International Journal of Basic and Applied Biology
Print ISSN: 2349 – 5820; Online ISSN: 2349 – 5839 ; Volume. 2 No. 2, November 2014. pp. 7 – 12
© Krishi Sanskriti Publications
http://www.krishisanskriti.org/ijbab.html

Enumerating All Parsimonious Sequences of Reversals and Translocations

Kanhaiya Singh[1]**, Amritanjali**[2]

[1, 2]Department of Computer Science and Engineering
Birla Institute of Technology
Mesra, Ranchi, Jharkhand, India 835215.

Abstract : **Reversals and translocations are most common rearrangement operations in the evolution of multichromosomal genomes. Using parsimony hypothesis it is possible to deduce the sequence of rearrangements that accompanied the evolution of genomes. The problem of finding the optimal sequence of reversals and translocations transforming the gene order of one genome into other is called the problem of sorting by reversals and translocations (SBRT). The SBRT problem was initially solved by a reduction to the problem of sorting by reversals, where each reversal simulates either a reciprocal translocation or an internal reversal. Later algorithms were proposed that treat reversals and translocations as distinct operations. All the existing algorithms for the SBRT problem provide only a single optimal sequence of reversals and translocations. Recently an attempt was made to list all optimal sorting sequences for the SBRT problem that works with limited permutations. In this paper we extend the approach and present a method to completely enumerate the solution space of the SBRT problem. Each solution presents a probable evolutionary scenario of the given genomes. The true evolutionary scenario will be represented by one of the given set of solutions.**

1. INTRODUCTION

The appearance of gene content and gene order data has greatly facilitated in carrying out phylogenetic analysis. The evolutionary distance between two species is estimated by the amount of differences between the order and orientation of the shared genes on their chromosomes. During evolution, genomes are subject to rearrangements, which are large scale mutations that can change not only the ordering and orientation (strandedness) of the genes on the chromosomes but also the chromosomes on which genes are present. Compared to point mutations, these events are very rare so they are more useful in deducing the evolutionary relationships.

One is interested in finding the most "plausible" sequence of rearrangements that resulted in the divergence of two genomes from their common ancestor. It describes their evolutionary scenario. From parsimony hypothesis the most plausible scenario is the one that involves minimum number of rearrangements as they are very rare events. Therefore, for given two genomes, one wants to find an optimal (minimal) sequence of rearrangements that transforms the gene order of the shared genes in one genome into that of the other. In the classical approach, each genome has exactly one copy of each gene, and only operations that do not change the genome content are considered. The minimum number of rearrangements required is known as the rearrangement distance between the two genomes.

The most common rearrangements events in multichromosomal genomes are reversals and translocations. They are balanced genome rearrangement operations, so the number of chromosomes and number of genes remains unchanged after these events. A reversal operation reverses the order and the orientation of the genes in a segment inside a chromosome. In a translocation operation the ends of two chromosomes gets exchanged. The problem of finding the optimal sequence of reversals and translocations transforming the gene order of one genome into other is called the problem of sorting by reversals and translocations (SBRT).

The area of sorting multichromosomal genomes has been subject of lot of research. The problem is solved by representing the relative order and orientation of the shared genes in the two genomes by signed permutations and constructing a graph that give information about the adjacent genes in each of the permutation. The SBRT problem was initially solved by a reduction to the problem of sorting by reversals, where each reversal simulates either a translocation or a reversal [1-4]. Recently, the algorithm proposed in [5] treats reversals and translocations as distinct operations. All these algorithms provide only a single optimal sequence of reversals and translocations. As there are multiple solutions to the sorting by reversals problem [6-8] and to the sorting by translocations problem [9], so there can be multiple solutions to the SBRT problem too. An attempt was made to list all optimal sorting sequences for the SBRT problem by [10] that works with limited permutations. In this paper we extend the approach to remove its limitations while completely enumerating all possible parsimonious sequences for given two genomes.

