BIOINFORMATICS

Vol. 00 no. 00 2005
Pages 1–4

HybridNET: a Tool for Constructing Hybridization Networks

Zhi-Zhong Chen [1] and Lusheng Wang [2] *

[1] Department of Mathematical Sciences,Tokyo Denki University, Ishizaka, Hatoyama, Hiki, Saitama, 359-0394, Japan.
[2] Department of Computer Science, City University of Hong Kong, 83 Tat Chee Avenue, Kowloon, Hong Kong.

Received on XXXXX; revised on XXXXX; accepted on XXXXX

Associate Editor: XXXXXXX

ABSTRACT

Motivations: When reticulation events occur, the evolutionary history of a set of existing species can be represented by a phylogenetic network instead of an evolutionary tree. When studying the evolutionary history of a set of existing species, one can obtain a phylogenetic tree of the set of species with high confidence by looking at a segment of sequences or a set of genes. When looking at another segment of sequences, a different phylogenetic tree can be obtained with high confidence, too. This indicates that reticulation events may occur. Thus, we have the following problem: Given two rooted phylogenetic trees on a set of species that correctly represent the tree-like evolution of different parts of their genomes, what is the phylogenetic network with the smallest number of reticulation events to explain the evolution of the set of species under consideration?

Results: We develop a program, named HybridNet, for constructing a hybridization network with the minimum number of reticulate vertices from two input trees. We first implement the $O(3^d n)$-time algorithm in Whidden *et al.*, 2010 for computing a maximum (acyclic) agreement forest. Our program can output all the maximum (acyclic) agreement forests. We then augment the program so that it can construct an optimal hybridization network for each given maximum acyclic agreement forest. To our knowledge, this is the first time that optimal hybridization networks can be rapidly constructed.

Availability: The program is available at http://rnc.r.dendai.ac.jp/∼chen/treeComp.html for non-commercial use.

Contact: lwang@cs.cityu.edu.hk

1 INTRODUCTION

When studying the evolutionary history of a set of existing species, one can obtain a phylogenetic tree of the set of species with high confidence by looking at a segment of sequences or a set of genes. When looking at another segment of sequences, a different phylogenetic tree can be obtained with high confidence, too. This indicates that reticulation events may occur. Thus, we have the following problem: Given two rooted phylogenetic trees on a set of species that correctly represent the tree-like evolution of different parts of their genomes, what is the phylogenetic network with the smallest number of reticulation events to explain the evolution of the set of species under consideration?

The problem was proved to be NP-hard (Hein *et al.*, 1996; Bordewich and Semple, 2005, 2007a), it is challenging to develop programs that can give exact solutions when the two given trees are large or have a large reticulate number. Recently, several software packages have been developed for these problems (Collins *et al.*, 2009; Wu, 2009; Wang and Wu, 2010; Whidden *et al.*, 2010). All those programs only output a number or a maximum (acyclic) agreement forest. None of them gives an optimal phylogenetic network.

We develop a program, named HybridNet, for constructing a hybridization network with the minimum number of reticulate vertices from two input trees. We first implement the $O(3^d n)$-time algorithm in Whidden *et al.*, 2010 for computing a maximum (acyclic) agreement forest. Our program can output all the maximum (acyclic) agreement forests. We then augment the program so that it can construct an optimal hybridization network for each given maximum acyclic agreement forest. To our knowledge, this is the first time that optimal hybridization networks can be rapidly constructed.

2 PROBLEM DEFINITIONS

Let X be a set of existing species. A *binary phylogenetic X-tree* is a tree whose leaf set is X, whose root has in-degree 0 and out-degree 2, and whose non-root and non-leaf vertices each has in-degree 1 and out-degree 2. A *phylogenetic network* on X is a directed acyclic graph D in which the set of vertices of out-degree 0 (still called *leaves*) is X, each non-leaf vertex has out-degree 2, and there is one vertex of in-degree 0 (called the *root*). Note that the in-degree of a non-root vertex in D may be larger than 1. A vertex of in-degree larger than 1 in D is called a *reticulate* vertex. Intuitively, a reticulate vertex corresponds to a *reticulation* event.

