lead to the polynomial solvable distance by reversals problem, and transpositions lead to the distance by transpositions problem, which is still open.

We define the identity permutation as $\iota = (1 \ 2 \ldots \ n)$. We define the “sorting distance” as the distance between $\pi$ and $\iota$. The sorting distance problem is equivalent to the distance between two genomes problem, because the distance between $\sigma$ and $\pi$ is equivalent to the distance between $\sigma \pi^{-1}$ and $\iota$.

The best studied genome rearrangement is the sorting by reversals problem. When the orientation of genes is known, it is possible to sort any permutation in polynomial time. However, when the orientation of the genes is unknown, the problem is NP-Hard.
The sorting by transposition problem is still open. The best known approximation algorithm is 1.375. Some exact approaches using Integer Linear Programming and Constraint Logic Programming are also known. However, they sort only permutations up to 30 elements.

2 Sorting by Transposition Problem

A transposition \( \rho(i, j, k) \), for \( 1 \leq i < j < k \leq n + 1 \), “cuts” an interval \([i, j - 1]\) of \( \pi \) and “pastes” it between \( \pi_{k - 1} \) and \( \pi_k \). The result is the permutation \( \pi \rho = (\pi_1 \ldots \pi_{i - 1} \pi_j \ldots \pi_{k - 1} \pi_i \ldots \pi_{j - 1} \pi_k \ldots \pi_n) \). It is important to realize that transpositions do not change the orientation of the genes. So, from now on the signs of the elements in the permutations is omitted.

The transposition distance \( d(\pi, \sigma) \), between two permutations \( \pi \) and \( \sigma \), is the minimum number \( t \) of transpositions \( \rho_1, \rho_2, \ldots, \rho_t \) such that \( \pi \rho_1 \rho_2 \ldots \rho_t = \sigma \).

There are some 1.5-approximation algorithms published so far. Bafna and Pevzner [1] proposed the first one using a graph structure called the cycle graph. Christie [2] introduced some improvements and Hartman and Shamir [4] proposed a simpler sub-quadratic algorithm. Recently, Mira et al. [6] proposed an approximation algorithm based on Bafna and Pevzner [1], but using the algebraic formalism from Meidanis and Dias [5]. The best known approximation algorithm has the 1.375-approximation ratio and was presented by Elias and Hartman [3] based on a new upper bound on the diameter of 3-permutations. A computer program assisted the proof this algorithm.

In the next section, we present some background in this area.

2.1 Background

We can extend a permutation by adding the elements \( \pi_0 = 0 \) and \( \pi_n = n \). A breakpoint for an extended permutation \( \pi \) is a pair \( (\pi_i, \pi_{i+1}) \) such that \( \pi_{i+1} \neq \pi_i + 1 \). We denote by \( b(\pi) \) the number of breakpoints in \( \pi \), and \( \Delta b(\rho, \pi) = b(\pi \rho) - b(\pi) \) the variation of the number of breakpoints when the transposition \( \rho \) is applied on \( \pi \). Figure 2 shows the transposition \( \rho(1, 6, 9) \) being applied on the permutation \( \pi = (6 \ 3 \ 2 \ 5 \ 8 \ 4 \ 7 \ 1) \). We can note that \( b(\pi) = 9 \) and that \( \Delta b(\rho, \pi) = 3 \).

Breakpoints divide a permutation into strips, that are maximal intervals with no breakpoints. Christie [2] shows that every permutation \( \pi \) can be uniquely transformed into a reduced permutation \( \pi_{\text{red}} \) with \( d(\pi) = d(\pi_{\text{red}}) \). The transformation consists of removing the first strip if it begins with 1 and
the last strip if it ends with $n$. After that, we replace every other strips with its minimal element. The final step is to renumber the resulting sequence to obtain a valid permutation. Figure 2 shows a reduction being applied on the permutation $\sigma = (6 \ 3 \ 4 \ 7 \ 1 \ 2 \ 5 \ 8 \ 9)$. After the reduction we have the equivalent permutation $(4 \ 2 \ 5 \ 1 \ 3)$.

A transposition acts on three points of a permutation. For this reason, it can decrease the number of breakpoints by at least one and at most three. So, we can derive the bounds on Lemma 1 and Lemma 2.

