A Pattern Matching Based Approach towards Phylogenetic Networks with Constrained Recombination

M. A. H. Zahid
Dept. of E&CE, IIT Roorkee,
Roorkee, Uttaranchal, India
zaheddec@iitr.ernet.in

Ankush Mittal
Dept. of E&CE, IIT Roorkee,
Roorkee, Uttaranchal, India
ankumfec@iitr.ernet.in

R. C. Joshi
Dept. of E&CE, IIT Roorkee,
Roorkee, Uttaranchal, India
rcjosfec@iitr.ernet.in

ABSTRACT

The tree representation of evolutionary relationship oversimplify the view of the process of evolution, as they can not take into account the events such as horizontal gene transfer, hybridization, homoplasy and genetic recombination. Several algorithms exist for constructing the phylogenetic networks, which results from the event horizontal gene transfer, hybridization and homoplasy. Very little work has been published on the algorithmic detail of phylogenetic networks due to recombination. The problem of minimizing the number of recombinations in a phylogenetic network, constructed using binary DNA sequences, is NP-hard. A $O(n^4)$ time algorithm for a special case called node disjoint recombination is proposed in [4]. In [5] conflict graph is used as the major tool to compute node disjoint network in $O(nm+n^3)$ time. In this paper we propose a pattern matching based efficient approach, which detects and construct the phylogenetic network for recombination events using binary sequences in $O(n\log n+n^2m)$ time. This method provides a novel pattern matching based approach to the constrained recombination problem for single and multiple crossovers.

Keywords

Evolutionary relationship, phylogenetic network, genetic recombination, SNP, and gall trees.

1. INTRODUCTION

The tree representation of evolutionary relationship oversimplifies the view of the process of evolution, as they can not take into account the events such as horizontal gene transfer, hybridization, homoplasy and genetic recombination. The network representation of the evolutionary relationship provides a better understanding of the evolutionary process and the non-tree like events [1, 2]. Detection of recombination is very important, because it locates the origin of the gene influencing the genetic disease. A case study on HIV, carried at the center for computational and experimental genomics, University of Southern California, shown that the most frequent recombination events make it difficult to design a drug for HIV. Recombination in HIV is recognized as an important mechanism by which the viruses escape from the attack of the drug [3].

Suppose each species is assigned a binary sequence. When recombination occurs, the child gets some parts of genetic sequence from one ancestor and rest of the sequence will be from another ancestor as shown in Figure 1. The recombination event cannot be modeled with a rooted structure. Instead it can be represented as network where some of the nodes may have two parents, are called as the recombination nodes, as shown in Figure 2. A heuristic algorithm is given in [6] to model the history of the sequences using recombination.

Presence of recombination allows different parts of a single sequence to display different evolutionary histories. This violates the traditional assumption of single evolutionary history underlying the sequences. Since long time, the consequences of the recombination are ignored, and phylogenies were constructed by neglecting the recombination events. Schierup and Hein in [2, 7] and Posada [8] shown the effect of negligence of recombination while constructing the phylogeny. The effects shown by Schierup and Hein include the long terminal branches in star like trees and the rate of heterogeneity among the sites is wrongly inferred. Despite of the above fact, very little have been published on the methods robust for recombination.

Wang et al. [4] showed the problem of finding a perfect phylogenetic network, network with minimum number of recombination nodes, is NP-hard and gave an algorithm for a restricted problem, called node disjoint network, with $O(n^4)$ computing time. The restriction was that network has only node disjoint cycles. In other words, no node can be shared by two recombination cycles. Network with disjoint nodes is called a "gall tree".

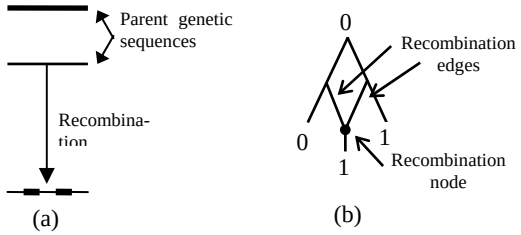

Figure 1. (a) The recombination event. (b) Phylogenetic network for binary sequences.

