GMR Genetics and Molecular Research
Online Journal
ISSN:1676-5680

A new algorithm to construct phylogenetic networks from trees

J. Wang

College of Computer Science, Inner Mongolia University, Hohhot, Inner Mongolia, China

Corresponding author: J. Wang
E-mail: wangjuanangle@hit.edu.cn

Genet. Mol. Res. 13 (1): 1456-1464 (2014)
Received January 21, 2013
Accepted September 27, 2013
Published March 6, 2014
DOI http://dx.doi.org/10.4238/2014.March.6.4

ABSTRACT. Developing appropriate methods for constructing phylogenetic networks from tree sets is an important problem, and much research is currently being undertaken in this area. BIMLR is an algorithm that constructs phylogenetic networks from tree sets. The algorithm can construct a much simpler network than other available methods. Here, we introduce an improved version of the BIMLR algorithm, QuickCass. QuickCass changes the selection strategy of the labels of leaves below the reticulate nodes, i.e., the nodes with an indegree of at least 2 in BIMLR. We show that QuickCass can construct simpler phylogenetic networks than BIMLR. Furthermore, we show that QuickCass is a polynomial-time algorithm when the output network that is constructed by QuickCass is binary.

Key words: Phylogenetic network; Evolution; BIMLR; QuickCass; Cass

INTRODUCTION

Developing appropriate methods to infer phylogenetic networks from biological data has been viewed as an important problem (Doolittle, 1999; Linder et al., 2004; Huson, 2005; Wang et al., 2012). There are many recent reports in this area (Huson and Scornavacca, 2011; Huson et al., 2011), which mainly generate 2 kinds of networks: unrooted phylogenetic networks and rooted phylogenetic networks. The unrooted phylogenetic networks include 2 important classes: split networks and quasi-median networks (Buneman, 1971; Bandelt and Dress, 1992; Bandelt et al., 1999). The SplitsTree4 program (Huson and Bryant, 2006) is a convenient tool for inferring unrooted phylogenetic networks from sequences, distances, trees, or splits. This tool collects a lot of methods, such as neighbor-net (Bryant and Moulton, 2004) and the Z-closure super-network method (Huson et al., 2004). The Dendroscope program (Huson and Scornavacca, 2012) is an available tool for computing rooted phylogenetic networks. This tool contains a lot of methods, such as Cass (van Iersel et al., 2010), the galled network (Huson et al., 2009), and the cluster network (Huson and Rupp, 2008).

The rooted phylogenetic networks can be classified into 4 categories. The first is hybridization networks (Maddison, 1997; Linder and Rieseberg, 2004), where the goal is to compute a rooted phylogenetic network from a set of incongruent trees. The second is recombination networks (e.g., Hein, 1993; Huson and Klöepper, 2005; Song and Hein, 2005) and has recently received more attention under the constraint of the galled tree property (e.g., Gusfield et al., 2003; Gusfield, 2005; Gusfield and Bansal, 2005). Here, the task is to construct a rooted phylogenetic network from a set of binary sequences. The third is horizontal gene transfer networks, where the goal is to explain the discrepancies between gene trees and a species tree (e.g., Gusfield et al., 2007; Semple, 2007; Gambette, 2009; Nakleh, 2009; van Iersel et al., 2010). The fourth is the construction of rooted phylogenetic networks from triplets (van Iersel et al., 2008; van Iersel and Kelk, 2011).

Let X be a set of taxa. A rooted phylogenetic tree T on X represents a cluster C, which is a proper subset of X, if there is an edge e in T such that the set of taxa below e equals C. Each rooted phylogenetic tree is uniquely defined by the set of clusters that are represented by it. What are clusters represented by a rooted phylogenetic network N on X? There are 2 different answers to this question. A rooted network N represents a cluster C in the hardwired sense if there is a tree edge $e = (u,v)$, i.e., the edge whose head v has an indegree of at most 1, such that the set of taxa below e equals C. Alternatively, we say that N represents C in the softwired sense if there is some tree edge e for each reticulate node, i.e., the node with an indegree of at least 2, switching 1 incoming edge on and all others off, such that the set of leaves that is reachable below e is labeled precisely by C. In this paper, we will always use "represent" in the softwired sense.

