

## Lecture 7: Triplet Methods

Date-September-24,2004

Lecturer: Wing-Kin Sung

Scribe: Cheng Weiwei, Yang Liang

### 7.1 Introduction

Due to some evolutionary events such as recombination, horizontal gene transfer, or hybrid speciation which suggest convergence between objects, cannot be adequately described in a single tree structure, in which every node has only one parent. In this situation, phylogenetic networks will be used. A phylogenetic network is a generalization of a phylogenetic tree in which internal nodes are allowed to have more than one parent. In fact, these types of events occur frequently in the evolutionary history of certain groups of organisms. Hence, to develop conceptual tools and efficient methods for computing with phylogenetic networks is an important issue.

### 7.2 Triplet method for phylogenetic tree

Rooted phylogenetic tree can be reconstructed using Triplets.

#### 7.2.1 What is a triplet?

A *rooted triplet* is a binary, rooted, unordered tree with three distinctly labeled leaves, which is also a phylogenetic tree.

For example,  $(\{b, c\}, e)$  is a rooted triplet shown in Figure 7.1(a).

We say that a triplet  $t$  is consistent with a tree  $T$  if  $t$  is an induced subtree of  $T$ . Figure 7.1(b) shows a tree  $T$ , with which the triplet  $(\{b, c\}, e)$  is consistent. The red edges in the figure indicate that  $t$  is an induced subtree of  $T$ , and thus  $t$  is consistent with tree  $T$ .

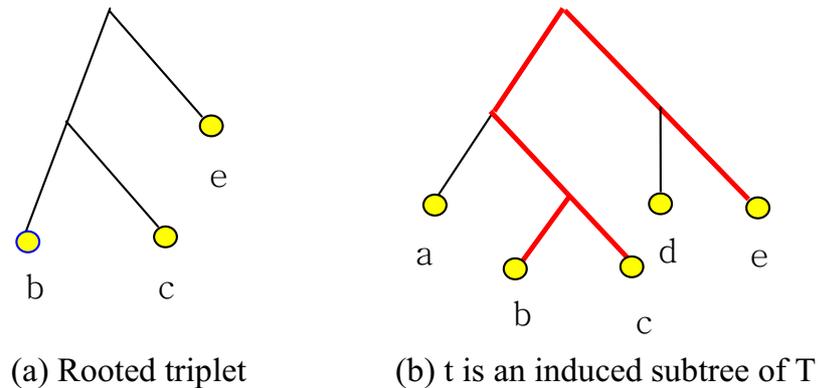


Figure 7.1 Rooted triplet

### 7.2.2 Construct a phylogenetic tree by Triplet Problem

A Triplet problem is defined as follows:

Given a set of  $k$  rooted triplets  $S$ , we would like to find a rooted tree  $T$  such that  $S$  is consistent with  $T$ , if exists.

For example, the rooted triplet  $S = \{(\{b,c\},a), (\{a,c\},d), (\{d,e\},b)\}$  is consistent with the tree  $T$  shown in Figure 7.1 (b).

This triplet problem is proposed by Aho, Sagiv, Szymanski, and Ullman (1981). Its original application is on database for optimizing the relational expression. By the triplet problem, rooted phylogenetic tree can be reconstructed, since triplets can be constructed accurately by using maximum likelihood method or by Sibley-Ahlquist-style DNA-DNA hybridization experiment directly. With those triplets, we can get the rooted phylogenetic tree by applying the triplet problem, if the triplet problem can be solved.

### 7.2.3 Algorithm to construct phylogenetic tree

#### Auxiliary graph:

Given a set of rooted triplets  $S$  that are labeled by  $L$ , for any subset  $L' \subseteq L$ , the auxiliary graph for  $L'$ , denoted by  $G(L')$ , is an undirected graph with vertex set  $L'$  and edge set  $E(L')$ , where for every  $(\{a,b\},c) \in S$  that satisfies  $a,b,c \in L'$ , the edge  $\{a,b\}$  is included in  $E(L')$ .

Figure 7.2 shows  $G(L')$  where  $L' = \{a,b,c,d\}$  and  $S = \{(\{b,c\},a), (\{a,c\},d), (\{d,e\},b)\}$ .