A phylogenetic tree T on X *fits* a phylogenetic network N if T can be obtained from N by first deleting some edges and then contracting vertices of out-degree 1 (resulted from the edge deletions).

We are now ready to define the problem of *constructing a phylogenetic network from two phylogenetic trees*:

Input: Two phylogenetic trees T and T' with the same leaf set.

*to whom correspondence should be addressed

Output: A phylogenetic network N with the minimum number r of reticulate vertices such that both T and T' fit N.

Here r is referred to as the *reticulate number* of T and T'. Optimal phylogenetic networks are closely related to maximum acyclic agreement forests (MAAFs). It is widely known that the reticulate number of two phylogenetic trees with the same leaf set is equal to the number of trees in their MAAF minus one.

3 CONSTRUCTING A PHYLOGENETIC NETWORK FROM AN MAAF

Let T and T' be the two input trees. Let $\overline{T}$ and $\overline{T'}$ be the augmented versions of T and T' by adding a dummy leaf. Assume that F is an MAAF of $\overline{T}$ and $\overline{T'}$. We present an algorithm to construct a phylogenetic network from $\overline{T}$, $\overline{T'}$, and F. We proceed as follows.

First, for each vertex u of F, we find the lowest vertex v (respectively, v') in $\overline{T}$ (respectively, $\overline{T'}$) such that all leaf descendants of u in F are also leaf descendants of v (respectively, v') in $\overline{T}$ (respectively, $\overline{T'}$). For convenience, we say that u, v, and v' are *mates* of each other. Moreover, if a vertex of $\overline{T}$ or $\overline{T'}$ has a mate in F, then we call it a *preserved* vertex; otherwise, we call it an *unpreserved* vertex.

There is a way to find the mates of the vertices in F in linear time. For details see the full version of the paper at http://rnc.r.dendai.ac.jp/∼chen/treeComp.html .

We can show that both the root of $\overline{T}$ and that of $\overline{T'}$ are preserved vertices. Thus, the reticulate number of two phylogenetic trees is also equal to the number of trees in an MAAF of the two original input trees (instead of their augmented versions) minus one.

We are now ready to construct a phylogenetic network N of $\overline{T}$ and $\overline{T'}$. Initially, we let N be a copy of $\overline{T}$. Obviously, $\overline{T}$ fits N; we will always maintain this property hereafter. We then add more vertices and edges to N so that $\overline{T'}$ also fits N, by performing the following four steps:

Step 1: In this step, we look at each edge (u, v) in F. Let $P'_{u,v}$ denote the path in $\overline{T'}$ from the mate of u to the mate of v. Note that the internal vertices of $P'_{u,v}$ are unpreserved vertices. In order for $\overline{T'}$ to fit N, we *embed* $P'_{u,v}$ into the path of N from the mate x of u to the mate y of v as follows: If $P'_{u,v} = u, w'_1, w'_2, \ldots, w'_k, v$, then we find the parent z of y in N and modify N by splitting the edge (z, y) into a path $Q = z, w_1, w_2, \ldots, w_k, y$. For convenience, for each $i \in \{1, 2, \ldots, k\}$, we call w_i and w'_i the *mates* of each other. Roughly speaking, after this step, for each edge $(u, v) \in F$, the path in N from the mate of u to the mate of v is an expansion of both $P'_{u,v}$ and the path of $\overline{T}$ from the mate of u to the mate of v.

Step 2: Note that there may exist vertices in $\overline{T'}$ that have no mates in N. For convenience, we call these vertices of $\overline{T'}$ *free* vertices. For each free vertex v' of $\overline{T'}$, we add a copy v of v' to N (as an isolated vertex) and again call v and v' the *mates* of each other.

Step 3: For each edge (u', v') of $\overline{T'}$ such that at least one of u' and v' is a free vertex, we add edge (u, v) to N, where u and v are the mates of u' and v' in N, respectively. Note that after this step, the in-degree of each vertex in N remains to be at most 1.

Step 4: For each preserved vertex v' of $\overline{T'}$ such that v' is not the root of $\overline{T'}$ but its mate in F is of in-degree 0, we find the parent u' of v' in $\overline{T'}$ and add the edge (u, v) to N, where u and v are the mates of u' and v' in N, respectively. Note that after this step, there are exactly $d - 1$ vertices of in-degree 2 in N, where d is the number of connected components in F. This completes the construction of N.

