# A Pattern Recognition Based Approach for Phylogenetic Network Construction with Constrained Recombination

M. A. H. Zahid, Ankush Mittal, and R. C. Joshi

*Department of Electronics and Computer Engineering, Indian Institute of Technology Roorkee, INDIA*

**Abstract**

The tree representation of evolutionary relationship oversimplifies the view of the process of evolution as it cannot take into account the events such as horizontal gene transfer, hybridization, homoplasy and genetic recombination. Several algorithms exist for constructing phylogenetic networks which result from events such as horizontal gene transfer, hybridization and homoplasy. Very little work has been published on the algorithmic detail of phylogenetic networks with constrained recombination. The problem of minimizing the number of recombinations in a phylogenetic network, constructed using binary DNA sequences, is NP-hard. In this paper, we propose a pattern recognition based $O(n^2)$ time approach for constructing the phylogenetic network, where n is the number of nodes or sequences in the input data. The network is constructed with the restriction that no two cycles in the network share a common node.

*Key words:* Evolutionary relationship, Phylogenetic network, Recombination, SNP, Gall trees, Pattern recognition.

## 1 Introduction

The large volume of genetic data generated through biological experiments is not useful until it is analyzed and classified properly. A single human genome project ensues in 3.2 million base pairs of nucleotide sequence data. Manual analysis of this kind of huge data is impossible. The potential applications of analysis and classification provide valuable insights of the biological system. These insights are largely responsible for the development of the new crop in agriculture, and the discovery of new treatments for infectious disease [1], ecology [2] and computational biology [3]. The quest of classification of the genetic material leads to one of the oldest field of biology called phylogenetics.

Phylogenetics is the study of finding relationship among species or genes with the combination of molecular biology and mathematics. The tree structure used as the classical way of representing the evolutionary relationship between the species is also called as phylogenetic tree.

The phylogenetic tree construction methods fail to find true relationships between the species, not because the methods are inadequate to do so, nor because of wrong selection of genes, but because the evolutionary relationship cannot be represented as tree. The tree representation of evolutionary relationship oversimplifies the view of the process of evolution as they cannot take into account 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 [4,5]. Detection of recombination is very important because it locates the origin of the gene influencing a genetic disease. A case study on HIV [6] carried at University of Southern California has shown that the most frequent recombination event has made it difficult to design a drug for HIV. Recombination in HIV is recognized as an important mechanism by which the viruses escape the attack against the drug [6].

The recombination event originates from modeling mutation in DNA sequences [9]. For example, consider that 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 from another ancestor as shown in Figure 1. Thus the recombination event cannot be modeled with a tree structure. Instead it should be represented as network where some of the nodes may have two parents. The nodes with two parents are called as the recombination nodes.

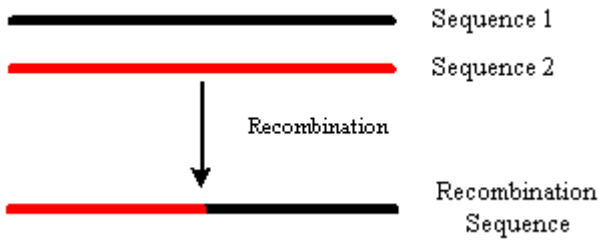



Fig. 1. The recombination event.

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 recombination are ignored, and phylogenies were constructed by neglecting the recombination events. Schierup and Hein [5,10] and Posada [11] have shown the effect of negligence of recombination while constructing the phylogeny. The effects shown by Schierup and Hein include the long ter-

minal branches in star like trees and that the rate of heterogeneity among the sites is wrongly inferred. Despite above facts, very little has been published on robust methods for recombination.

There exist several algorithms for the detection of the non-tree like events such as lateral gene transfer (horizontal gene transfer) [12,16], hybridization [15,16], homoplasy [16] and genetic recombination [9,16–18]. However, the computational aspects of the phylogenetic network with constrained recombination except the work by Wang et al. [7] and Gusfield et. al. [8,24] have been largely ignored. A comprehensive survey of different phylogenetic networks algorithms is given in [21,22].

Wang et al. [7] showed the problem of finding a perfect phylogenetic network, a network with minimum number of recombination nodes, is NP-hard. They gave an algorithm for a restricted problem, called node disjoint network, with $O(n^4)$ computing time, where $n$ is number of leaf nodes or species. The restriction was that in the merged path of the recombination node, there is no node that is in the merge path of a different recombination node. In other words, no node can be shared by two recombination cycles, if underlying undirected graph has cycles. A recombination cycle that shares no node with any other recombination cycle is called a "gall" and a phylogenetic network with disjoint recombination cycles is called a "gall tree" [8]. Gusfield et al. [8] showed that the algorithm in [7] is incomplete and does not constitute the necessary test conditions for the existence of the gall tree. The most efficient solution for the gall tree problem, till date, is given in [8]. It 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 to the final gall tree. The algorithm can compute the gall tree, if one exists, in $O(nm + n^3)$ time, where $m$ is the length of the binary sequence [8]. In this paper, we propose an efficient algorithm for the construction of the gall tree. We use the similarity and dissimilarity between the binary sequences as a major tool for detecting and constructing the gall tree. In addition to using similarity we demonstrate how dissimilarity measure can be effectively used. We employ pattern recognition technique in the algorithm to computes a gall tree, if one exists, in $O(n^2)$ time.