$(\{d,e\},b)$  with  $L = \{a,b,c,d,e\}$

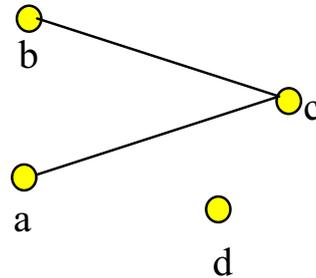


Figure 7.2 Auxiliary Graph

Given a set of rooted triplets  $S$ , the algorithm of Aho et al. calculates the connected components  $C_1, \dots, C_q$  and builds  $G(L)$  recursively.

When  $q = 1$ , it means there is just one connected component  $C_1$ . If  $C_1$  consists of a single leaf  $a$ , the algorithm returns a tree with one leaf labeled by  $a$  (see Figure 7.3). If  $C_1$  contains more than one leaf, the algorithm aborts its execution and returns null since no tree can be consistent with all of the rooted triplets in this case (see figure 7.4).

When  $q > 1$ , the algorithm recursively constructs a tree for each connected component, attaches these trees to a common parent node, and returns the resulting tree. The  $q$  recursive calls to itself are done on the sets  $S_1, \dots, S_q$  obtained by scanning  $S$  (for  $1 \leq p \leq q$ , let  $L_p$  be the set of leaves in  $C_p$ , and for each  $(\{a,b\},c) \in S$ , if all of  $a, b$ , and  $c$  belong to the same  $L_p$ ,  $(\{a, b\}, c)$  will be placed in  $S_p$ ).

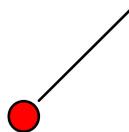


Figure 7.3 A phylogenetic tree with a single leaf

The following is the algorithm to construct a phylogenetic tree, which takes a set of triplets  $S$  as input:

TreeConstruct( $S$ )

1. Let  $L$  be the set of species appear in  $S$ . Build  $G(L)$

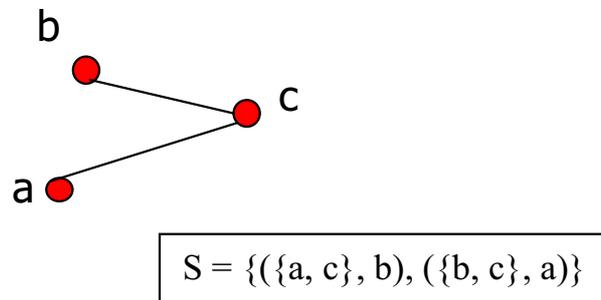


Figure 7.4 No tree can be consistent with single connected component

2. Let  $C_1, C_2, \dots, C_q$  be the set of connected components in  $G(L)$
3. If  $q > 1$ , then
  - For  $i = 1, 2, \dots, q$ , let  $S_i$  be the set of triplets in  $S$  labeled by the set of leaves in  $C_i$ .
  - Let  $T_i = \text{TreeConstruct}(S_i)$
  - Let  $T$  be a tree formed by connecting all  $T_i$  with the same parent node. Return  $T$ .
4. If  $q=1$  and  $C_1$  contains exactly one leaf, return the leaf; else return fail.

Figure 7.5 illustrates how this algorithm works with  $S = \{(\{b, c\}, a), (\{a, c\}, d), (\{d, e\}, b)\}$

### 7.2.4 Time analysis:

Let  $n$  be the number of leaves and  $k$  be the size of  $S$ . There are at most  $n$  recursive calls, and at each recursive level it requires  $O(k)$  time to scanning the constraints to compute the sets  $S_p$  and  $O(k)$  time to build all auxiliary graphs and to find their connected components. So the total complexity of this algorithm is  $O(kn)$ . For the proof of this algorithm, please refer to Aho, Sagiv, Szymanski, and Ullman (1981).

Henzinger, King, and Warnow(1999) showed how to implement the algorithm of Aho et al. to run in  $\text{Min}\{O(kn^{0.5}), O(k + n^2 \log n)\}$ . By the result of Holm, Lichtenberg, and Thorup(2001), the time complexity is further improved to  $\text{Min}\{O(k \log^2 n), O(k + n^2 \log n)\}$  (Jansson, Ng, Sadakane, and Sung, 2004)