Obviously, $\overline{T}$ fits N because initially $\overline{T}$ fits N and Steps 1 through 4 do not invalidate this property. Moreover, $\overline{T'}$ fits N because each edge of $\overline{T'}$ is either embedded in N or copied to N. Now, since N is a phylogenetic network with exactly $d-1$ reticulate vertices, it is optimal by Lemma 2.

If we want a phylogenetic network of T and T' (instead of their augmented versions), it suffices to modify the above N by removing the root and its dummy child. The whole algorithm for constructing the optimal phylogenetic network N of T and T' runs in linear time.

In the above construction of N, when we embed a path P of $\overline{T'}$ into N, we may have multiple choices to do so. That is, it may be possible to construct more than one optimal phylogenetic network from T, T', and F.

4 IMPLEMENTATION

We have implemented the algorithm in Whidden *et al*., 2010 in ANSI C, obtaining a program $HybridNet$ for computing the rSPR distance, a single MAF, all MAFs, the hybridization/reticulate number, a single MAAF together with an optimal hybridization network, and all MAAFs together with an optimal hybridization network for each MAAF, respectively. $HybridNet$ is available at

http://rnc.r.dendai.ac.jp/∼chen/treeComp.html,

where one can download executables that can run on a Windows XP (x86), Windows 7 (x64), or Linux machine.

After downloading $HybridNet$, one can run it as follows:

HybridNet -OPTION TreeFile1 TreeFile2

Here, TreeFile1 and TreeFile2 are two text files each containing a phylogenetic tree in the Newick format. The label of each leaf in an input tree should be a string consisting of letters in $\{0, 1, \ldots, 9, a, b, \ldots, z, A, B, \ldots, Z, \_, .\}$. There is no limit on the length of the label of each leaf.

OPTION is a string in the set {HN, MAAF, MAAFs, rSPRDist, MAF, MAFs} controlling the output as follows:

- HN: The output is the hybridization number between the two input trees.
- MAAF: The output is one MAAF of the two input trees together with one optimal hybridization network for the MAAF.
- MAAFs: The output is all MAAFs of the two input trees together with one optimal hybridization network for each MAAF.
- rSPRDist: The output is the rSPR distance between the two input trees.
- MAF: The output is one MAF of the two input trees.
- MAFs: The output is all MAFs of the two input trees.

$HybridNet$ outputs an MAAF (respectively, MAF) by printing out the leaf sets of the trees in the MAAF (respectively, MAF), while it outputs a hybridization network in its extended Newick format. When OPTION is MAAFs (respectively, MAFs), $HybridNet$ uses a red-black tree to store all MAAFs (respectively, MAFs) that have been found so far. If an MAAF (respectively, MAF) is found in the red-black tree, then $HybridNet$ will not output it again. In this way, $HybridNet$ outputs the MAAFs (respectively, MAFs) without repetition.

We remind the reader that one can view a tree in the Newick format and a network in the extended Newick format by using Dendroscope due to Huson *et al*. (2007).

To compare the efficiency of $HybridNet$ with the previously best exact programs (namely, *SPRDist* by Wu (2009) and *HybridInterleave* by Collins *et al*. (2009)), we have run $HybridNet$, *SPRDist*, and *HybridInterleave* on both simulated data and biological data. We omit the comparison with the other known programs such as $EEEP$, $HorizStory$, $DarkHorse$, $RIATA$-HGT, $LatTrans$ because according to Wu (2009) and Collins *et al*. (2009), they are slower than *SPRDist* or *HybridInterleave*. The experiment was performed on a 3.33 GHz Linux PC. Note that *SPRDist* computes the rSPR distance of two phylogenetic trees while *HybridInterleave* computes the hybridization number of two phylogenetic trees. Recently, Wang and Wu (2010) announced that they have obtained a program for computing the hybridization number of two phylogenetic trees. However, it turns out that their program is slower than *HybridInterleave*.