Pattern recognition has emerged as a major tool for bioinformatics applications such as DNA sequence analysis and DNA Microarray analysis [19]. In general, given a new DNA sequence, it is compared with the sequences that has already been studied and analyzed. The sequences that are similar would probably have similar functional and structural properties in case of genes and proteins. Relationship between the homologous sequences has an important implication in phylogenetics. Sequence alignment is one of the methods of sequence comparison, similar to the string matching, which is studied extensively in pattern recognition. Sequence alignment is the procedure for comparing two

or more sequences by searching a series of individual characters or character patterns in the sequences. The distance based phylogenetic methods make use of the dissimilarity between the sequences to find the evolutionary relationships between the species. We use both similarity and dissimilarity for the classification of the nodes into mutation and recombination nodes. A similar approach for the clustering of symbolic objects is given in [20].

The paper is organized as follows. Section 2 deals with the formal definitions and assumptions related to phylogenetic networks. Section 3 deals with the combinatorial background and conditions for the detection of recombination. The algorithm for the construction of the network with an example is given in section 4. Section 5 gives the results and comparisons with the existing methods.

## 2 Preliminaries

This section deals with the basic terminology and assumptions made for the development of the algorithm. We follow the terminology given in [8,24] for simplicity.

A phylogenetic network is a directed acyclic graph, however, the 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 due to the recombination of genetic material of two parent species of the node. A special node with indegree 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 in their sequences, from 0 to 1, when compared with their parent nodes. The sequence of a recombination node is part of its two ancestor's sequences.

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 $v$. Each node in the phylogenetic network is represented with a binary number of some specified length $m$, each element in the array of $m$ binary numbers is called a site or characteristic. The node with all zeros in its sequences is called the root of the network $N$. In the perfect phylogeny the transformation of a site from 0 to 1 occurs at most once for each site or the column in the binary sequence. The nodes on perfect phylogenetic networks are organized in such a way that there is a unique node having state 1 in site $i$ whereas every other node having state 1 at site $i$ is descendent of this unique node. The transformation from 0 to 1 is possible in case of recombination, where the crossovers can change the state from 0 to 1. A phylogenetic network with recombination is said to be perfect if it

has minimum number of recombination nodes and follows all the restrictions mentioned above.

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 2. The biological interpretation of a phylogenetic network $N$ for $n$ sequences represents the possible history of the 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 [23].

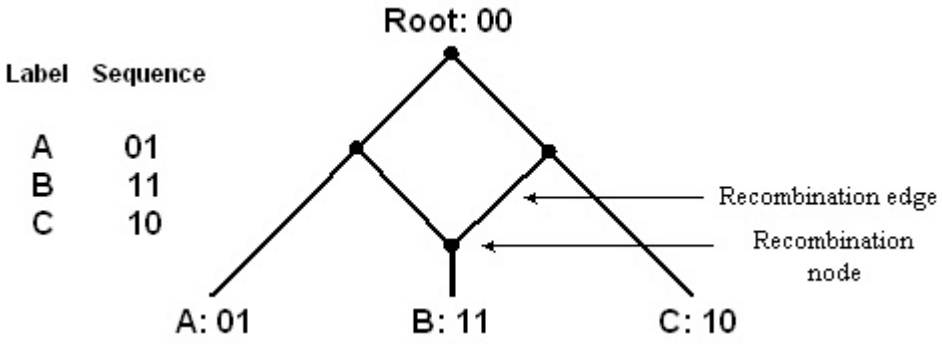



Fig. 2. A Phylogenetic network for three binary sequences.

Given a set of species and their binary sequences, a perfect phylogenetic network, always exist with $O(mn)$ 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 node $x$ in a phylogenetic network $N$ is said to be coalescent if it has two paths out of it that meet at recombination node $r$. Those two paths together with the coalescent node and recombination node represent a recombination cycle.

A recombination cycle in a phylogenetic network that does not share any node with any other recombination cycles is called a gall. But the path from root to any of the gall or a node, which is not on the gall, can pass through the gall. A phylogenetic network is said to be gall tree if every recombination cycle is a gall in the network.

A simplified constrained recombination network with recombination, coalescent and mutation nodes is shown in Figure 3.

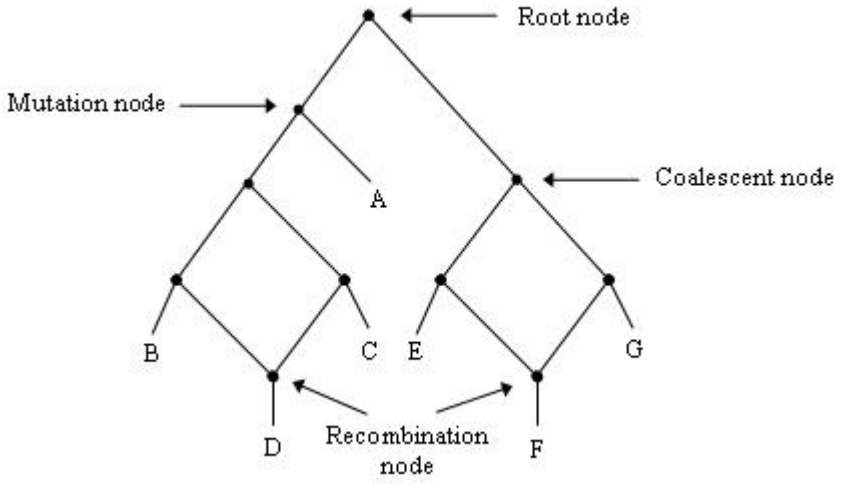



Fig. 3. A gall tree with recombination, coalescent and mutation nodes.