Gusfield et al. [5] shown that the algorithm in [4] is incomplete and does not constitute a necessary test for the

existence of the gall tree. The method proposed in [5] uses the conflict graph as the main tool for the detection of the conflicting sites. The components of the conflict graph are used to construct galls in the gall tree. Finally all the galls are connected, leading the final gall tree. The algorithm can compute the gall tree, if one exists, in time. In this paper we propose an efficient algorithm for the construction of the gall tree. We used the similarity and dissimilarity between the binary sequences, as a major tool for detecting and constructing the gall tree. The algorithm computes a gall tree, if one exists, in time.

The paper is organized as follows. Section 2 deals with the formal definitions and assumptions related to phylogenetic networks. Section 3 is dedicated to the combinatorial background and conditions for the detection of the recombination. The algorithm for the construction of the network is given in section 4 followed by discussion and conclusion.

2. Preliminaries

This section deal with the basic terminology and assumptions made for the development of algorithm. We followed the terminology from [5] for simplicity.

A phylogenetic network is a directed acyclic graph, but underlying undirected graph can have cycles. Each node in the phylogenetic network *N* has indegree 1 or 2. Nodes with indegree 1 are called tree nodes and the nodes with indegree 2 are called recombination nodes. A tree node is the result of mutation and the recombination node is the due the recombination of genetic material of two parent species of the node. A special node with indgree 0 is called the root of the network. Each node in the network *N* is assigned a binary sequence of length m. The tree nodes have a single site or character change from 0 to 1, when compared with the parent nodes. The sequence of recombination node is the parts of its two ancestor's sequences.

A set of binary sequences represents a phylogenetic network *N*, if and only if each sequence labels exactly one leaf of the network *N*. A phylogenetic network on a set of three binary sequences is shown in figure 1(b). The biological interpretation of a phylogenetic network for *M* sequences is that *N* represents the possible history of the *M* sequences under the following assumptions. (1) There is a single known ancestral sequence. (2) The change in one site, from 0 to 1, is permitted only once (called mutation). (3) Two sequences are permitted to recombine as result of recombination event. (4) Each site in the sequence represents a SNP (single nucleotide polymorphism), a site where two of the four possible nucleotides appear in the population with the frequency above some threshold.

Given a set of species and their binary sequences, a perfect phylogenetic network, always exist with recombination nodes, where n is number of species and m is the length of binary sequences. Recombination is a rare event in the evolutionary process. Therefore a phylogenetic network with minimum number of recombination nodes is informative. A recombination cycle in a phylogenetic network that does not share any node with any other recombination cycles is called a gall. A phylogenetic network is said to be gall tree if every recombination cycle is a gall in the network. But the path from root to any other gall *G* or a node, which is not on the gall *G'*, can pass through the gall *G'*.

3. COMBINATORIAL BACKGROUND FOR RECOMBINATION NODES

In the section we formulate the necessary and sufficient condition for the detection of the recombination nodes. We used the similarity and dissimilarly between the sequences as the major tool for the detection of recombination nodes.

Definition 1. If a node *u* is reachable from a node *v* via a directed path, then *v* is an ancestor of *u*, and *u* is the descendent of *v*.

Lemma 1 is curial for the detection of the gall in the given binary sequences. The similarity and dissimilarity between the sequences, which shares a common parent should be computed after the removing the parent's characteristics from each of the child. This avoids the misleading similarity between the species.

Lemma 1. *Let S and S' be the sequences of the children of node v. if S' is not the result of the mutation or recombination in S then the similarity between S and S' is due to common ancestry*

Proof. Let s' is not a child of the s. By the definition 1, s' is not reachable from S, therefore all the sites or character of S' are different from the characters of the node S or vise versa. Let s and S ' are children of node v, then both s and s' are reachable from node v, and shows the similarity by at least one character (of site) with the parent node v. Both the nodes s and s' show the similarity with their parent node by at least one character. Hence this proves that the distinct nodes will show similarity due to common ancestry.

Given a set of binary sequences these can be assigned to each node in the network based on the following properties. For a non-recombination or the mutation node, the sequence is same as its parent, except at a single site *i*, where the mutation has occurred and the value would be changed from '0' to '1'. This gives the basis for the assumption that if we consider each sequence as a binary number then the parent will always have less value than its children do. Recombination is the process of exchanging the genetic material between the species. Therefore the sequence of recombination node will either be only two substrings of both the parents, for single crossover, and sometimes contains many smaller substrings of both the parents, called multiple crossovers.