2. PROBLEM FORMULATION

The evolutionary scenario of given pair of genomes is described by the sequence of rearrangements that can transform the shared gene order of one genome into that of the other in minimal number of steps. To determine this one of the genome is taken as source and the other one as target. The shared gene order in the target genome is marked in ascending order and is represented by an identity permutation. The relative order and orientation of these genes in the source genome is denoted by a signed permutation, where '-' sign indicates opposite orientation. As the genomes are multichromosomal so the shared genes are distributed over multiple chromosomes. The rearrangement operations are performed on the source genome permutation such that it is transformed into the target genome permutation in minimum number of steps. Each rearrangement operation cuts the source genome at two different positions, referred as cut points. For reversals, the two cut points will be on the same chromosome. And, for translocation the cut points are taken on two different chromosomes. Here we have considered internal reversals and reciprocal translocations. A reversal is *internal* if it does not involve ends of the chromosomes. A translocation is *reciprocal* if none of the exchanged ends is empty. If the head of a chromosome (chromosomal segment before the cut point) is exchanged with the tail (chromosomal segment after the cut point) of another chromosome then it is called prefix-suffix translocation otherwise if tails of both the chromosomes are exchanged then it is called prefix-prefix translocation.

The rearrangement distance and the possible cut points for sorting rearrangements are determined with the help of the graph proposed by [1]. Each gene is represented by two vertices such that the signed permutation for the source genome is transformed into unsigned permutation, while preserving the orientation information for each gene. For positive sign gene $+x_i$, the ordered pair is (x_i^h, x_i^t) and for negative signed gene $–x_i$, the ordered pair is (x_i^t, x_i^h). Vertices in the graph are the set of ordered pairs corresponding to each gene in the genome. They are displayed linearly according to the order of genes in the chromosomes. The adjacencies in the source and target permutations are shown with the help of bi-colored edges. Black edges connect adjacent genes in the chromosomes of the source genome. Gray edges connect those genes of the source genome that are adjacent in the target chromosomes. If both the ends of the edge are on the same chromosome then it is called internal else if they are on different chromosomes then it is called external. A gray edge can be internal or external but all the black edges are internal. An example graph (also known as cycle graph) is shown in Figure 1.

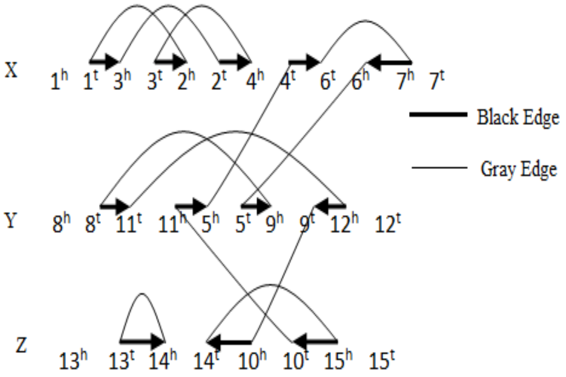
X 1^h 1^t 3^h 3^t 2^h 2^t 4^h 4^t 6^t 6^h 7^h 7^t
Y 8^h 8^t 11^t 11^h 5^h 5^t 9^h 9^t 12^h 12^t
Z 13^h 13^t 14^h 14^t 10^h 10^t 15^h 15^t
Black Edge
Gray Edge

Figure 1. The graph G (A, B), where source genome A = {(1, 3, 2, 4, -6, 7), (8, -11, 5, 9, 12), (13, 14, 10, 15)} and target genome B = {(1, 2, 3, 4, 5, 6, 7), (8, 9, 10, 11, 12), (13, 14, 15)} distributed on 3 chromosomes X, Y and Z, respectively. (Arrow shows the direction of black edges when traversing the cycles).

The graph consists of cycles of alternate gray and black edges. A cycle is trivial if it consists of only a pair of gray and black edges. Intersecting cycles form one component. A trivial component consists of a trivial cycle. A gray edge is said to be oriented if it connects genes with opposite orientation otherwise unoriented. A cycle is oriented if it contains one or more oriented gray edge(s) otherwise it is unoriented. Similarly, a component is oriented if it has one or more oriented cycles otherwise it is unoriented. A benign component refers to a trivial or an oriented component. An unoriented internal component is termed as knot if it does not separate two other unoriented components. A knot can be simple knot or a superknot. A superknot protects other unoriented internal components from becoming knots i.e. when a superknot is eliminated a non-knot becomes a knot.

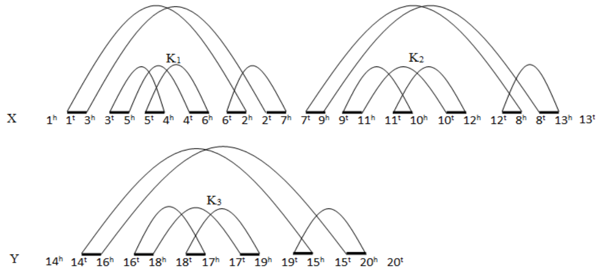
X
1^h 1^t 3^h 3^t 5^h 5^t 4^h 4^t 6^h 6^t 2^h 2^t 7^h 7^t 9^h 9^t 11^h 11^t 10^h 10^t 12^h 12^t 8^h 8^t 13^h 13^t
K_1
K_2
Y
14^h 14^t 16^h 16^t 18^h 18^t 17^h 17^t 19^h 19^t 15^h 15^t 20^h 20^t
K_3

Figure 2. Fortress with three superknots K_1, K_2 and K_3.