## 3 Conditions for the detection of recombination nodes and construction of Gall tree

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

Lemma 1 is crucial for the detection of the gall in the given binary sequences. It says that the similarity and dissimilarity between the sequences, which share a common parent should be computed after removing the parent's characteristics from each 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 $S$, then $S'$ is not reachable from $S$, therefore, all the sites or characters of $S'$ are different from the characters of the node $S$. Let $S$ and $S'$ are children of node $v$, then both $S$ and $S'$ are reachable from node $v$, and show similarity by at least one character (or site) with the parent node $v$. Therefore both the nodes $S$ and $S'$ show the similarity with each other at the parent characters or sites. Hence, this proves that the distinct nodes will show similarity due to common ancestry. □

A set of binary sequences 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 has 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. As recombination is the process of exchanging the genetic material between the species, the sequence of recombination node will either be only two substrings of both the parents (for single crossover) or many smaller substrings of both the parents (called multiple crossovers).

The similarity and dissimilarity measures give important structural details about the gall trees. We show that the nodes can be classified into mutation and recombination nodes using the distance measure. We construct the gall tree and prove that this algorithm displays minimum number of recombinations needed by the sequence matrix, which has all 0's 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 of lemma 2 is quite obvious as only one site is changed from 0 to 1 during mutation and the back mutation is not permitted.

Lemma 3 plays an important role in the detection of the recombination nodes. It states that if a node is the result of recombination then it should be greater than at least one of the parents. This can be used to classify the input sequences in mutation and recombination classes.

**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** Let us consider the proof for the binary sequences of length two. 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 4. It is proved by Gusfield et al. [8] that these are the only possible ways of representing the above sequences. In all the cases, the sequence with all 0's is considered as root. The case of child having less value than both of its parent is not possible because back mutation is not permitted and the assumption that the root of the gall tree will have all 0's in it. Let us now consider the cases (a), (b), and (c)individually below:

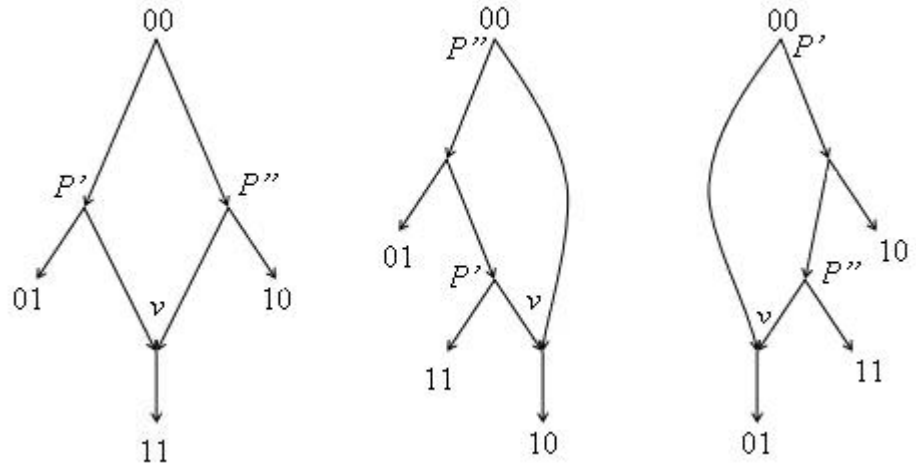



Fig. 4. Three cases for Lemma 3.

(a) Here, the two mutations from root node lead to the species 01 and 10. The recombination node $v$, with sequence 11 is the result of recombination of 01 and 10. Clearly, $v$ is greater than its two parents. This is shown in leftmost network of the Figure 4.
(b) In this case, the sequence 01 mutated 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 $S" < S$. This is shown in middle network of the Figure 4.
(c) In this case, the sequence 10 is mutated from root and the sequence 11 is mutated from 10. The recombination node $v$ is the result of recombination between root and $P"$. It satisfies the case (3) stating, $S' < S$ and $S" > S$. This is shown in rightmost network of the Figure 4.

It can be easily seen that the proof follows for binary sequences of length greater than two. □

Converting the binary sequences into distance matrix based on similarity or dissimilarity leads to the information loss. For example, consider the sequences $S_1$: 00010, $S_2$: 00100, $S_3$:10010 and $S_4$:10100. The distance or dissimilarity between $S_1$ and $S_2$ is 2, and the distance between $S_3$ and $S_4$ is also 2. According to the definitions of child parent relationship mentioned in section 2 and 3, the sequences $S_1$ and $S_2$ do not share any relationship with each other. On the other hand, the sequences $S_3$ and $S_4$ share a child parent relationship even though they have the same dissimilarity, i.e., 2 as for $S_1$ and $S_2$. This clearly indicates loss of information due to transition. Its effects when applied to different methods, which construct the network based on distance data, are shown in section 5. To avoid this loss, we consider both similarity and dissimilarity for the construction of gall trees.

The theorem 4 uses the lemma 2 and 3 for the detection of the recombination nodes. It helps in finding the parents of the recombination node, which is particularly useful when any one of the parent is greater than the child. It states that the similarity between the parents of the recombination node is 0. Given three nodes indicating a recombination event, using lemma 2 and lemma 3 we can find one parent easily and the other parent can be can be detected with aforementioned fact.