The similarity and dissimilarity measure gives important structural details about the gall trees. We will show that the nodes can be classified into mutation and recombination

nodes using the distance measure. We will construct the gall tree and prove that this algorithm will display minimum number of recombinations needed by the sequence matrix, which has all 0s in the ancestral sequence.

Lemma 2. *If a node v' is the result of mutation from its parent v then* $v < v'$ *, when the sequences are considered as the binary numbers.*

Proof. We prove this by mathematical induction on the length of the sequence *m*. In the first step consider a parent *v*, with sequence *S*, contains all 0's in its sequence. According to the definition of the mutation only one site can change the state from 0 to 1 and rest of the sequence remains same. If a mutation occurs at site *i* of *v* leading to at least on of the sites of the sites of sequence *S'* of the node *v'* is set to 1 and the rest of the sequence will remain same as the parent sequence. Thus making *S* less than *S'*, *S<S'*. Now consider the case where the node *v* has *m-2* number of 1's in sequence *S* a mutation leads to *m-1* number of ones in *S'* making *S<S'*. Now we prove it for a generalized case of *n-1* number of ones. If a node *v*, with sequence *S* having *m-1* number of ones mutates to result in new child node *v'* with sequence *S'* having *m* number of ones, which is the highest value binary number for a given length *m*. therefore $v < v'$ when mutation occurs. □

Lemma 3 plays an important role in the detection of the recombination nodes. It proves that if a node is the result of recombination then it should be greater than at least on of the parents. This fact is used for finding the other parent, which is less than its child.

Lemma 3. *Let v be a recombination node with sequence S. if P' and P" are two parent nodes of v, with sequences S' and S" respectively, then any of the following will hold.*

(a) S'>S and S">S

(b) S'>S and S"<S

(c) S'<S and S">S

Proof. It is enough to prove it for the binary sequences of length 2. Let three binary sequences, which give a recombination node *v* are 01, 10, 11. These sequences can be placed in only three different ways to represent the recombination as shown in Figure 2.

It is proved by Gusfield et al. [4] that these are the only possible ways of representing the above sequences. In all the cases the sequence with all 0s is considered as root.

Case (a): Here the two mutations from root node lead to the species 01 and 10. The taxon, node *v*, with sequence 11 is the result of recombination of 01 and 10. Clear *v* is greater than its two parents.

Case (b): In this case the sequence 01 muted from root and the sequence 11 is mutated from 01. The recombination node *v* is the result of recombination between root and *P'*. It satisfies the case (b) stating, *S'>S* and *S"<S*.

Case (C): In this case the sequence 10 muted from root and the sequence 11 is mutated from 10.

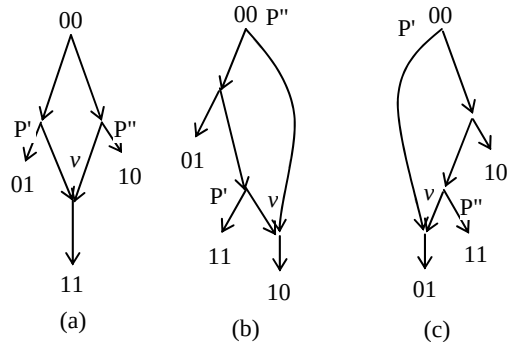

Figure 2. Three cases for lemma 3.2

The recombination node v is the result of recombination between root and *P"*. It satisfies the case (c) stating, *S'<S and S">S.* □

Theorem 1. *Let M be the given sequence matrix representing the gall tree. A species or sequence is said to the result of recombination if it holds any of the following conditions.*

(a) If two species have, 0<similarity<=100% and dissimilarity > (100/m) %, where m is the length of the sequence, one sequence represent parent and another sequence represent the child, which is the result of recombination.

(b) Parents of the recombination node are have similarity=0.

Proof (a) suppose that at some node *x*, mutation occurred at site *i*, representing the change at site *i* from 0 to 1 and rest of the sequence remains same. If we calculate it's similarity and dissimilarity corresponding to the value 1 at each site, the similarity will be *100%* and dissimilarity will be *100/m %* exactly. This indicates that only one site has modified its value from 0 to 1. By the assumptions we made for phylogenetic network, there is no provision for back mutation, that is transformation from 1 to 0, the mutation of more than one site at the same instance of time is also ruled out. This restricts the dissimilarity to be *100/m %* for mutation. But in case of recombination both restrictions of the mutation are ruled out due to the fact that the resulting sequence will carry a part of the sequence from its one parent and rest will be imitated from the other parent (in single crossover). This fact indicates that the similarity can be *0<similarity<=100%* and dissimilarity > *(100/m) %*. Hence the condition (a) is proved.