The rooted phylogenetic network can, in theory, be used to explicitly describe evolution in the presence of reticulate events, such as hybridization, horizontal gene transfer, and recombination. However, in biology, these reticulate events are considered "expensive". In the context of phylogenetic analysis, the clusters represent putative monophyletic groups of related species. Therefore, we invoke the parsimony principle to argue that the best network representing a set of clusters in the softwired sense is one that minimizes the number of clusters that are represented by the network on the premise of as few reticulate nodes as possible. The Cass algorithm, which was developed by van Iersel et al. (2010), can construct a phyloge-

netic network in the softwired sense. The phylogenetic network that is constructed by the Cass algorithm has fewer reticulate nodes than other methods. The BIMLR algorithm, which was developed by Wang et al. (2013), is an improved version of the Cass algorithm. The phylogenetic network constructed by BIMLR has fewer reticulate nodes than almost all other methods and represents fewer clusters than all other methods. BIMLR saves a lot of running time over the Cass algorithm.

PRELIMINARY INFORMATION

A rooted phylogenetic network N on X is a directed acyclic graph with a single node of indegree 0, called the root, and leaves that are bijectively labeled by X. The indegree of a node v is denoted by $\delta^-(v)$. Any node of $\delta^-(v) \geq 2$ is called a reticulate node or reticulation, and all others are called tree nodes. Any edge leading to a reticulate node is called a reticulate edge, and all others are called tree edges. The reticulation number of a phylogenetic network $N = (V,E)$ is defined as the following:

$$\sum_{v \in V:\delta^-(v)>0} (\delta^-(v)-1) = |E| - |V| + 1.$$

A graph is called connected if every pair of nodes is connected by some (undirected) path. A cut node or cut edge is a node or edge (except the leaves and the edges leading to leaves), respectively, whose removal disconnects the graph. A graph is biconnected if it contains no cut nodes. A biconnected component of a graph is a maximal biconnected subgraph.

A phylogenetic network is called a level-k network if each biconnected component has a reticulation number of at most k. A level-k network is called a simple level-$\leq k$ network if it has no cut nodes. A phylogenetic network is binary if each reticulate node has indegree 2 and outdegree 1 and each internal tree node has outdegree 2.

Let X be a set of taxa. A cluster C on X is any proper subset of X, excluding both the empty set $\varnothing$ and the full set X. Two different clusters C_1 and C_2 on X are called compatible if they are disjoint or one contains the other, that is, if $C_1 \cap C_2 = \varnothing$, $C_1 \subset C_2$ or $C_2 \subset C_1$; otherwise, it is called incompatible. A set of clusters C on X is called compatible if and only if C is pairwise compatible; otherwise, it is called incompatible. The incompatibility graph $IG(C) = (V,E)$ for C is the undirected graph with node set V = C and edge set E such that any 2 clusters $C_1, C_2 \in C$ are connected by an edge if and only if they are incompatible. The incompatibility degree of a cluster set C, denoted by $d(C)$, is the number of edges in the incompatibility graph $IG(C)$. The incompatibility degree of a taxon $x \in X$ with respect to C denoted by $d(x)$, is $d(x) = d(C) - d(C|_{X\{x\}})$, where $C|_{X\{x\}}$ denotes the resultant set of clusters removing x from each cluster in C. The frequency of a taxon $x \in X$ with respect to C, denoted by $f(x)$, is $f(x) = |\{C: x \in C, C \in C\}|$.