**Theorem 4** *Let M be the given sequence matrix representing the gall tree. A species or sequence is said to be a result of recombination if any one of the following conditions holds good:*

*(1) If two species have* $0 < similarity <= 100\%$ *and dissimilarity* $> (100/m)\%$*, where m is the length of the sequence, one sequence represents parent and other sequence represents the child, which is the result of recombination.*
*(2) Parents of the recombination node have similarity* $= 0$*.*

**Proof**

(1) Suppose that at some node $x$, mutation occurred at site $i$, which represents the change at only site $i$ from 0 to 1 in the mutated node $y$. If we calculate the similarity and dissimilarity between the nodes $x$ and $y$ corresponding to the value 1 at each site, the similarity between them will be 100% and dissimilarity will be $100/m\%$. 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 and in addition, the mutation of more than one site at the same instance of time is ruled out. This restricts the dissimilarity as $100/m\%$ for mutation. But in case of recombination the two aforementioned restrictions of the mutation are ruled out due to the fact that the resulting sequence will carry a part of the sequence from one of its parents and rest will be copied 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 (1) holds true.
(2) Let us prove this by contradiction. Suppose the recombination node $v$ has two parents with the sequences $S'$ and $S$" showing some similarity. From the assumptions made in section 2 and lemma 1, the similarity between the species in node disjoint recombination networks is due to following three reasons: (1) common parent, (2) child parent relationship with a single mutation, and (3) recombinations. If $S'$ and $S$" show some similarity then one of the above relations holds. The relation 1 is eliminated by removing the parent characteristics while computing the similarity between the children. Since we focus on a constrained recombination problem, where two recombination nodes are disjoint, the relation 3 is avoided.

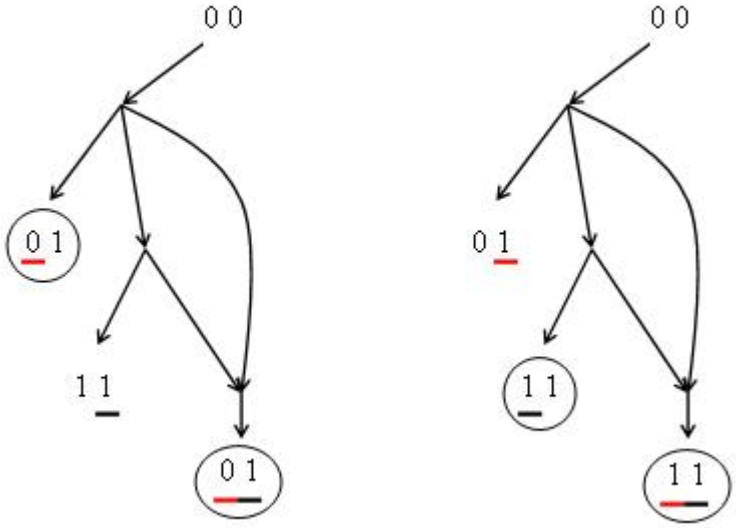



Fig. 5. Result of recombination between parent and child.

Let $S$" is an immediate child of $S'$, due to mutation in $S'$. If the nodes $S'$ and $S$" are the parents for recombination node $N$, then the sequence of recombination node will be same as any of its parents,$N \in (S', S")$ , instead of a new sequence as shown in Figure 5. Hence, the parents of the recombination node will be dissimilar to each other. □

Theorem 5 gives a strong basis for the detecting the galls in the given input data. Any data satisfying the conditions given in theorem 2 will have gall tree. Otherwise, the data will not represent the gall tree structure.

**Theorem 5** *If C, C', and C" are child list of nodes S, S', and S" then the following conditions should hold for the gall trees:*

*(1) Recombination node will be* $C \cap C' \neq \emptyset$ *and* $|C \cap C'| = 1$ .
*(2) If the number of recombination nodes in any of the parents is greater than 1, and* $C \cap C' \neq \emptyset$ *then* $(C \cap C" = \emptyset$ *or* $S" \cup C" \subseteq C)$ *and* $(C' \cap C" = \emptyset$ *or* $S" \cup C" \subseteq C')$ .

**Proof**

(1) This is proved by contradiction. Let $|C \cap C'| > 1$ represents that the nodes $S$ and $S'$ are involved in more than one recombination with each other. Each recombination node represents a cycle in the gall tree. The path from root node to the recombination node always has two alternatives, one from each of its parents. If there are more than one recombination node for the single pair of parents $S$ and $S'$, then there are two paths for

each recombination node which involves the same set of parents $S$ and $S'$. In other words, the parent nodes are shared by two recombination cycles. But according to the definition of gall tree, the nodes in one cycle should not be shared with other recombination cycle. This contradicts the assumption made and hence proves the condition.

(2) The proof is similar as in case (1). Let $C \cap C' \neq \emptyset$ and the number of recombination nodes in $C$ is two. If $C \cap C' = X$ and $C \cap C" = Y$, then there exists a path from root node to the recombination cycle of node $X$ and node $Y$, which passes through the node $S$. This violates the node disjoint rule of gall trees. Let child list $C"$ and the node $S"$ itself be a subset of the child list of node $S$. If the similarity and dissimilarity between the children of $S$ are computed after removing the $S$'s characteristics from each node then the recombination node will not show $S$ in its parent list, according to lemma 1. This parent relationship with the recombination node is due to the common ancestor of all the nodes in the recombination cycle. Thus, when there is common ancestor for the parents of a recombination node then the parent of all the nodes in that cycle is also added as the parent to the recombination node. □

If child list is associated with every node that indicates its potential children, then any two nodes, having child list $C$ and $C'$, share more than one child, i.e., $|C \cap C'| > 1$, and it indicates that the parents are involved in more than one recombination resulting into a non gall tree structure. The condition (1) in theorem 5 restricts the two parents from having more than one recombination node.