We prove this by contradiction. Suppose the recombination node v has two parents with the sequences *S'* and *S"*, show some similarity. From the assumptions made in section 2 and lemma 1, the similarity between the species is due to two reasons (1) common parent, (2) child parent relationship with a single mutation and (3) due to recombination. If *S'* and *S"* shows some similarity then any of the above relation holds. The relation 1 avoided by removing the parent

characteristics while computing the similarity between the children. We are focusing on a constrained recombination problem, where two recombination nodes are disjoint, avoid the relation 3. If *S′* and *S″* shares child parent relationship with mutation then the result of recombination will be a sequence *N*, similar to child or parent instead of the new sequence, $N \in (S', S'')$. It proves that the parents of recombination node are dissimilar to each other. □

Theorem 2. *If C, C ', and C" are child list of two sequences S, S', and S" then the following conditions should hold for the gall trees.*

(a) Recombination node will be $C \cap C' \neq \phi$ *and* $|C \cap C'| = 1$.

(b) *If* $C \cap C' \neq \phi$ *then* $C \cap C'' = \phi$ *and* $C' \cap C'' = \phi$.

Proof this is obvious. If child list is associated with every node indicating it's potential children, then any two nodes, having child list *C* and *C'*, shares a single common child, $|C \cap C'| > 1$, indicates the parents are involved in more than one recombination resulting into a non gall tree. This condition, (a), restricts two parents to have more than one recombination nodes.

Similarly the condition (b) states that if two nodes involve in a recombination they cannot have any other recombination with any other node. If two nodes shares a recombination node, $C \cap C' \neq \phi$, and if a new node with the child list *C"* occur which also shares a recombination node with *C* or *C'*, such that $C \cap C'' \neq \phi$ or $C' \cap C'' \neq \phi$. This indicates that any node which is parent of *C* or *C'* is a part of one more recombination cycle involving the parent of *C"*. This leads to the non-gall tree like situation. Hence condition (b) should also be satisfied for the gall trees. □

4. PHYLOGENETIC NETWORK RECONSTRUCTION ALGORITHM

In this section we propose an algorithm for the phylogenetic network reconstruction with constrained recombination. We also prove that this algorithm results in minimum number of galls in the resulting network. We conclude the section with an example for the algorithm.

4.1 The CRECOMBINATION Algorithm

The algorithm makes the child list of each node given in the data matrix based on similarity and dissimilarity. We assume that all the sequences represent a unique leaf node in the network. The arrangement of nodes start with the root node, assume to have all 0s in it. Each mutation will lead to change at only one site and recombination may lead to more than one change. Back mutation is not permitted. The parent node is considered as the root to all the nodes in the child list except the recombination node. If data does not represent gall tree the algorithm terminates by reporting an error message.

Data structures:

d ← *is an input matrix of size* $n \times m$ *,where n is number of species and m is length of sequences.*

sim_dis_{ij} ← *is similarity and dissimilarity matrix corresponds to 1s in the comparing sequences.*

Node is a record with three variables: Label, Count, and Type.

An array of child nodes labels for each node.

INPUT: - binary matrix of $n \times m$ *size.*

OUTPUT: - child list for each node.

ALGORITHM: CRECOMBINATION (*d*)

Sort the matrix by considering each row as binary number.

For each row in the input binary matrix **do**

Label ← *row _ value*; *Count* ← 0;

Type ← *Null*;

For each sorted node $i \leftarrow \{1 \le i \ge n\}$ **do**

$Child_i \leftarrow Null$;

For each node $j \leftarrow \{1 \le j \ge n\}$ **do**

if $sim\_dis_{ij} \leftarrow Null$ **then**

Compute Similarity and dissimlarity between *i* and *j*;

Modify *sim _ dis* matrix;

if $Node_i.Type = Null$ / *mutation* **then**

if *Similarity = 100%* and *Dissimilarity = 100/m %* **then**

$Node_j.Count \leftarrow Node_j.Count + 1$;

if $Node_j.Count > 2$ **then**

Exit "data does not represent gall tree";

else if $Node_j.Count = 2$ **then**

$Noide_j.Type = recombination$;