Let C be a set of clusters on X. Given a subset S of X, the restriction of C to S, denoted by $C|_S$, is the result of removing all elements of $X \backslash S$ from each cluster in C. S with $|S| > 1$ is called an ST-set (strict tree set) with respect to C, if S and any 1 cluster $C \in C$ is compatible and any 2 clusters $C_1, C_2 \in C|_S$ are also compatible. An ST-set S is maximal if there is no ST-set W containing S except itself. There is a subtree with respect to a maximal ST-set S, which is constructed for the cluster set $\{C|C \in C, C \subset S\} \cap S$.

BIMLR

This section briefly describes the BIMLR algorithm. The readers can gain more detail in the report (Wang et al., 2013). BIMLR first decomposes the incompatibility graph *IG*(C) into biconnected components and then constructs a simple level-≤*k* network using the BIMLR(*k*) algorithm for each biconnected component separately. Then, BIMLR combines those simple level-≤*k* networks into the resulting phylogenetic networks. Informally, the BIMLR(*k*) algorithm for constructing a simple level-≤*k* network operates as follows.

BIMLR(*k*) removes an incompatible taxon with maximal frequency from each cluster of a given cluster set. It subsequently collapses all maximal ST-sets of the resulting cluster set. The algorithm repeats this step *k* times. After that, the resulting cluster set is compatible, and the second phase of the algorithm begins. BIMLR(*k*) creates a network, i.e., the unique phylogenetic tree, for the resulting cluster set. Then, the algorithm "decollapses", i.e., it replaces each leaf that is labeled by a maximal ST-set with its corresponding subtree. Subsequently, BIMLR(*k*) adds a new leaf below a new reticulate node and labels it by the latest removed taxon.

Following this, BIMLR(*k*) tries adding the reticulate node below each pair of edges. The algorithm continues with a new decollapse step followed by hanging the next leaf below a reticulate node. This network represents the cluster set of this step and then saves it. BIMLR(*k*) finds all networks representing the cluster set of this step and sorts them in descending order of the number of clusters represented, which aims to reduce the number of redundant clusters of the resulting phylogenetic network and weaken the influence of input data order for the Cass algorithm. These steps are also repeated until all removed taxa are appended to the phylogenetic networks, which represent all input clusters. BIMLR outputs the resulting network, which has a minimal number of redundant clusters.

METHODS

Definitions

Let C be a set of clusters on X. For any 1 taxon $x \in X$, we remove it from each cluster in C and obtain the cluster set C_x on $X_x = X \setminus \{x\}$ after collapsing all maximal ST-sets with respect to $C|_{X\setminus\{x\}}$. Then, x is called the real incompatible taxon with respect to C if $d(C_x) = 0$. Otherwise, there exists an incompatible taxon z with respect to C_x. Then, the incompatibility degree of $C_x|_{X\setminus\{z\}}$ is called a potential incompatibility degree of x denoted by $pd(x)$. Here, we specifically point out that if x is the real incompatible taxon with respect to C, then we define its potential incompatibility degree as -1, i.e., $pd(x) = -1$. A taxon x is called a potential incompatible taxon with respect to C if $pd(x) = min\{pd(y)| y \in X\}$. Obviously, a real incompatible taxon with respect to C is also a potential incompatible taxon with respect to C.

QuickCass

When constructing phylogenetic networks, BIMLR, in each step, finds all networks representing the cluster set of this step and sorts them in descending order of the number of clusters represented. Then, the networks representing the minimum number of clusters can be used for the next step. Those operations are mainly used to reduce the number of redundant

clusters of the resulting phylogenetic network. However, the number of reticulations of the resulting phylogenetic network that is constructed by BIMLR is mainly decided by the number of removed incompatible taxa. Importantly, when the resulting phylogenetic network that is constructed by BIMLR is binary, the two networks are equals. However, the number of removed incompatible taxa is the number of steps from the set of input clusters C to the final compatible set of clusters by removing the incompatible taxa with maximal frequency. In each step of constructing phylogenetic networks, BIMLR removes the taxon with a maximal incompatibility degree based on the greedy strategy in order to allow the resulting set of clusters to meet the compatibility requirements as soon as possible. There may be a pathway from the set of input clusters to the compatible set of clusters that has no incompatible taxa. Based on this heuristic, we improved the BIMLR algorithm by using the potential incompatibility degree.