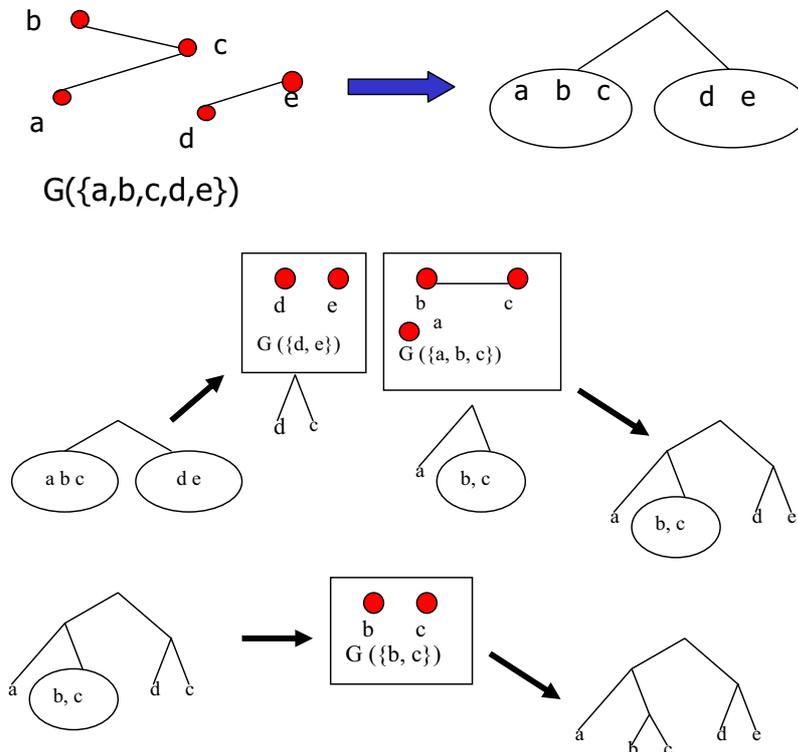


Figure 7.5 Construct a phylogenetic tree from rooted triplets

## 7.3 Phylogenetic network

Phylogenetic tree is a mathematical model for representing evolutionary histories. Here, we assume that the evolution is a point mutation, which means the evolution only contains one type of mutation. However, evolution is in fact more than point mutation; there are some other types of evolutions such as Hybridization and Horizontal gene transfer. Hybridization is the process of mating of individuals from different species or sub-species. For example, a tiger and a lion can give birth to a tiglion. Horizontal gene transfer is any process in which an organism transfers genetic material (i.e. DNA) to another cell that is not its offspring. For example, by horizontal gene transfer, a bovine corona virus and murine hepatitis virus will form the SARS virus.

Since it is not possible for a tree node to have 2 parents, phylogenetic tree cannot be used to model the latter two types of evolution, a new structure is needed.

### 7.3.1 Definition of phylogenetic network

Phylogenetic network is a directed acyclic graph (i.e. no cycle) with the following properties:

- Every node has outdegree at most 2
- Has exactly one node with indegree = 0, which is root node
- Except the root, every node has indegree 1 or 2.
- No node has both indegree 1 and outdegree 1

The nodes with outdegree = 0 are called leaves. They are labeled by distinct elements, and each represents a species.

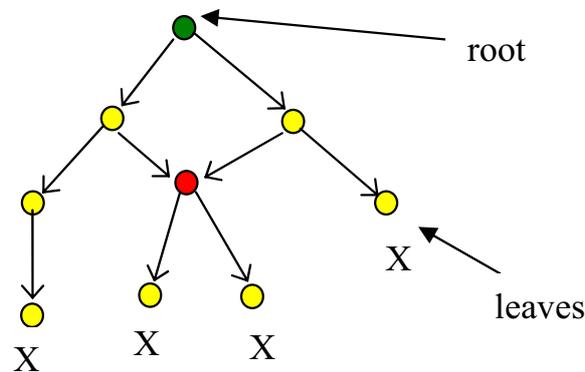


Figure 7.6 A Phylogenetic Network

There are two special kinds of nodes, which are Hybrid node and Split node. Hybrid nodes are those nodes with indegree 2. A node  $s$  is a split node corresponding to a hybrid node  $h$  if there exists two disjoint paths between  $s$  and  $h$ . In Figure 7.6, the red node is a hybrid node and the split node corresponding to the red node is represented by the green node.

### 7.3.2 Level of a network

Let  $U(N)$  be the corresponding undirected graph of a phylogenetic network  $N$ .  $N$  is called a level- $f$  network, if the maximum number of hybrid nodes in every biconnected component in  $U(N)$  is  $f$ . A biconnected component is a subgraph such that for any 2 points of that component, you can find (at least) 2 disjoint paths connecting them.

For the example in Figure 7.7, the maximum number of hybrid nodes (red nodes) in every biconnected component is 2, so it is a level-2 network.