$Child_i \leftarrow Child_i \cup Node_j$;

else

$Noide_j.Type = mutation$;

$Child_i \leftarrow Child_i \cup Node_j$;

else if *Similarity < 100% and Dissimilarity ≥ 100/n %* **then**

$Node_j.Count \leftarrow Node_j.Count + 1;$

if $Node_j.Count > 2$ **then**

Exit "data does not represent gall tree";

$Noide_j.Type = recombination;$

$Child_i \leftarrow Child_i \cup Node_j;$

if $Node_i.Type = recombination$ **then**

if $Similarity \leq 100\%$ *and* $Dissimilarity \geq 100/n\ \%$ **then**

$Child_j \leftarrow Child_j \cup Node_i;$

$Node_i.Count \leftarrow Node_i.Count + 1;$

for each x, $1 \leq x \geq |Child_i|$, and $Node_x \in Child_i$ **do**

Compare $Node_i$ with other element of $Child_i$ after removing parents characteristics;

Modify *sim _ dis* matrix;

return;

4.2 Evaluation with an Example

The input matrix for the algorithm is shown in Figure 4(a), which consist of seven leaf nodes and a binary number represents each of the nodes. As the first step in the algorithm we sort the nodes considering each row represents a node and is a binary number. The sorted matrix is shown in Figure 3(b).

A	0	0	0	1	0
B	1	0	0	1	0
C	0	0	1	0	0
D	1	0	1	0	0
E	0	1	1	0	0
F	0	1	1	0	1
G	0	0	1	0	1

(a)

A	0	0	0	1	0
C	0	0	1	0	0
G	0	0	1	0	1
E	0	1	1	0	0
F	0	1	1	0	1
B	1	0	0	1	0
D	1	0	1	0	0

(b)

Figure 3. (a) Input binary matrix with labels. (b) Sorted binary matrix each row as a binary number.

Table 2. Values of each property of Node record after the processing input matrix shown in Figure 3(a)

Node Label	Type	Count
A	Null	0
B	Mutation	1
C	Null	0
D	Recombination	2
E	Mutation	1
F	Recombination	2
G	Mutation	1

After processing each node the values assigned to each variable or properties of the nodes are shown in Table 1. The values for nodes *A* and *C* are '*Null*' because they are mutated from the root node, not from any given nodes. The nodes *D* and *F* are the result of the recombination and have two parents. Rests of the nodes are the result of mutation from their respective parents.

	A	*B*	*C*	*D*	*E*	*F*	*D*
A	1,0	1,1	0,2	0,3	0,3	0,4	0,3
B	1,1	2,0	0,3	1,2	0,4	0,5	0,4
C	0,2	0,3	1,0	0,1	0,1	0,2	0,1
D	0,3	1,2	0,1	2,0	1,2	1,3	1,2
E	0,3	0,4	0,1	1,2	2,0	2,1	1,2
F	0,4	0,5	0,2	1,3	2,1	3,0	2,1
G	0,3	0,4	0,1	1,2	1,2	2,1	2,0

Figure 4. Similarity-Dissimilarity matrix for the input shown in Figure 4(a)

Table 1. Child list of each node for the binary input matrix

Node Label	Child List
A	B
B	D
C	D,E,F,G
D	Null
E	F
F	Null
G	F

The similarity-dissimilarity matrix computed during the detection of the recombination nodes is shown in Figure 4. The node *C* is parent for the nodes *D*, *E*, *F*, and *G*, so the similarity dissimilarity measure between the children is computed after removing the parent's characteristics.

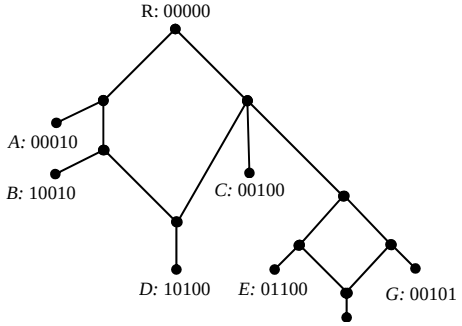

Figure 5. Gall tree for the given input matrix as shown in Figure 3(a).

Table 2 shows the child list of each node. The nodes *D* and *F* don't have any child so their list carries null value in it. On the other hand the nodes *D* and *F* are in the child list of *B, C* and *E, G* nodes respectively making them recombination nodes.