Similarly, the condition (2) states that only the node, which is parent of the two or more galls in the networks can have more than one recombination nodes in its child list.

## 4 Phylogentic network reconstruction algorithm

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

The root sequence contains all zeros in it and only a state change from 0 to 1 gives information about mutation or recombinations. Thus the similarity and dissimilarity with respect to 1 are considered. The similarity and dissimilarity with respect to 1's in the sequence can be calculated using AND and XOR

binary number operations respectively. For example, consider the sequences $S_1$ and $S_2$, having zero similarity and 2 dissimilarities as shown below.

| | | |
|---|---|---|
| 0 0 0 1 0 | : $S_1$ | |
| 0 0 1 0 0 | : $S_2$ | |
| 0 0 0 0 0 | AND | (Similarity) |
| 0 0 1 1 0 | XOR | (Dissimilarity) |

The number of 1's in the result represents similarities or dissimilarities with respect to the operation performed. Similarly, the parent's characteristics can be eliminated from children using the XOR operation. This approach significantly reduces the number of comparisons performed to compute the similarity and dissimilarity, hence reduces the complexity of the algorithm.

### *4.1 The Node Class_Algorithm*

The algorithm makes the child 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 starts with the root node, assumed 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.

The algorithm accepts a $n \times m$ binary matrix as input, where each row represents a node in the phylogenetic network. Similarity and dissimilarity matrices are generated based on the input matrix and are computed corresponding to the value 1 at the sites. The distances (similarity and dissimilarity) between the siblings are measured after removing the parent's characteristics from the children. The parent node is considered as the root to all the nodes in the child list except the recombination node. If the data does not represent gall tree, then the algorithm terminates by reporting an error message. We use a special data structure $Node$ with three variables $Lable$, $Type$ and $Count$. $Label$ is used to represent node label, $Type$ for representing the type of the node, and $count$ is used to represent number of parents of the node. There are three types of node $Recombination$, $Mutation$ and $Null$. $Recombination$ is assigned to the node which is the result of recombination, $Mutation$ is assigned to the node which results from mutation, and $Null$ is used to represent the node, which is the child of the root node or is not a $Mutation$ or $Recombination$ node.

If the data does not represent gall tree the algorithm terminates by reporting an error message. The algorithm is as follows:

*DATA STRUCTURE:*
*Input* : Binary matrix of $n \times m$.
*Output* : Parent and Child list for each node.
$D \longleftarrow$ An input matrix of size $n \times m$ ,where $n$ is number of species and $m$ is length of sequences. Each row is considered as a Node
$Sim_{ij} \longleftarrow$ The similarity with respect to 1's on comparing rows $i$ and $j$.
$Dis_{ij} \longleftarrow$ The dissimilarity with respect to 1's on comparing rows $i$ and $j$.
$Node \longleftarrow$ A record with three variables: Label, Count, and Type. Initially the Count and Type variables of a $Node$ are set to zero and Null respectively. The variable Label is set to the labels of rows in the input matrix.
$Child_i \longleftarrow$ An array of child labels for $Node_i$ initially set to $Null$.

ALGORITHM: $Node\_Class(D)$

**Begin**
1. **for** each Row (Node) $i$ in $D$ **do**
2. **for** each Row (Node) $j$ in $D$ **do**
3. **if** $0 < sim_{ij} \leq 100\%$ and $Dis_{ij} \geq 100/m\%$ **then**
4. **if** $Node_i < Node_j$ **then**
$prnt \leftarrow i$
$chld \leftarrow j$
**else**
$prnt \leftarrow j$
$chld \leftarrow i$
**end if**(4)
$Node_{chld}.Count \leftarrow Node_{chld}.Count + 1$
$Child_{prnt} \leftarrow Child_{prnt} \cup Node_{chld}.Label$
5. **if** $0 < Node_{chld}.Count \geq 2$
$Node_{chld}.Type \leftarrow Recombination$
**else**
$Node_{chld}.Type \leftarrow Mutation$
**end if**(5)
**end if**(3)
**end for**(2)
**end for**(1)
**end Algorithm**

The next function *Test_Gall* takes Child list and Node record list as input and based on theorem 5, it verifies whether gall tree exists in the given data. If the data does not represent the gall tree then the function terminates giving an error message as an output, otherwise it returns a message confirming presence of gall tree. The function is as follows:

FUNCTION: $Test\_Gall(Child, Node)$

**Begin**
1. **for** each Row (Node) $i$ in $D$ **do**
2. **for** each Row (Node) $j$ in $D$ **do**
3. **if** $|Child_i \cap Child_j| > 1$ **then**
**return** Gall tree does not exist
**end if**(3)
**end for**(2)
**end for**(1)
4. **for** each Row $l$ with more than one recombination node as child **do**
5. **for** each child $c$ of $child_l$ **do**
6. **if** $Node_c.Label \cup Child_c \subseteq Child_l$
Add a new node in the Child list same as $Node_c$
Add the Children of $Node_c$ to the new node
Remove the children of $Node_c$ from Child list of $Node_l$
Add the new $Node$ as the child of $Node_c$
**else**
**return** Gall tree does not exist
**end if**(6)
**end for**(5)
**end for**(4)
**end Function**

### *4.2 An Example*

The input matrix for the algorithm is shown in Figure 6, which consists of seven leaf nodes and a binary number to represent each of the nodes. The initial value for the variables of each Node record is set to Null. The similarity and dissimilarity matrix after removing the parent's characteristic are shown in Figure 7.

| Label | Sequence |
|---|---|
| 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 |

Fig. 6. Input binary matrix with labels.