4.1 Simulated Data

We use the benchmark dataset provided by Beiko and Hamilton (2006). To obtain a pair (T, T') of trees, Beiko and Hamilton (2006) first generate T randomly and then obtain T' from T by performing a specified number $\tilde{d}$ (say, 10) of random rSPR operations on T. So, the actual rSPR distance of T and T' is at most $\tilde{d}$. Moreover, the hybridization number of T and T' can be $\tilde{d}$, smaller than $\tilde{d}$, or larger than $\tilde{d}$. In this way, they obtain a lot of benchmark tree pairs. To compare the efficiency of our program with *SPRDist* and *HybridInterleave*, we only pick the 10 tree pairs with the largest size (100 leaves) and the most random rSPR operations performed (10). See Table 1 for the experimental results.

The experimental results in Table 1 indicate that $HybridNet$ can give the exact solutions within a second. *SPRDist* takes 9 seconds to 14.5 minutes for some easy cases. However, when the number of leaves or the rSPR distance is large, *SPRDist* often crashes. *HybridInterleave* is quite slow for simulated data and it takes more than one day to finish for many cases. Therefore, $HybridNet$ is more efficient and stable than *SPRDist* and *HybridInterleave*.

4.2 Biological Data

We use the Poaceae dataset from the Grass Phylogeny Working Group (Grass PWG, 2001). The dataset contains sequences for six loci: internal transcribed spacer of ribosomal DNA (ITS); NADH dehydrogenase, subunit F (ndhF); phytochrome B (phyB); ribulose 1,5-biphosphate carboxylase/oxygenase, large subunit (rbcL); RNA polymerase II, subunit β'' (rpoC2); and granule bound starch synthase I (waxy). The Poaceae dataset was previously analyzed by Schmidt (2003), who generated the inferred rooted binary trees for these loci. See Table 2 for the experimental results.

As can be seen from Table 2, $HybridNet$ is generally more efficient and stable than *SPRDist* and *HybridInterleave*. In more details, $HybridNet$ is always faster than *SPRDist*; this is particularly obvious for the tree pair (ndhf,ITS). *HybridNet* compares well with *HybridInterleave*; in particular, for the tree pair (rbcL,ITS), it runs much faster. Of special interest is that even when we turn on the option MAAFs or MAFs to find all solutions, $HybridNet$ runs faster than $HybridInterleave$ and $SPRDist$ which find only a single solution. So, one can conclude that $HybridNet$ runs faster not because of luck.

Table 1. Computing the rSPR distance and the hybridization number on simulated data from Beiko and Hamilton (2006). Columns d and h show the rSPR distance and the hybridization number, respectively. Columns $HybridNet$, *SPRDist*, and *HybridInterleave* show the running times of $HybridNet$, *SPRDist*, and *HybridInterleave*, respectively. Time is measured in seconds (s), minutes (m), hours (h), and days (d). When a program crashes, we use symbol '$-$' to show its running time. When a program did not stop after one day, we simply stopped it and use '$> 1d$' to show its running time.

d	$HybridNet$	*SPRDist*	h	$HybridNet$	*HybridInterleave*
10	<1s	−	10	<1s	>1d
10	<1s	−	10	<1s	>1d
9	<1s	9.1m	9	<1s	>1d
9	<1s	13m	9	<1s	>1d
10	< 1s	−	10	1s	>1d
9	<1s	12s	9	<1s	>1d
10	<1s	−	10	<1s	>1d
10	< 1s	−	10	3s	>1d
10	<1s	−	10	<1s	>1d
10	<1s	−	10	<1s	>1d
8	<1s	9s	8	<1s	>1d
7	<1s	−	7	<1s	10.4h
8	<1s	9.6m	8	<1s	>1d
8	<1s	−	8	<1s	>1d
8	<1s	9.8m	8	<1s	>1d
8	<1s	−	8	<1s	>1d
7	<1s	14s	7	<1s	6.7m
8	<1s	8s	8	<1s	>1d
7	<1s	25s	7	<1s	>1d
8	<1s	29s	8	<1s	>1d

ACKNOWLEDGMENTS

Supported by the Grant-in-Aid for Scientific Research of the Ministry of Education, Science, Sports and Culture of Japan, under Grant No. 20500021 and a grant from RGC of the Hong Kong Special Administrative Region, China [Project No. CityU 121207].