Finally a gall tree is constructed based on the assumption that the parent serves as the root to its child nodes. The final gall tree is shown in Figure 5.

Now we prove that the algorithm results in a gall tree, if one exists, with the minimum number of galls in it.

Theorem 3. *If the for the input matrix M there are k recombination nodes then any gall tree that minimizes the recombination will have exactly k recombinations.*

Proof. Let *T* be a gall tree for the input binary matrix *M*. If there is a gall *Q* in *T* that contain only the mutation nodes, then the sequence labeling the nodes on *Q* can be derived from the perfect phylogeny. The root of the phylogeny is the sequence of their parent. Replacing *Q* with perfect phylogeny will result in a gall tree with fewer recombinations than *T*. Hence in any gall tree using the minimum number of recombinations, every gall contains at least one node of type recombination. Therefore the number of galls is exactly the number of nodes with recombination type. □

4.3 Correctness and time complexity

All the results and facts in the above sections assume the existence of the gall tree for the input matrix *M*. The results in section 3 and 4 give the proof of correctness of the algorithm. When the input data does not display a gall tree structure, the algorithm reports an error message and terminates. The gall tree computed by the algorithm will have minimum number of recombinations.

The algorithm computes a gall tree, if one exists, in $O(n\log n + n^2 m)$ time, where *n* is number of nodes, and *m* is the length of the each binary number. The algorithm sorts the *n* rows, considering each row as a binary number, using quick sort, which takes $O(n\log n)$ time. In the next step the algorithm computes the similarity and dissimilarity between each of the node with respect to the sites with the value 1. The second step takes $O(n^2 m)$ time. On the basis of the similarity and dissimilarity measure the type of each node is decided, and at the end of each iteration the child list for a specific node is created. This child list can be further used to the construct the gall tree.

5. CONCLUSION

In this paper we proposed a pattern matching based approach for the construction of phylogenetic network with constrained recombination. The construction of perfect phylogenetic network is proved to be NP-hard by Wang et al. [4]. Wang et al. [4] gave a polynomial time algorithm for a restricted problem called gall trees or node disjoint network, in which a node can not be a part of two cycles in the network. It has both algorithmic and biological significance. The method in [4] computes the gall tree in $O(n^4)$ time. Guesfield et al. [5] proved that the [4] does not give the necessary and sufficient conditions for the gall tree construction and gave a sufficient combinatorial basis for gall tree construction. The method proposed in [5] used the conflict graphs as the major tool for the detection and construction of the gall tree. The algorithm computes the gall tree in $O(nm + n^3)$ time.

In this paper, we proposed an algorithm, which computes the gall tree in $O(n\log n + n^2 m)$ time and established the necessary and sufficient condition for the gall trees. Unlike the other algorithms, we followed a row-based search to detect the recombination nodes. Other algorithms search the columns for the detection of recombination. The number of columns in a sequence may be far greater than the row, which increases the complexity of the previous algorithms.

REFERENCES

[1] Posada, D., and Crandall, K. Intraspecific gene genealogies: trees grafting into networks. *Trends in Ecology and Evolution*, 16, (2001), 37–45.

[2] Schierup, M.H., and Hein, J. Consequences of recombination on traditional phylogenetic analysis. *Genetics*, 156, (2000), 879-891.

[3] Savai, P., Abulleef, H., Chun, L.L., and Skvortsov, D. Phylogenetic analysis. MS. Project. University of southern California, 2002.

[4] Wang, L., Zhang, K., Zhang L. Perfect phylogenetic networks with recombination. Journal of computational biology, 8, (2001), 69-78.

[5] Guesfield, D., Satish, E., and Langley, C. optimal efficient reconstruction of phylogenetic network with constrained recombination. *Journal of computer and system science*, 70, (2005), 381-398.

[6] Hein, J. A heuristic method to construct the history of sequences subject to recombination. *Journal of Molecular Evolution*, 36,(1993), 396-405.

[7] Schierup, M.H., and Hein, J., Recombination and the molecular clock. *Mol. Biol. Evol.*, 17, (2000), 1578–1579.

[8] Posada, D., and Crandall, K. The effect of recombination on the accuracy of phylogeny estimation. *Journal of molecular evolution*, 54, (2002), 396-402.