**Lemma 1** For any permutation $\pi$, $d(\pi) \geq \frac{b(\pi)}{3}$.

**Lemma 2** For any permutation $\pi$, $d(\pi) \leq b(\pi)$.

Another useful tool was proposed by Bafna and Pevzner [1]. They proposed a graph structure named cycle graph associated with every permutation $\pi$. The cycle graph $G(\pi)$ is constructed from $\pi$ by first extending $\pi$, $|\pi| = n$, with the elements $\pi_0 = 0$ and $\pi_{n+1} = n + 1$. Then, we define $G(\pi)$ as the graph with vertex set $\{0, 1, \ldots, n+1\}$ and edge set as follows: for all $1 \leq i \leq n+1$, gray edges are directed from $i - 1$ to $i$ and black edges from $\pi_i$ to $\pi_{i-1}$.

There is another equivalent way to represent the cycle graph $G(\pi)$. The vertex set is defined by first representing each element of $\pi$, $|\pi| = n$, as a tuple $(-\pi_i, +\pi_i)$, and after we append the elements $0$ and $-(n+1)$. For example, the permutation $\pi = (5 \ 2 \ 1 \ 4 \ 3)$ generates the vertex set $\{0, -5, 5, -2, 2, -1, 1, -4, 4, -3, 3, -6\}$.

The gray edges set is $\{(i-1), -i\}$, for $1 \leq i \leq n+1$ and the black edges set is $\{(-\pi_i, +\pi_{i-1})\}$, for $1 \leq i \leq n+1$ (Figure 3).

As we can see in Figure 3, a gray edge is paired with a black edge for each vertex in $G(\pi)$. So, we can find a unique decomposition of the edges of
Figure 3: Cycle graph $G(\pi)$ for $\pi = (5 \ 2 \ 1 \ 4 \ 3)$. The graph contains two cycles: $C_1 = (5 \ 3 \ 1)$ and $C_2 = (6 \ 4 \ 3)$. $C_1$ is an oriented cycle and $C_2$ is nonoriented.

$G(\pi)$ into cycles of alternating colors. We represent the number of cycles in $G(\pi)$ by $c(\pi)$. For example, we can find two cycles in figure 3, $c(\pi) = 2$.

We represent a cycle $C$ by listing (in order) the labels of the black edges on it, $C = (i_1, \ldots, i_k)$. We assume that the initial black edge $i_1$ of cycle $C$ is the black edge labeled with the highest number in $C$. In figure 3, we represent the two cycles as $(5, 1, 3)$ and $(6, 4, 2)$ according to the numbering of black edges.

We denote a cycle with $k$ black edges as a $k$-cycle and classify it as long if $k > 2$, and short, otherwise. A $k$-cycle is odd if $k$ is odd, and even, otherwise. The number of odd cycles on $G(\pi)$ is denoted as $c_{odd}(\pi)$. The identity permutation is the only permutation with $n + 1$ cycles. Thus, the sequence of transpositions that sorts $\pi$ must increase the number of cycles from $c(\pi)$ to $n + 1$.

The reduction operation defined previously has the effect of removing all the cycles of size 1. Figure 4 shows what happens in the cycle graph of the permutation $\pi = (6 \ 3 \ 4 \ 7 \ 1 \ 2 \ 5 \ 8)$ after applying a reduction. This operation decreases the time and space consumption of the algorithms.

Let $\Delta c(\rho) = c(\pi \rho) - c(\pi)$ be the variation in the number of cycles and $\Delta c_{odd}(\rho) = c_{odd}(\pi \rho) - c_{odd}(\pi)$ be the variation in the number of odd cycles after a transposition $\rho$ be applied. Bafna and Pevzner [1] showed $\Delta c, \Delta c_{odd} \in \{2, 0, -2\}$.

Figure 5 shows the possible change in the number of cycles that might occur. Figure 5.a shows a valid transpositions that must never be used if we are interested in the minimum number of transpositions, because it decreases the number of cycles. The other transpositions could be used depending on each case.

For $x \in \{2, 0, -2\}$, we define an $x$-move as a transposition $\rho$ such that $\Delta c(\rho) = x$. We denote a transposition $\rho$ as valid if $\Delta c(\rho) = \Delta c_{odd}(\rho)$. Figure 7.a is an example of a valid 2-move and Fig. 7.b is an example of a valid 0-move. It is interesting to use as many valid 2-moves as possible in order to sort using the minimum amount of transposition.
Bafna and Pevzner [1] studied the structure of cycles which allow 2-moves and valid 2-moves. They used these cycles to devise an algorithm with 1.5 performance guarantee. In fact, they show that only one type of oriented cycle do not allow valid 2-moves. We show this cycle on Fig. 6.