A fortress exist iff there are odd number of knots and all are superknots. If the graph has knot(s) but fortress is not present then either there exist one or more simple knot(s) or number of superknots is even. If fortress is present then there are at least three superknots and no simple knots. Figure 2 shows an example of fortress with 3 superknots.

For two genomes with 'n' shared genes distributed over N chromosomes, the rearrangement distance 'd' is given in [1] as: d = n – N – c + k + f where 'c' is number of cycles in the graph, 'k' is number of knots and 'f' is 1 if fortress is present otherwise 0. This gives the minimal number of rearrangements required to transform the source gene order into the target gene order which is a sorted order. Therefore, we have to list all possible sequences of reversals and translocations that sort the source gene order in 'd' steps. Let Δc denote the change in the number of cycles after performing a rearrangement operation, then $\Delta c = \{-1, 0, +1\}$. Based on the value of Δc, a reversal operation can be classified as split ($\Delta c = 1$), neutral ($\Delta c = 0$) or joint ($\Delta c = -1$) [11]. Similarly, translocation operations are classified as proper ($\Delta c = 1$), improper ($\Delta c = 0$) or bad ($\Delta c = -1$) [12]. Any rearrangement operation applied on the source genome is valid if the rearrangement distance between the rearranged genome and the target reduces by 1, i.e. $\Delta d = \Delta(k + f - c) = -1$. An optimal (minimal) sequence consists of only valid rearrangements.

3. FINDING VALID REARRANGEMENTS

It has been shown that several valid rearrangements can exist. When these rearrangements are applied over the given source genome sequence, a new set of resulting genome sequences are obtained. Again for each of these genome sequences next set of valid rearrangements are obtained and the same process is repeated. After applying *d*-sequence of valid rearrangements, the source genome is transformed into the target genome, as each rearrangement is reducing the distance by one.

The graph of the source genome may contain internal and external components whereas the graph for the target genome contains only (internal) trivial components/cycles. To transform the source genome into the target genome the rearrangements are applied such that all the non-trivial components are eliminated. [1, 2, 6,11-13] describe the effect of applying a reversal or a translocation on the components in the graph. External components are eliminated by applying translocation on external cycles. The internal components may be oriented or unoriented. Oriented internal components can be transformed into trivial components by applying reversals on oriented cycles. Unoriented internal components are eliminated by applying reversals on single chromosome and/or translocations on two chromosomes. The rearrangement applied at each step must be valid so that the transformation takes place in least number of steps.

3.1. Rearrangements Eliminating Unoriented Internal Components

The given source genome cannot be sorted until all the non-trivial unoriented internal components get eliminated if present. Both reversals and translocations can be used for removing these components. Siepel [6] has described in detail the process for finding all valid reversals that can eliminate unoriented components from the graph. Neutral reversals can be applied by taking cut points on a pair of black edges in same unoriented cycle. This does not change number of cycles but reduces number of knots by 1.If fortress is not present then such a reversal is valid iff the unoriented cycle belongs to a simple knot and either the number of superknots is even or it is not the only simple knot. This ensures that number of knots is reduced without creating fortress. Thus, $\Delta k = -1$, $\Delta c = \Delta f = 0$, and $\Delta d = -1$. If fortress is already present then the unoriented cycle either belongs to a superknot that protects a single non-knot or to the non-knot protected by it. In the first case a superknot is eliminated and the nonknot becomes a simple knot and in the second case the nonknot is changed into oriented component transforming the superknot into a simple knot. In both the cases, the fortress is destroyed and number of knots and cycles remains the same, i.e. $\Delta f = -1$, $\Delta c = \Delta k = 0$, thereby reducing the distance.

Also, joint reversals can be applied by taking cut points on black edges of two cycles in different components. The two components can be a pair of knots, a pair of a knot and a benign component separated by another knot or a pair of benign components separated by two distinct knots. In case where a pair of simple knots is merged it must be ensured that either the number of superknots is even or they are not the only simple knots in the graph. For all such reversals $\Delta k = -2$, Δc = -1 and Δf = 0, thus Δd = -1. If fortress is already present then the reversal merging the two knots is valid if it destroys fortress by creating another knot.