REFERENCES

Beiko,R.G. and Hamilton,N. (2006) Phylogenetic identification of lateral genetic transfer events. *BMC Evol. Biol.*, **6**, 159-169.

Bordewich,M. and Semple,C. (2005) On the computational complexity of the rooted subtree prune and regraft distance, *Annals of Combinatorics*, **8**, 409-423.

Bordewich,M. and Semple,C. (2007) Computing the minimum number of hybridization events for a consistent evolutionary history, *Discrete Applied Mathematics*, **155**, 914-928.

Collins,L., Linz,S., and Semple,C. (2009) Quantifying hybridization in realistic time. To appear in *Journal of Computational Biology*.

Table 2. Computing the rSPR distance and the hybridization number on 15 pairs of trees for the Poaceae data. Column *pair* shows the tree pairs. Column $\#taxa$ shows the number of leaves in an input tree, while columns d and h show the rSPR distance and the hybridization number, respectively. Columns $HybridNet$, *SPRDist*, and *HybridInterleave* show the running times of $HybridNet$, *SPRDist*, and *HybridInterleave*, respectively. Time is measured in seconds (s), minutes (m), and hours (h).

pair	*#taxa*	d	$HybridNet$	*SPRDist*
ndhF,phyB	40	12	<1s	2m9s
ndhF,rbcL	36	10	< 1s	29s
ndhF,rpoC2	34	11	< 1s	50s
ndhF,waxy	19	7	<1s	12s
ndhF,ITS	46	19	20s	**14h**
phyB,rbcL	21	4	< 1s	5s
phyB,rpoC2	21	6	< 1s	4s
phyB,waxy	14	3	< 1s	2s
phyB,ITS	30	8	< 1s	22s
rbcL,rpoC2	26	11	< 1s	1.3m
rbcL,waxy	12	6	< 1s	3s
rbcL,ITS	29	13	< 1s	8.5m
rpoC2,waxy	10	1	< 1s	< 1s
rpoC2,ITS	31	14	< 1s	27m
waxy,ITS	15	7	< 1s	8s

pair	*#taxa*	h	$HybridNet$	*HybridInterleave*
ndhF,phyB	40	14	14s	8s
ndhF,rbcL	36	13	2s	2s
ndhF,rpoC2	34	12	< 1s	7s
ndhF,waxy	19	9	< 1s	1s
ndhF,ITS	46	19	3.75m	4m
phyB,rbcL	21	4	< 1s	< 1s
phyB,rpoC2	21	7	< 1s	< 1s
phyB,waxy	14	3	< 1s	< 1s
phyB,ITS	30	8	< 1s	< 1s
rbcL,rpoC2	26	13	< 1s	7s
rbcL,waxy	12	7	< 1s	1s
rbcL,ITS	29	14	2s	**1.5h**
rpoC2,waxy	10	1	< 1s	< 1s
rpoC2,ITS	31	15	4s	14.5m
waxy,ITS	15	8	< 1s	< 1s

Grass Phylogeny Working Group (2001) Phylogeny and subfamilial classification of the grasses (poaceae). *Ann. Mo. Bot. Gard.*, **88**, 373-457.

Hein, J., Jing, T., Wang, L., and Zhang, K. 1996. On the complexity of comparing evolutionary trees. *Discrete Appl. Math.*, **71**, 153-169.

Huson, D.H., Richter, D.C., Rausch, C., Dezulian, T., Franz, M., and Rupp, R. Dendroscope: An interactive viewer for large phylogenetic trees. *BMC Bioinformatics*, **8**, 460-460.

Schmidt,H.A. (2003) Phylogenetic trees from large datasets. Ph.D. thesis, Heinrich-Heine-Universitat, Dusseldorf.

Wang,J. and Wu,Y. (2010) Fast computation of the exact hybridization number of two phylogenetic trees. *Proceedings of ISBRA 2010*, 203-214.

Wu,Y. (2009) A practical method for exact computation of subtree prune and regraft distance. *Bioinformatics*, **25**(2), 190-196.

Whidden C., Beiko R.G., and Zeh N. (2010) Fast FPT Algorithms for Computing Rooted Agreement Forest: Theory and Experiments, *LNCS* bf 6049, 141-153.