A level-0 Phylogenetic network is actually a tree; a level-1 network is also called a galled network. Since people expect most of recombination hybridization are

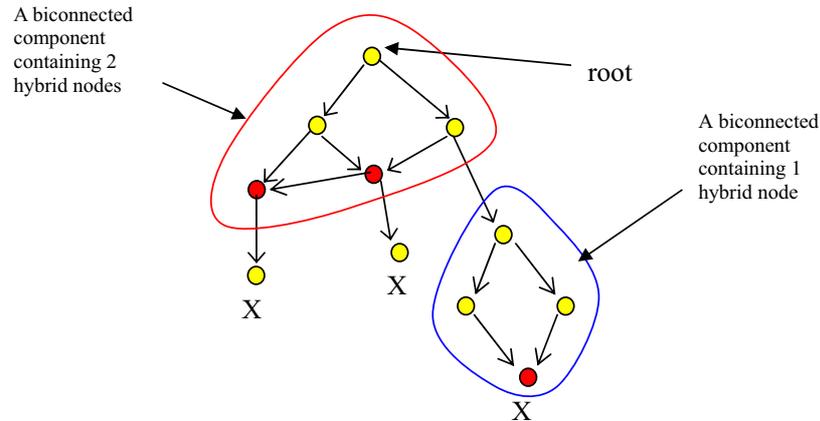


Figure 7.7 Level-2 Phylogenetic Network

in the lower layer, level-1 networks seem biologically meaningful. You may refer to Gusfield, Eddhu, and Langley 2003 for details.

### 7.3.3 Open Problem

How to reconstruct a phylogenetic network?

One of the interesting topics is by using the triplet method for reconstructing a level-1 network (or a galled network)

## 7.4 Triplet method for phylogenetic network

Since computing a phylogenetic network is important, this section discusses how to construct a network using triplet method.

### 7.4.1 Reconstruct a network by sorting network

Given a set of triplets, this section shows that we can always find a network which satisfies all triplets by sorting network.

### 7.4.2 Comparator networks

Def: Let  $J = \{0, \dots, n-1\}$  be an index set and let  $A$  be a set with an order relation  $\leq$ . A data sequence is a mapping:  $J \rightarrow A$ , i.e. a sequence of length  $n$  of data. The set of all data sequences of length  $n$  over  $A$  is denoted by  $A^n$ .

Def: The sorting problem consists of reordering an arbitrary data sequence  $a_0, \dots, a_{n-1}$ , where  $a_i \in A$ , to a data sequence  $a_{\varphi(0)}, \dots, a_{\varphi(n-1)}$  such that

$a_{\varphi(i)} \leq a_{\varphi(j)}$  for  $i < j$   
 Where  $\varphi$  is a permutation of the index set  $J = \{0, \dots, n-1\}$ .

A comparator  $[i:j]$  sorts the  $i$ th and  $j$ th element of a data sequence into nondecreasing order.

Figure 7.8 is the graphic representation of a comparator:

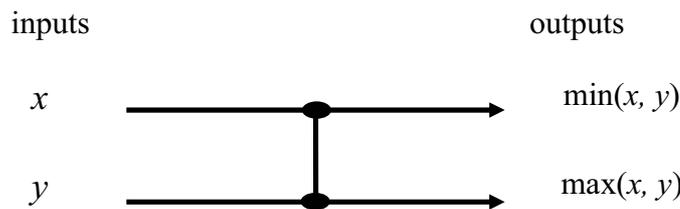


Figure 7.8 A comparator

Def: A comparator stage  $S$  is a composition of comparators  
 $S = [i_1:j_1] \bullet \dots \bullet [i_k:j_k], k \in N$

Such that all  $i_r$  and  $j_s$  are distinct

A comparator network is composition of comparator stages. Figure 7.9 is an example of a comparator network:

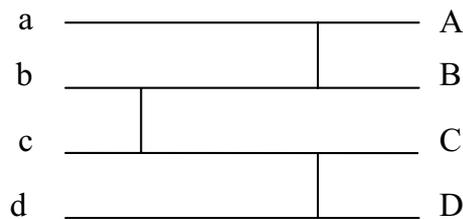


Figure 7.9: Comparator Network

To sort  $n$  numbers, there existing sorting network using  $O(n \log n)$  comparators, and we can find such sorting networking in  $O(n \log^2 n)$  time.