QuickCass first decomposes the incompatibility graph IG(C) into biconnected components like BIMLR, and then it constructs a simple level-$\leq k$ network using the BIMLR(k) algorithm for each biconnected component separately. Then, QuickCass combines those simple level-$\leq k$ networks into the resulting phylogenetic networks like BIMLR. The method of constructing a network for a biconnected component QuickCass(k) for a simple level-$\leq k$ network is as follows. QuickCass(k) is almost the same as BIMLR(k). The only difference between them is that QuickCass(k) removes the potential incompatible taxa with maximal frequency rather than the incompatible taxa with maximal frequency from clusters. The following is the pseudo-code of QuickCass(k).

QuickCass(k) algorithm: constructing a simple level-$\leq k$ network

Input: a set of clusters C

Output: a simple level-$\leq k$ network

1. networks: N: = $\varnothing$

2. cluster sets: C: = $\varnothing$

3. removed taxa: X: = $\varnothing$

4. $k = 0$;

5. if C is compatible, then

compute the rooted phylogenetic tree T for C, and let N = $\{T\}$

return N

6. while C is incompatible, do

$k = k + 1$

add C to C

add x to X, where x is the potential incompatible taxon with maximal frequency with respect to C

$\mathcal{C} := \mathcal{C}|_{X \setminus \{x\}}$

$X := X \setminus \{x\}$

collapse: C: = COLLAPSE(C)

7. construct the unique tree T representing C and push T to N

8. while the reticulation number of the top network N of N is not k, do

9. steps: $s = k - 1 - r$ (r is the reticulation number of N)

10. networks: $N_0 = \varnothing$

11. removed taxon: $x = X[s]$

12. set: $\mathcal{C}' = \mathcal{C}[s]$

13. decollapse: replace each leaf of N that is labeled by a maximal ST-tree S with respect to C

14. for each pair of (not necessarily distinct) edges e_1 and e_2, do
15. create 2 nodes r and t, add an edge from r to t, and label t by x
16. insert a new node u_1 into e_1 and connect u_1 to r
17. insert a new node u_2 into e_2 and connect u_2 to r
18. if the resulting network N represents C, then add N to N_0
19. if $N_0 \neq \varnothing$, then
20. sort all networks in N_0 in descending order of the number of clusters that are represented and then push them to N
21. else

$k = k + 1$
add C to C
add δ to X // δ as a dummy taxon not in X

22. continue
23. return N representing the minimal number of clusters among all phylogenetic networks in N

- **LEMMA 1**. Let N be a rotated binary phylogenetic network with n leaves and k reticulations. Then, N has $2n + 3k - 2$ edges.
- It is easy to prove by inducing the number n of leaves and the number k of reticulations.
- **LEMMA 2**. (Performance of QuickCass(k) algorithm). If the resulting phylogenetic network that is constructed by QuickCass(k) for a cluster set C is a binary, then the time complexity of QuickCass(k) is $O(k|\mathbf{X}|^3|C|^2)$.
- **PROOF**. Paragraph 6 in pseudo-code of QuickCass(k) needs time $O(k|X|^2|C|^2)$. Lines 14-18 will be executed in time $O(|\mathbf{X}|^3|C|)$. If the resulting phylogenetic network for C is a binary network, then lines 19-20 will be executed. In this case the worst case is that there are $O(|X|^2)$ phylogenetic networks that will be sorted in descending order of the number of clusters that are represented, for which the time complexity is $O(|X|^3|C|)$. The best case is that there is only 1 phylogenetic network representing C, for which the time complexity is $O(|X||C|)$. Thus, the lemma follows.