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

Fig. 7. Similarity and dissimilarity matrix for the input data shown in Figure 6.The first element represents similarity and second one represents dissimilarity.

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, and not from any other nodes. The nodes D and F are the result of the recombination and have two parents. The rest of the nodes are the result of mutation from their respective parents.

Table 1
Values of each property of node record after processing the input matrix shown in Figure 6.

| Node Label | Type | Count |
|---|---|---|
| A | Null | 0 |
| B | Mutation | 1 |
| C | Null | 0 |
| D | Recombination | 2 |
| E | Mutation | 1 |
| F | Recombination | 3 |
| G | Mutation | 1 |

Table 2 shows the child list of each node. The nodes $D$ and $F$ do not have any child, thus their child lists carry $Null$ values. On the other hand, the nodes $D$ and $F$ are in the child list of $B$, $C$ and $E$, $G$ nodes respectively; thus making them recombination nodes. The function $Test\_Gall$ results in Table 3 when applied to Table 2 and $Node$ list. The child list is computed based on the gall tree conditions proved in theorem 5. The child list for the node $C$ has two recombination nodes $D$ and $F$, and the child $F$ has count value 3 indicating three parents. But, it satisfies second condition in theorem 5, the steps 4,5,and 6 of function $Test\_Gall$ are executed, which results in a new node $C'$. The node $C'$ is added to child list of $C$. The child nodes $E$,$F$, and $G$

are removed from the child list of node $C$ and added as the children of node $C'$. The modified child list is shown in Table 3.

Table 2
Child list for the input matrix shown in Figure 6.

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

Table 3
Modified Child list for the input matrix shown in Figure 6.

| Node Label | Child List |
|---|---|
| A | B |
| B | D |
| C | D,C' |
| C' | E,F,G |
| D | NULL |
| E | F |
| F | Null |
| G | F |

Given the child and node record list for each of the node in the input data, it is easy to construct the gall tree for it. The procedure starts by including a new node called Root, with all zeros in it's sequence. All the nodes with a *Null* entry in their *NodeType* are attached to the root node. The rest of the nodes are then added and connected accordingly to the child list entries. The child list is preprocessed and analyzed for coalescent node, therefore, the phylogenetic network can be constructed directly with the help of child list and node record itself. It is obvious that the network can be constructed with a single scan of child list and node record. Both the lists contain $n$ entries and can be computed in linear time, i.e., $O(n)$.

The network construction for the above example starts with the first entry in the node record. Here, the first node is A which has a *Null* entry in its *Node.Type* as shown in Table 1, so it is attached to the Root node of the gall

tree. Node A's child list shows B as child, therefore, the node $B$ is attached to node $A$ as child. The nodes $B$ and $C$ have the node $D$ as child, therefore, node $D$ is attached to both of them. The whole network is constructed in similar fashion. The final gall tree for the input shown in Figure 6 is shown in Figure 8.

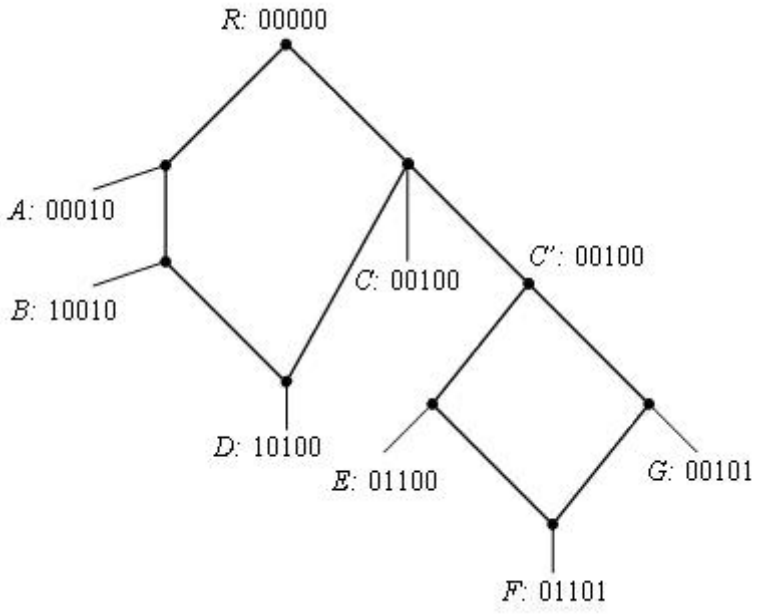



Fig. 8. Final gall tree for the input data shown in Figure 6.

The following theorem proves that the algorithm results in a gall tree, if one exists, with the minimum number of galls in it.