For translocations we cannot take cut points on black edges in same internal cycles. So to eliminate unoriented internal components by translocation, we have to take cut points on black edges of two cycles of different components on separate chromosomes. Such a translocation is bad as it merges two cycles. Both prefix-prefix and prefix-suffix translocations can be done on the pair of cut points. Let 'v' be the number of chromosomes having knots. Following cases describe the possible cut points for valid bad translocations:

Case 1: *f* = 1, k ≥ 3 and k is odd.
If v=1 then the translocation is performed between a chromosome having knot and any other chromosome such that one knot is eliminated and Δ (*c* – *k* – *f*) =Δ (-1 - (-1) - (-1)). If v≥ 2 then choose any two chromosomes for performing bad translocation such that

i) At least one chromosome has knot, Δ (*c* – *k* – *f*) =Δ (-1 - (-1) - (-1)) = 1. The translocation eliminates the knot on the chromosome as well as fortress is destroyed as number of knots becomes even.

ii) Or, both the chromosomes has knot, if *k* = 3**,** then Δ (*c* – *k* – *f*) = Δ (-1 - (-1) - (-1)) = 1. The translocation eliminates the knots on both the chromosomes but turns a non-knot into a knot. So, the fortress is destroyed. Otherwise, if *k* > 3**,** then Δ (*c* – *k* – *f*) = Δ (-1 - (-2) - 0) = 1. In this case the translocation eliminates two knots but the number of knots is still odd so fortress is not destroyed.

Case 2: *f* = 0 and *k* ≥2.
The possible cut points for valid bad translocation are those that reduce number of knots by two. Therefore, a translocation is possible when *v* ≥ 2 as two chromosomes are required. The possible cut points for translocation are between any pair of chromosomes having knots. In all these cases Δ (*c* – *k* – *f*) = Δ (-1 - (-2) - 0) = 1. However, a pair of simple knots can be merged iff the number of superknots is even or they are not the only simple knots of the graph. Otherwise, it will lead to formation of fortress.

3.2. Translocations on External Cycles

Translocation is performed on pair of black edges on different chromosomes in same (external) cycle. The translocations on external cycles are always proper as it increases number of cycles by one. The cycles are traversed starting from any black or gray edge, marking the direction in which each black edge is traversed. A pair of black edges in same cycle are said to be converging if their direction of traversal is same otherwise they are diverging [14]. If the cycle is unoriented then all the pairs of black edges in it are converging. Otherwise, the cycle has at least one diverging pair of black edges. We take cut points for prefix-suffix translocation on every diverging pairs and prefix-prefix translocation on every converging pairs. Figure 3 shows the graph for the rearranged genome on applying each type of translocation. Such a translocation is called as proper translocation as it increases number of cycles by one. This is added to the solution set only if it reduces the distance by 1 i.e. Δd = -1. Note that not all the proper translocations are valid as it may create new unoriented internal components due to which distance may not reduce. From the distance formula a proper translocation is valid iff Δ(k+f) = 0 as Δc = 1. This means either it is not creating any new unoriented internal components or it satisfies following conditions: (a) If fortress is not present in the graph, then after applying the translocation the number of knots must not change. (b) If fortress is present in the graph, then after applying the translocation either the number of

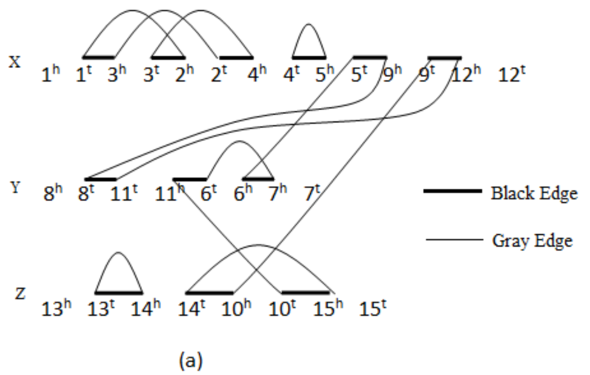
X 1^h 1^t 3^h 3^t 2^h 2^t 4^h 4^t 5^h 5^t 9^h 9^t 12^h 12^t
Y 8^h 8^t 11^t 11^h 6^t 6^h 7^h 7^t
Z 13^h 13^t 14^h 14^t 10^h 10^t 15^h 15^t
Black Edge
Gray Edge
(a)

Fig. 3. Graph after applying a prefix-prefix translocation on the source genome A.