### 7.4.3 Sorting networks

A sorting network is a comparator network that sorts all input sequences.

The problem whether an arbitrary comparator network is a sorting network or not is not easy to solve in general. It is an NP-complete problem. Besides that, sorting networks can of course be constructed systematically and proved to be correct.

Sorting networks are special cases of general sorting algorithms, since all comparisons are data-independent.

Examples of the some sorting networks:

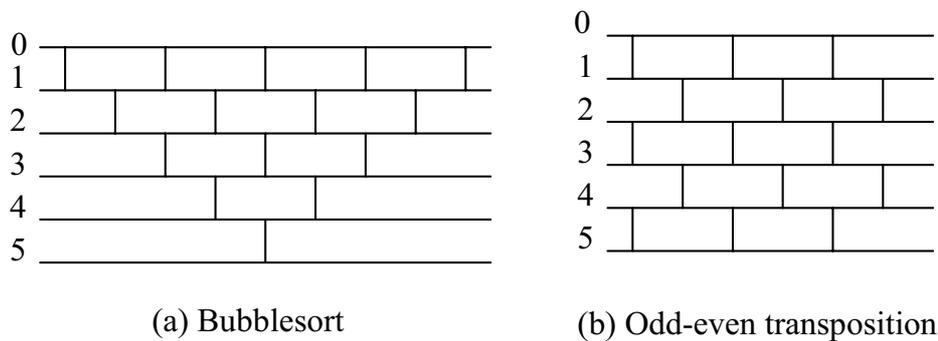


Figure 7.10 Sorting networks

To sort  $n$  numbers, the lower bound for the sorting problem is  $O(n \log(n))$ . This bound is tight even for sorting networks. We can find such sorting networking in  $O(n \log^2 n)$  time.

#### 7.4.4 Relating sorting network with phylogenetic network

Given a rooted triplets  $S$ , a phylogenetic network  $N$  can always be constructed such that every  $S_i \in S$  is consistent with  $N$ . This construction can be carried out in polynomial time in the size of  $S$  as follows.

Let  $P$  be any sorting network for  $n$  elements with  $p$ , which is a polynomial number, comparator stages. Build a directed acyclic graph  $Q$  from  $P$  with  $(p+1)n$  nodes  $\{Q_{i,j} | 1 \leq i \leq p+1, 1 \leq j \leq n\}$  such that there is a directed edge  $(Q_{i,j}, Q_{i+1,j})$  for every  $1 \leq i \leq p$  and  $i \leq j \leq n$ , and two directed edges  $(Q_{i,j}, Q_{i+1,k})$  and  $(Q_{i,k}, Q_{i+1,j})$  for every comparator  $(j,k)$  at edge  $i$  in  $P$  for  $1 \leq i \leq p$ . Then, for  $1 \leq j \leq n-1$ , add the directed edge  $(Q_{1,j}, Q_{1,j+1})$ . Finally, distinctly label the nodes  $\{Q_{p+1,j} | 1 \leq j \leq n\}$  by  $L$ , and for each node in  $Q$  having indegree 1 and outdegree 1 (if any), construct its outgoing edge to obtain  $N$ . See Figure 7.11. It is easy to show that for any  $\{x,y,z\} \subseteq L$ , all three of  $(\{x,y\},z)$ ,  $(\{x,z\},y)$ , and  $(\{y,z\},x)$  are consistent with  $N$ .

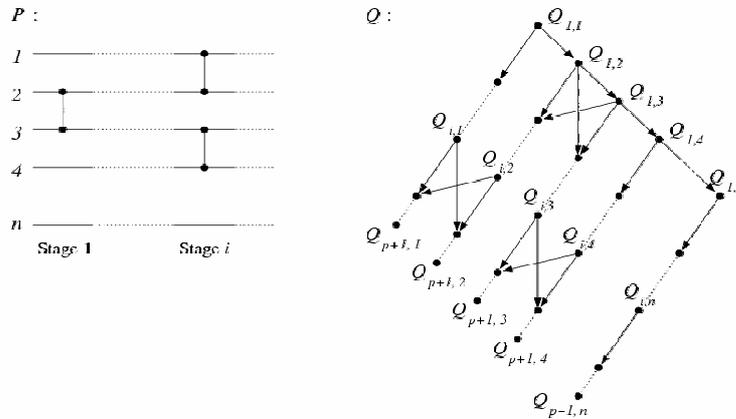


Figure 7.11 Construct a phylogenetic network by using sorting network

Here is an example: for any three species  $\{x,y,z\}$ , we can find a phylogenetic network  $Q$ , in which  $(\{x,y\},z), (\{x,z\},y), (\{y,z\},x)$  are consistent with. See Figure 7.12, for any  $\{x,y,z\}$ , we can find the corresponding sorting subnetwork, and  $(\{y,z\},x), (\{x,z\},y), (\{x,y\},z)$  are consistent with the subnetwork (red point is the root and red lines are edges).