**Theorem 6** *For the input matrix M, if there are k recombination nodes then any gall tree that minimizes the recombination will have exactly k galls or node disjoint cycles.*

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

### *4.3 Correctness and time complexity analysis of the algorithm*

All the results and facts in sections 4 are based on the existence of the gall tree for the input matrix $M$. The theorems in section 3 and 4 show 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 have minimum number of recombinations.

The algorithm proposed in this paper computes a gall tree, if one exits, in $O(n^2)$ time, where $n$ is the number of nodes in the input data. The complexity is reduced with the help of small bookkeeping, which maintains a child list and a record for each node. There are three phases for finding a gall tree.

In the first phase, the similarity and dissimilarity matrices are computed based on the binary AND and XOR operations. The comparison is considered as a major operation contributing to the complexity of the computation and counting the number of ones in the result is taken as an elementary operation, this phase takes at most $O(n^2)$ time to compute the similarity and dissimilarity.

In the second phase, the nodes are classified into three types $Null$, $Mutation$, and $Recombination$ using $Node\_Class$ algorithm. The first two steps for loops, i.e., step 1 and 2, are major steps contributing towards the complexity of the algorithm and thus the steps take $O(n^2)$ time.

The final phase is used to find a gall tree based on child list and node record using $Test\_Gall$ method. The first two for loops, i.e., steps 1 and 2, run for $n$ times and takes $n^2$ time. The step 4 runs for each node with more than one recombination node, which is far less than the total number of nodes, i.e., $n$, and the step 5 runs for each child and its child list which is also less than the total number of nodes. The first two steps dominate the complexity of the $Test\_Gall$ method and lead to $O(n^2)$ time for the algorithm.

The total time required to compute the gall tree, if one exist, is $O(n^2+n^2+n^2)$ which is $O(n^2)$. This gives the better computing time than the best known algorithm [8] with time complexity of $O(n^3)$.

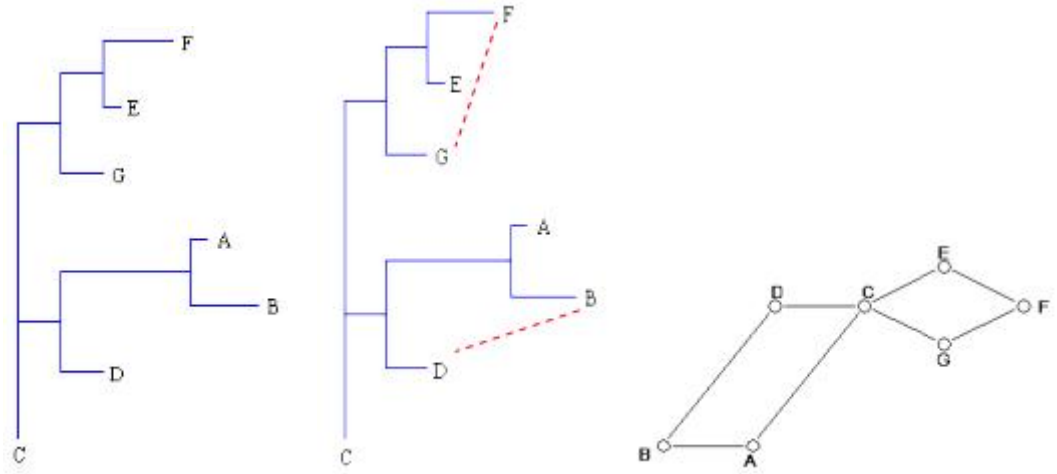
## 5 Results and discussion

In this section, we compare the results of the proposed method with the existing methods, such as T-Rex [25], NeighborNet [15], and SplitsTree [26]. The comparison is made with these methods based on the resulting network and the time complexity.

The distance matrix used for the input to the different methods is shown in Figure 9. The distance between the nodes represent the number of mismatches in the sequences. For example, consider the sequences $S_1$:00010 and $S_2$:10010 the distance is 1 as only one site has a mismatch and all other sites have the matching values. When the input matrix is applied to different network construction methods the resulting networks are shown in Figure 10.

| | A | B | C | D | E | F | G |
|---|---|---|---|---|---|---|---|
| A | 0 | 1 | 2 | 3 | 3 | 4 | 3 |
| B | 1 | 0 | 3 | 2 | 4 | 5 | 4 |
| C | 2 | 3 | 0 | 1 | 1 | 2 | 1 |
| D | 3 | 2 | 1 | 0 | 2 | 3 | 2 |
| E | 3 | 4 | 1 | 2 | 0 | 1 | 2 |
| F | 4 | 5 | 2 | 3 | 1 | 0 | 1 |
| G | 3 | 4 | 1 | 2 | 2 | 1 | 0 |

Fig. 9. The distance matrix used as input to T-Rex, NeighborNet, and SplitsTree.




(a) (b) (c)

Fig. 10. (a)Underlying tree for T-Rex, constructed using Neighbor Joining method, (b) T-Rex phylogenetic network, (c) Phylogenetic network by NeighborNet and SplitsTree.

Figure 10(a) shows the tree constructed using Neighbor Joining method and is used as the underlying tree structure by T-Rex for constructing phylogenetic network, shown in Figure 10(b). There is a reticulation edge between the nodes $B$, $D$ and $G$, $F$. The nodes $E$ and $F$ have evolved from a common ancestor and $F$ shares a reticulation edge with $G$. This indicates that the node $F$ is the result of reticulate evaluation. However, in the actual data $F$ is the result of recombination between $E$ and $G$. It can be observed that the

other reticulation edge is missing between $E$ and $F$. Similarly, a reticulation edge is missing between $C$ and $D$ as the node $D$ is the result of recombination between $B$ and $C$.