Let’s define the distance $\text{dist}(r, t)$ between two black edges $r$ and $t$ in the same cycle $C$ as the number of gray edges in $C$ that must be traversed to go from $r$ to $t$. The reason why the cycle in Fig. 6 does not allow a valid 2-move is that we can not find an oriented triple $(x, y, z)$ with two odd distances in the set \{$\text{dist}(x, y), \text{dist}(y, z), \text{dist}(z, x)$\}. So, the transposition will generate three cycles $c_1$, $c_2$ and $c_3$ with sizes $|c_1| = \text{dist}(x, y)$, $c_2 = \text{dist}(y, z)$ and $|c_3| = \text{dist}(z, x)$, and at most one of these cycles will be odd.

Bafna and Pevzner also defined a 0-move as good if it augments the odd cycles set by two. Figure 7.c is an example of a good 0-move. The good 0-move is a better choice than a valid 0-move, because the identity has only odd cycles. So, it is very important to use transpositions that increase the number of odd cycles even if it does not change the aggregate number of cycles.

We denote an $(x, y)$-sequence of transpositions as a sequence of $x$ transpositions, such that at least $y$ of them are valid 2-moves. For example, Fig. 8.a shows a sequence of 4 transpositions, such that 3 of them are valid 2-moves. Another example is the so-called $(0, 2, 2)$-sequence in previous papers. This sequence is a $(3, 2)$-sequence (Figure 8.b).

We classify cycles in two different groups, nonoriented for which no 2-moves are possible, and oriented, otherwise. For all $k > 1$, a cycle $C = (i_1, \ldots, i_k)$ is nonoriented if $i_1, \ldots, i_k$ is a decreasing sequence, otherwise $C$ is an oriented cycle. For example, Figure 3 presents two cycles $C_1 = (5, 1, 3)$
and $C_2 = (6, 4, 2)$, $C_1$ is an oriented cycle and $C_2$ is a nonoriented one.

For example, in Fig. 7.b, the transposition $\rho(2, 4, 6)$ is a shuffling transposition with respect to the black edges of the cycle $(1, 3, 5)$. In this example, the shuffling transposition created an oriented cycle $(5, 1, 3)$. In fact, Let $(x, y, z)$ be three black edges on a nonoriented cycle. A shuffling transposition with respect to $(x, y, z)$ leads to an oriented cycle. So, After the shuffling transposition we can apply at least one valid 2-move $\rho'(x', y', z')$, where $x'$, $y'$ and $z'$ are the new positions of $x$, $y$ and $z$.

A class of cycles that always allow valid 2-moves is called strongly oriented cycles. This class is defined as follows: let $C = (i_1, i_2, \ldots, i_k)$ be a cycle in $G(\pi)$ and let $C^*$ be the sequence of black edges of $C$ in decreasing order. A strongly oriented cycle is an oriented cycle for which $C^*$ could be transformed into $C$ by a single transposition. A strongly oriented cycle $C$ and a nonoriented cycle $C' = (j_1, \ldots, j_k)$ are strongly crossed if the valid 2-move on $C$ is a shuffling transposition on $C'$. 
Figure 7: Examples of moves: a) valid 2-move, b) valid 0-moves and c) good 0-moves.

Figure 8: Examples of sequences: a) (4, 3)-sequence, b) (3, 2)-sequence, also known as (0, 2, 2)-sequence in previous papers.

3 Bafna-Pevzner Algorithm

Bafna and Pevzner [1] presented the algorithm below using some of the concepts presented in the previous section. As in the worst case the algorithm will make a (3, 2)-sequence, the approximation ratio is 1.5.
Algorithm 1. Bafna-Pevzner Algorithm

Require: \( \pi \)
Ensure: A sorting for \( \pi \) with 1.5 approximation ratio

while \( \pi \) is not the identity permutation do

if there is a long cycle \( C \) on \( G(\pi) \) then

if \( C \) is a strongly crossed cycle then

Apply two consecutive valid 2-moves
else

if \( C \) is a strongly oriented cycle then

Apply a valid 2-move
else

if \( C \) is an oriented cycle then

Apply a valid 2-move or a \((3,2)\)-sequence
else

Apply a \((3,2)\)-sequence
end if
end if
end if
else

Apply a good 0-move followed by a valid 2-move
end if
end while

References