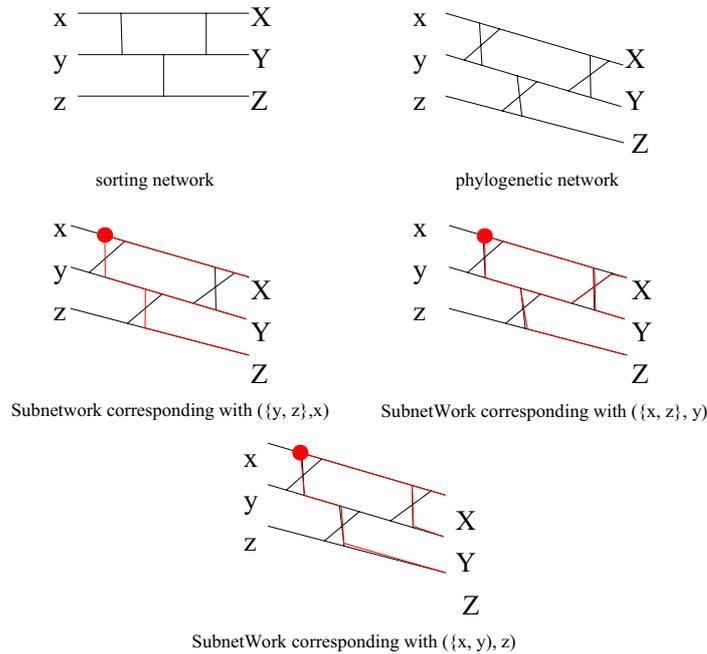


Figure 7.12 An example of constructing a phylogenetic network by using sorting network for three species

Hence, by sorting network, we can always build a general networking using  $O(n \log^2 n)$  time.

### 7.4.5 Building a special type of network

For the more interesting (and biologically more motivated) case where the solution is required to be a level-1 phylogenetic network

A dense set of triplets is the set containing at least one rooted triplet for every cardinality 3 subset of the leaf set.

Given a dense set  $S$  of triplets, output a network with one hybrid node where there is exactly one leaf attached to the hybrid node, which is a galled network, if exists.

Let  $L$  be the set of leaves for  $S$ . For  $L' \subseteq L$ , let  $S|L'$  be the subset of  $S$  consisting of all triplets  $(\{x, y\}, z)$  with  $\{x, y, z\} \subseteq L'$ . We can construct the corresponding phylogenetic network using the following algorithm (BuildTree refers to the fast implementation of the algorithm of Aho et al which can be implemented to run in  $\min\{O(k \log^2 n), O(k + n^2 \log n)\}$  time).

**Algorithm OneHybridLeaf**

**For** each leaf  $c \in L$  **do**

1. Let  $a$  be a leaf in  $L \setminus \{c\}$ .

2. Initialize  $A = \{a\}$  and  $B = \emptyset$ .

**for** every leaf  $x \in L \setminus \{c, a\}$  **do**

    If the edge  $\{x, a\}$  belongs to  $G(L)$  then

$A = A \cup \{x\}$

    else

$B = B \cup \{x\}$

**endfor**

3. Let  $R_1 = \text{BuildTree}(S|A \cup \{c\})$  and  $R_2 = \text{BuildTree}(S|B \cup \{c\})$ .

4. If both  $R_1$  and  $R_2$  are trees then form a phylogenetic network  $N$  by introducing a root node connecting  $R_1$  and  $R_2$  and merge the two leaves labeled by  $c$ .

Return  $N$ .

Figure 7.13 gives the idea of the algorithm, and Figure 7.14 is the graphic presentation.