RESULTS

All experiments were performed on machines with an Intel Xeon E5504 2.0 GHz CPU, 8 GB RAM, and 147 GB HDD. The operating system was Debian 4.1 32-bit with Java 1.6 installed. QuickCass was written in Java.

QuickCass has been tested on both practical and artificial data (https://sites.google.com/site/cassalgorithm/data-sets) (van Iersel et al., 2010), and the results are summarized in Table 1 and Figure 1. The table shows the results of an application of QuickCass and BIMLR on several artificial datasets. Here, for each algorithm, the level k, reticulation number r, and the number c of redundant clusters of the output network, as well as the running time t in minutes and seconds, are given, and the last row gives the average values. Table 1 shows that QuickCass consumes less time than BIMLR, and the networks that are constructed by QuickCass have the same reticulation number as those constructed by BIMLR for those artificial datasets. Table 1 also shows that the networks that are constructed by QuickCass represent fewer redundant clusters than those that are constructed by BIMLR for some datasets.

Table 1. Results of QuickCass compared with FastCass for several artificial datasets with |C| clusters and |X| taxa.

Data		QuickCass				FastCass			
|C|	|X|	*t*	*k*	*r*	*c*	*t*	*k*	*r*	*c*
30	5	1s	4	4	0	3s	4	4	1
62	6	4s	5	5	0	11s	5	5	1
42	10	1s	4	4	8	2s	4	4	8
38	11	1s	5	5	8	2s	5	5	8
61	11	1s	5	5	11	3s	5	5	11
77	22	1s	3	3	5	1s	3	3	5
75	30	1s	2	2	19	1s	2	2	19
89	31	1s	4	4	52	1s	4	4	52
180	51	1s	2	2	0	3s	2	2	0
270	76	6s	2	2	0	11s	2	2	0
404	122	25s	2	2	0	51s	2	2	0
120.7	34.1	3.9s	3.4	3.4	9.4	8.1s	3.4	3.4	9.5

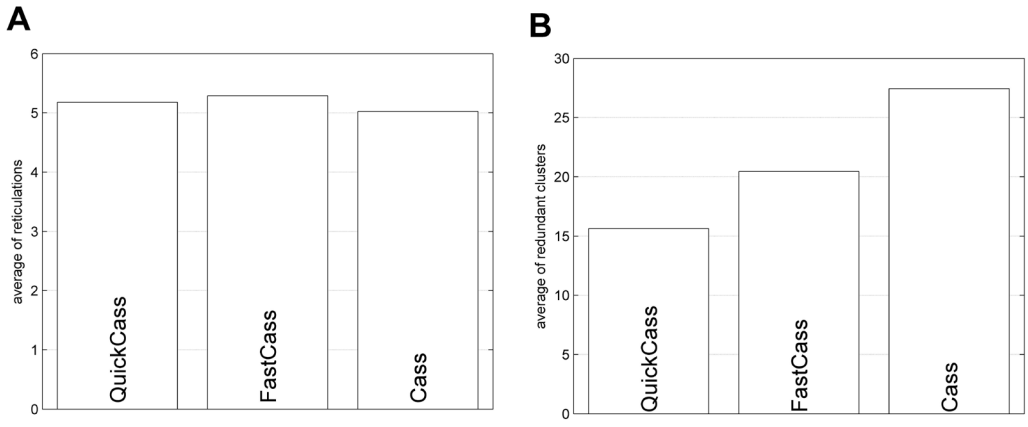

Figure 1. A. The average number of reticulations used by the compared programs. **B.** The average number of redundant clusters represented by the phylogenetic networks constructed by the compared programs.