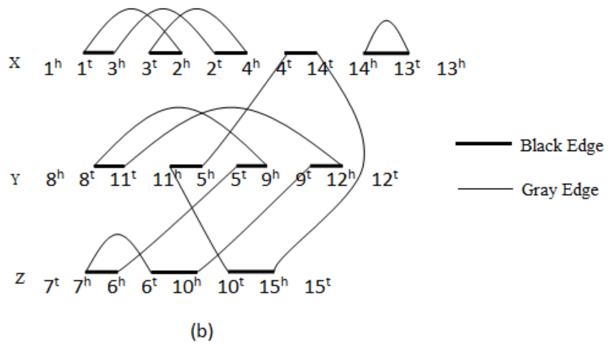
X 1^h 1^t 3^h 3^t 2^h 2^t 4^h 4^t 14^t 14^h 13^t 13^h
Y 8^h 8^t 11^t 11^h 5^h 5^t 9^h 9^t 12^h 12^t
Z 7^t 7^h 6^h 6^t 10^h 10^t 15^h 15^t
Black Edge
Gray Edge
(b)

Fig. 3. Graph after applying a prefix-suffix translocation on the source genome A.

knots must not change or the number knots must increase by 1. In all such cases Δ(k+f) = 0.

3.3. Reversals on Oriented Internal Cycles

Reversal is performed by taking cut points on every diverging pair of black edges in same cycle. It divides the cycle into two, increasing the number of cycles by 1. Such reversals are valid if it reduces the distance. Like translocations, new unoriented internal components may form after applying a reversal. Due to which Δ(k+f) may increase and distance will not reduce. The conditions described for translocations in the preceding section are applicable to reversals also. Only valid reversals are added to the solution set.

4. ENUMERATING ALL PARSIMONIOUS REARRANGEMENT SEQUENCES

The algorithm for enumerating sorting sequences of reversals and translocations is extension of the algorithm proposed by [8] for listing sorting sequences of reversals. The set of valid rearrangements (reversals and translocations) for a given pair of source and target genomes are obtained as described above. They are added to the initial solution set and termed as 1-sequences of rearrangement. The final solution set consists of *d*-sequences of rearrangements.

Sl. No.	Permutation	*d*	No. of Solutions
1.	{(1, 3, 2, 4, 5), (6, 7)}	3	6
2.	{(1, -3, 7), (5, -6, -2, 10), (8, 9, 4)}	4	54
3.	{(1, 3, 2, 4, -9, 6, 7), (8, -5, 10)}	5	120
4.	{(1, 3, 2, 4, 5), (6, 8, 7, 9, 10)}	6	1264
5.	{(1, 3, 5, 4, 6, 2, 7), (8, 10, 9, 11, 12)}	8	166488
6.	(1, 3, 2, 4, -6, 7), (8, -11, 5, 9, 12) (13, 14, 10, 15)	10	>4000000

Taking each operation *ρ* (reversal or translocation) from the set of 1-sequences and applying it over the source permutation (A), a new set of permutations is obtained. Each permutation A' of this set is one step closer to the target permutation. By repeating the above procedure over A', a new set of optimal 1-sequences is generated. When these 1-sequences are combined with their predecessor *ρ*, a set of optimal 2-sequences is generated. Therefore, by iterating this algorithm, the set of all optimal sequences of reversals and translocations are obtained that transforms the genome A into B. This procedure is described by Algorithm *listAllSortingSequences*.

Algorithm *listAllSortingSequences* (A, B)

[Enumerating all optimal sorting sequences of reversals and translocations]

Input: Source genome A and Target genome B

Output: The set of all sequences of reversals and translocations sorting the source genome A into the target genome B

```
begin
  $d \leftarrow$ rearrangement distance between $A$ and $B$
  $R \leftarrow \{\rho \mid \rho$ is an optimal 1−sequence of reversal for A and B$\}$
  $R' \leftarrow \{\rho \mid \rho \in R \wedge \rho$ is an internal reversal$\}$
  $T \leftarrow \{\rho \mid \rho$ is an optimal 1−sequence of
      translocation for A and B$\}$
  $S \leftarrow R' \cup T$      [solution set]
  for each integer $i$ from 2 to $d$ do
    $S' \leftarrow 0$   [contains the $i$-sequences]
    for each $s$ [ $(i-1)$-sequence] in $S$ do
      $A' \leftarrow A \circ s$   [apply $(i-1)$- sequence of rearrangements to A]
      $R \leftarrow \{\rho \mid \rho$ is an optimal 1-sequence
      of reversal for A′ and B$\}$
      $R' \leftarrow \{\rho \mid \rho \in R \wedge \rho$ is an internal reversal$\}$
      $T \leftarrow \{\rho \mid \rho$ is an optimal 1-sequence of
      translocation for A′ and B$\}$
      $G \leftarrow R' \cup T$
      for each $\rho$ in $G$ do
        $s' \leftarrow s.\rho$ [concatenate $\rho$ at the end of sequence $s$]
        insert $s'$ in $S'$  [$s'$ is an $i$-sequence]
      end for
    end for
    $S \leftarrow S'$
  end for
  return $S$  [$S$ is the final set of $d$−sequences]
end
```