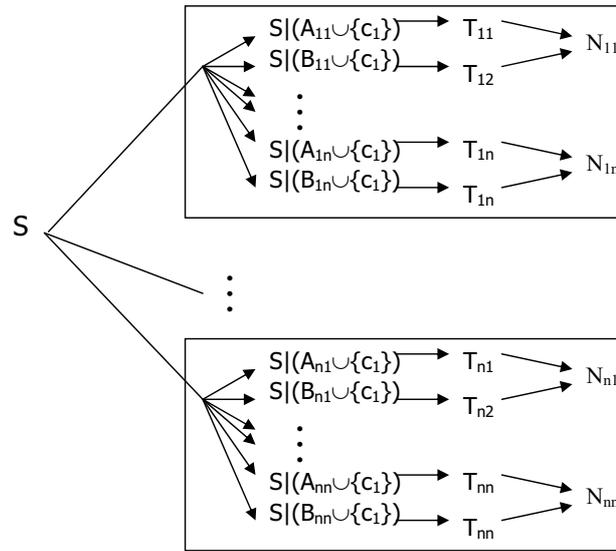


Figure 7.13 Idea of OneHybridLeaf

**Time analysis**

The iteration will iterate  $O(n)$  time; in each iteration, except for Step 3, which takes  $O(k+n^2 \log n)$  time, all the other steps take  $O(n)$  time. In total, the algorithm will take  $O(nk+n^3 \log n)$  time. Since  $S$  is dense,  $k=O(n^3)$ . Thus, the running time of the algorithm OneHybridLeaf is  $O(n^4)$ .

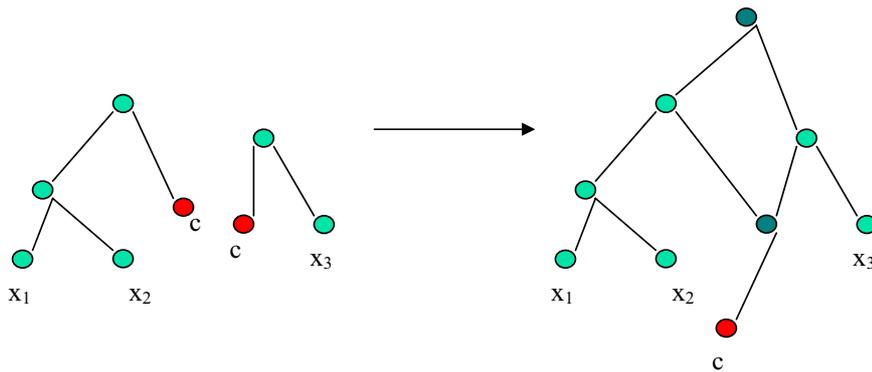


Figure 7.14 Combine the two trees to become a phylogenetic network

## References:

- [ASSU81] A. V. AHO, Y. SAGIV, T. G. SZYMANSKI, and J. D. ULLMAN. Inferring a tree from lowest common ancestors with an application to the optimization of relational expressions. *SIAM Journal on Computing*, 10(3):405-421, 1981.
- [AKS81] M. AJTAI, J. KOMLOS, S. SZEMEREDI: An  $O(N \log N)$  Sorting Network. *Proceedings of the 25th ACM Symposium on Theory of Computing*, 1-9 (1981)
- [GEL03] D. GUSFIELD, S. EDDHU, and C. LANGLEY. Efficient reconstruction of phylogenetic networks with constrained recombination. In *CSB 2003*.
- [HKW99] M. R. HENZINGER, V. KING, and T. WARNOW. Constructing a tree from homeomorphic subtrees, with applications to computational evolutionary biology. *Algorithmica*, 24(1):1-13, 1999.
- [HLT01] J. HOLM, K. DE LICHTENBERG, and M. THORUP. Poly-logarithmic deterministic fully dynamic algorithms for connectivity, minimum spanning tree, 2-edge, and biconnectivity. *Journal of the ACM*, 48(4):723-760, 2001.
- [K73] D.E. KNUTH: *The Art of Computer Programming, Vol. 3: Sorting and Searching*. Addison-Wesley (1973)
- [JNS04] J. JANSSON, H. K. NG, K. SADAKANE, and W. K. SUNG. Rooted maximum agreement supertrees. In *LATIN 2004*.
- [JNS05] J. JANSSON, N. B. NGUYEN, and W. K. SUNG. Algorithms for Combining Rooted Triplets into a Galled Phylogenetic Network. In *SODA 2005*.
- [JS04] J. JANSSON and W. K. SUNG. Inferring a Level-1 Phylogenetic Network from a Dense Set of Rooted Triplets. In *COCOON 2004*.