Wang et al. (2013) compared BIMLR with Cass, the galled network (Huson et al., 2009), and the cluster network algorithms (Huson and Rupp, 2008). van Iersel et al. (2010) compared HYBRIDINTERLEVE (Collins et al., 2011), which is restricted to 2 input trees, and PIRN (Wu, 2010) with Cass. In this paper, we compare QuickCass with BIMLR and Cass. Figure 1 visually shows the results of an application of QuickCass, BIMLR, and Cass with practical data by a histogram. The phylogenetic networks that are constructed for practical datasets are slightly more complicated than those that are constructed for artificial datasets. We averaged over those subsets where all of the programs terminated within 44 h. Here, some results are eliminated, such as the data with |C| = 127 and |X| = 47 because Cass did not finish it within 44 h, while BIMLR and QuickCass finished it within 1 h. As illustrated in Figure 1, it follows that the required reticulations of QuickCass and BIMLR are slightly greater than those of Cass on average, and QuickCass requires fewer reticulations than BIMLR on average. On the other hand, the phylogenetic networks that are constructed by QuickCass represent fewer redundant clusters on average than the phylogenetic networks that are constructed by the other programs, which is also important for constructing parsimony phylogenetic networks. In terms of running time, QuickCass is superior to BIMLR and Cass.

CONCLUSIONS

We have introduced the QuickCass algorithm, which can be used to compute a phylogenetic network for any cluster set. By experiments, it follows that the algorithm performs well on practical data and artificial data. Additionally, experiments show that QuickCass can construct simpler networks than BIMLR. Thus, it provides a useful addition to existing software. Furthermore, we have shown that QuickCass is a polynomial-time algorithm when the resulting phylogenetic network that is constructed by QuickCass is a binary network.

ACKNOWLEDGMENTS

Research supported by the Natural Science Foundation of China (Grant #60932008 and #61172098), the Specialized Research Fund for the Doctoral Program of Higher Education of China (Grant #20112302110040), and the Fundamental Research Funds for the Central Universities (Grant #HIT.ICRST.2010 022).

REFERENCES

Bandelt HJ and Dress AWM (1992). A canonical decomposition theory for metrics on a finite set. *Adv. Math.* 92: 47-105.

Bandelt HJ, Forster P and Rohl A (1999). Median-joining networks for inferring intraspecific phylogenies. *Mol. Biol. Evol.* 16: 37-48.

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

Buneman P (1971). The Recovery of Trees from Measures of Dissimilarity. In: Mathematics in the Archaeological and Historical Sciences (Kendall DG and Tautu P, eds.). Edinburgh University Press, Edinburgh, 387-395.

Collins J, Linz S and Semple C (2011). Quantifying hybridization in realistic time. *J. Comput. Biol.* 18: 1305-1318.

Doolittle WF (1999). Phylogenetic classification and the universal tree. *Science* 284: 2124-2129.

Gambette P (2009). Who's who in phylogenetic networks. Available at [http://www.lirmm.fr/?gambette/PhylogeneticNetworks/]. Accessed April 24, 2010.

Gusfield D (2005). Optimal, efficient reconstruction of root-unknown phylogenetic networks with constrained and structured recombination. *J. Comput. Syst. Sci.* 70: 381-398.

Gusfield D and Bansal V (2005). A Fundamental Decomposition Theory for Phylogenetic Networks and Incompatible Characters. In: Proceedings of the Ninth International Conference on Research in Computational Molecular Biology (RECOMB), Volume 3500 of LNCS. Springer, Berlin, 217-232.

Gusfield D, Eddhu S and Langley C (2003). Efficient Reconstruction of Phylogenetic Networks with Constrained Recombinations. In: Proceedings of the IEEE Computer Society Conference on Bioinformatics (CSB). IEEE Computer Society, Los Alamitos 363.

Gusfield D, Hickerson D and Eddhu S (2007). An efficiently computed lower bound on the number of recombinations in phylogenetic networks: theory and empirical study. *Discrete Appl. Math.* 155: 806-830.