5. IMPLEMENTATION

The proposed algorithm has been implemented and tested with random permutations.

Table 1 Computation Results

The results obtained on a personal computer with 3 GB RAM are shown in Table 1. For sequence number 6 the complete set of solutions was not obtained due to memory limitations.

6. CONCLUSION

This paper solves the problem of finding optimal sequences of rearrangements for given two genomes that have evolved through reversals and translocations only. A method is presented for enumerating all possible optimal sequences that can transform the order and orientation of one of the given genome into that of the other genome. The solution space is huge due to large number of cut points detected at each step which requires large memory for storing all the results. Also, large number of solutions makes their analysis difficult. As a future work, solutions with same type of rearrangements can be merged into a single class of solutions. Some additional biological constraints may be applied to further reduce the size of solution set. Another future work is the addition of other rearrangement operations like fission and fusion, which remove the limitation of unique genes and same number of chromosomes in the given genomes.

7. REFERENCES

[1] Hannenhalli, S. and Pevzner, P., "Transforming men into mice: Polynomial algorithm for genomic distance problem", *Proc. 36th Ann. Symp. Foundations of Computer Science*, pp. 581-592, 1995.

[2] Kececioglu, J. and Ravi, R., "Of mice and men: Algorithms for evolutionary distances between genomes with translocation", *Proc. 6th Annu. ACM-SIAM Symp. Discrete Algorithms*, pp. 604–613, 1995.

[3] Tesler. G., "Efficient Algorithms for multichromosomal genome rearrangements", *Journal of Computer and System Sciences*, vol. 65, pp. 587-609, 2002.

[4] Ozery-Flato, M. and Shamir, R., "Sorting by reciprocal translocation via reversals theory", Journal of Computational Biology, vol. 14, pp. 408-422, 2007.

[5] Yin, X. and Zhu, D., "Sorting Genomes by Reversals and Translocations", *Asia-Pacific Conference on Information Processing*, pp. 391-394, 2009.

[6] Siepel, A., "An Algorithm to Enumerate Sorting Reversals for Signed Permutations", *Journal of Computational Biology*, vol. 10, pp. 575-597, 2003.

[7] Braga, M. D., Sagot, M. F., Scornavacca, C. and Tannier, E., "Exploring the Solution Space of Sorting by Reversals, with Experiments and an Application to Evolution", *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 5, no. 3, pp. 348-356, 2008.

[8] M.D. Braga, "Exploring the Solution Space of Sorting by Reversals When Analyzing Genome Rearrangements", *Ph.D. Thesis*, University Lyon 1, France 2008.

[9] Amritanjali, Anand, S. and Sahoo, G., "Exploring the Solution Space of Sorting by Translocations", *Procedia Computer Science*, vol. 11, pp. 160-168, 2012.

[10] Amritanjali and Sahoo, G., "Listing All Sorting Sequences of Reversals and Translocations". *Proc. of ACM Conf. on Bioinformatics and Computational Biology (ACM BCB '13)*, pp. 712, 2013.

[11] Hannenhalli, S. and Pevzner, P., "Transforming cabbage into turnip (polynomial algorithm for sorting signed permutations by reversals)", *Journal of the ACM*, vol. 46, no. 1, pp.1–27, 1999.

[12] Hannenhalli, S., "Polynomial algorithm for computing translocation distance between genomes", *Discrete Applied Mathematics*, vol. 71, pp. 137-151, 1996.

[13] Bergeron, A., Mixtacki, J., and Stoye, J., "On Sorting by Translocations", *Journal of Computational Biology*, vol. 13, pp. 567-578, 2006.

[14] Setubal, J. and Meidanis, J., "Introduction to Computational Molecular Biology", 1st ed., PWS Publishing, Boston, MA, 1997.