Figure 10(c) shows the network constructed using NeighborNet and SplitsTree, both of which give the same result. NeighborNet and SplitsTree give closer visualization to actual representation than T-Rex does. The evolution of node $F$ is properly indicated as the recombination node of $E$ and $G$. The evolution of $D$ is also interpreted properly. However, NeighborNet and SplitsTree added an additional edge between the node $A$ and $C$. In actual representation there is no relationship between A and C as they have evolved independently from root node. Thus, the output is not fully correct for these algorithms.

The variation in the networks is due to the information loss while converting the sequences into distances. To avoid this our algorithm utilizes both similarity and dissimilarity measures for the construction of the network. This gives a nearest visualization of the data to the actual representation. The resulting network of the proposed method is shown in Figure 8. Our algorithm gives the correct output as verified by the actual data representation.

The time required to compute the phylogenetic network by SplitTree is $O(n^5)$, T-Rex is $O(n^4)$ and NeighborNet is $O(n^3)$, where $n$ is the number of node. On the other hand the algorithm proposed in this paper computes the phylogentic network in $O(n^2)$ time.

## 6 Conclusion

In this paper we proposed a pattern matching based approach for the construction of phylogenetic network with constrained recombination. The proposed algorithm, computes the gall tree in $O(n^2)$ time. We also formulated the necessary and sufficient condition for constructing 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. The comparison with the result of other algorithms like T-Rex, NeighborNet, and SplitsTree shows that our algorithm is accurate and efficient.

## References


[1] P. H. Harvey, and S. Nee, Phylogenetic epidemology lives, *Trends Ecol. Evol.* **9** (1994) 361–363.

[2] D. R. Brooks, and D. A. McLennan, Phylogeney, Ecology, and Behavior. *Chicago, IL: Univ. of Chicago Press.* (1991).

[3] P. H. Harvey, and M. D. Psgel, The comparative method in evolutionaly biology, *Oxford: Oxford univ. press.* (1991).

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

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

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

[7] L. Wang, K. Zhang, and L. Zhang, Perfect phylogenetic networks with recombination, *Journal of Computational Biology.* **8** (2001) 69–78.

[8] D. Gusfield, E. Satish, and C. Langley, optimal efficient reconstruction of phylogenetic network with constrained recombination, *Journal of bioinformatics and computational biology.* **2** (2004) 173–213.

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

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

[11] D. Posada, and K. Crandall, The effect of recombination on the accuracy of phylogeny estimation, *Journal of Molecular Evolution.* **54** (2002) 396–402.

[12] M. T. Hallet, and J. lagergren, Effieceint algorithms for lateral gene transfer problems, *In proceedings of 5th international conference on computational molecular biology (RECOMB01), New York, ACM press.* (2001) 149–156.

[13] L. Addario-Berry, M. Hallett and J.Lagergren, Towards identifying lateral gene transfer events,*PSB.* **8** (2003) 279–290.

[14] A. Boc and V. Makarenkov, New Efficient Algorithm for Detection of Horizontal Gene Transfer Events, *In Algorithms in Bioinformatics, Springer, WABI.* (2003) 190–201.

[15] D. Bryant and V. Moulton, Neighbor-Net: an agglomerative method for the construction of phylogenetic networks, *Mol Biol Evol.* **21** (2004) 255–265.

[16] V. Makarenkov and P. Legendre, From a phylogenetic tree to a reticulated network, *J. Comput Biol.* **11** (2004) 195–212.

[17] J. Hein, Reconstructing evolution of sequences subject to recombination using parsimony, *Math. Biosci.* **98** (1990) 185–200.

[18] J. D. Kececioglu and D. Gusfield, Reconstructing a history of recombinations from a set of sequences, *Symposium on Discerete Algorithms (1994)* 471–480.

[19] A. W. Liew, H. Yan, M. Yang, Pattern recognition techniques for the emerging field of bioinformatics: a review, *Journal of Pattern Recognition.* **38** (2005) 2055–2073.

[20] K. C. Gowda, and T.V. Ravi, Divisive clustering of symbolic objects using the concept of both similarity and dissimilarity, *Journal of pattern recognition.* **28** (1995) 1277–1282.

[21] C. R. Linder, B.M.E. Moret, L. Nakhleh, and T. warnow, Reconstructing networks part II: computational aspects, *A tutorial presented at the ninth pacific symposium on Biocomputing (PSB 2004).*

[22] M. A. H. Zahid, A. Mittal, R.C. joshi, Use of phylogenetic networks and its reconstruction algorithms, *Journal of Bioinformatics India, ISSN 0972-7655.* **4** (2005) 47–58.

[23] A. Chakravarthi, It's raining SNP's hallelujah?, *Nature Genetics.* **19** (1998) 216–866.

[24] D. Gusfield, E. Satish, and C. Langley, The fine structure of galls in phylogenetic networks, *INFORMS J. on computing, special issue on Computational Biology.* **16** (2004) 459–469.

[25] V. Makarenkov, T-Rex: reconstructing and visualizing phylogenetic trees and reticulation networks, *Bioinformatics.* **17** (2001) 664–668.

[26] D. H. Huson, SplitsTree: A program for analyzing and visualizing evolutionary data, *Bioinformatics.* **141**(1998) 68–73.