Hein J (1993). A heuristic method to reconstruct the history of sequences subject to recombination. *J. Mol. Evol.* 36: 396-405.

Huson DH (2005). Introduction to Phylogenetic Networks. Tutorial Presented at ISMB, Turbingen.

Huson DH and Klöepper TH (2005). Computing recombination networks from binary sequences. *Bioinformatics* 21 (Suppl 2): ii159-ii165.

Huson DH and Bryant D (2006). Application of phylogenetic networks in evolutionary studies. *Mol. Biol. Evol.* 23: 254-267.

Huson DH and Rupp R (2008). Summarizing Multiple Gene Trees Using Cluster Networks. In: Algorithms in Bioinformatics (WABI), Vol. 5251 of Lecture Notes in Bioinformatics. Springer, Berlin, 296-305.

Huson DH and Scornavacca C (2011). A survey of combinatorial methods for phylogenetic networks. *Genome Biol. Evol.* 3: 23-35.

Huson DH and Scornavacca C (2012). Dendroscope 3: an interactive tool for rooted phylogenetic trees and networks. *Syst. Biol.* 61: 1061-1067.

Huson DH, Dezulian T, Klopper T and Steel MA (2004). Phylogenetic super-networks from partial trees. *IEEE/ACM. Trans. Comput. Biol. Bioinform.* 1: 151-158.

Huson DH, Rupp R, Berry V, Gambette P, et al. (2009). Computing galled networks from real data. *Bioinformatics* 25: i85-i93.

Huson DH, Rupp R and Scornavacca C (2011). Phylogenetic Networks: Concepts, Algorithms and Applications. Cambridge University Press, Cambridge.

Linder CR and Rieseberg LH (2004). Reconstructing patterns of reticulate evolution in plants. *Am. J. Bot.* 91: 1700-1708.

Linder CR, Moret BME, Nakhleh L and Warnow T (2004). Network (Reticulate) Evolution: Biology, Models, and Algorithms. In: Proceedings of the PSB04, Hawaii.

Maddison WP (1997). Gene trees in species trees. *Syst. Biol.* 46: 523-536.

Nakleh L (2009). Evolutionary Phylogenetic Networks: Models and Issues. In: The Problem Solving Handbook for Computational Biology and Bionformatics (Heath L and Ramakrishnan N, eds.). Springer, Berlin. Doi. 10.1007/978-0-387-09760-2_7.

Semple C (2007). Hybridization Networks. In: Reconstructing Evolution - New Mathematical and Computational Advances (Gascuel O and Steel M, eds.). Oxford University Press, Oxford, 277-314.

Song YS and Hein J (2005). Constructing minimal ancestral recombination graphs. *J. Comput. Biol.* 12: 147-169.

van Iersel L and Kelk S (2011). Constructing the simplest possible phylogenetic network from triplets. *Algorithmica* 60: 207-235.

van Iersel L, Keijsper J, Kelk S, Stougie L, et al. (2008). Constructing Level-2 Phylogenetic Networks from Triplets. In: Research in Computational Molecular Biology: Proceedings of the 12th International Conference on Research in Computational Molecular Biology (RECOMB), Vol. 4955, LNCS. Springer, Berlin, 450-462.

van Iersel L, Kelk S, Rupp R and Huson D (2010). Phylogenetic networks do not need to be complex: using fewer reticulations to represent conflicting clusters. *Bioinformatics* 26: i124-i131.

Wang J, Guo MZ and Xing LL (2012). FastJoin, an improved neighbor-joining algorithm. *Genet. Mol. Res.* 11: 1909-1922.

Wang J, Guo MZ, Xing LL, Che K, et al (2013). BIMLR: A method for constructing rooted phylogenetic networks from rooted phylogenetic trees. *Gene* 527: 344-351.

Wu Y (2010). Close lower and upper bounds for the minimum reticulate network of multiple phylogenetic trees. *Bioinformatics* 26: i140-i148.