Tanglegrams for rooted phylogenetic trees and networks

Celine Scornavacca, Franziska Zickmann, Daniel Huson

▶ **To cite this version:**

Celine Scornavacca, Franziska Zickmann, Daniel Huson. Tanglegrams for rooted phylogenetic trees and networks. Bioinformatics, 2011, 27 (13), pp.i248-i256. 10.1093/bioinformatics/btr210 . hal-02155176

HAL Id: hal-02155176
https://hal.science/hal-02155176

Submitted on 18 Dec 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

BIOINFORMATICS

Vol. 27 ISMB 2011, pages i248–i256
doi:10.1093/bioinformatics/btr210

Tanglegrams for rooted phylogenetic trees and networks

Celine Scornavacca*, Franziska Zickmann and Daniel H. Huson*

Center for Bioinformatics (ZBIT), Tübingen University, Sand 14, 72076 Tübingen, Germany

ABSTRACT

Motivation: In systematic biology, one is often faced with the task of comparing different phylogenetic trees, in particular in multi-gene analysis or cospeciation studies. One approach is to use a tanglegram in which two rooted phylogenetic trees are drawn opposite each other, using auxiliary lines to connect matching taxa. There is an increasing interest in using rooted phylogenetic networks to represent evolutionary history, so as to explicitly represent reticulate events, such as horizontal gene transfer, hybridization or reassortment. Thus, the question arises how to define and compute a tanglegram for such networks.

Results: In this article, we present the first formal definition of a tanglegram for rooted phylogenetic networks and present a heuristic approach for computing one, called the *NN-tanglegram* method. We compare the performance of our method with existing tree tanglegram algorithms and also show a typical application to real biological datasets. For maximum usability, the algorithm does not require that the trees or networks are bifurcating or bicombining, or that they are on identical taxon sets.

Availability: The algorithm is implemented in our program Dendroscope 3, which is freely available from www.dendroscope.org.

Contact: scornava@informatik.uni-tuebingen.de; huson@informatik.uni-tuebingen.de

1 INTRODUCTION

In systematic biology, one is often faced with the task of comparing different phylogenetic trees, in particular in multi-gene analysis or cospeciation studies (Burt and Trivers, 2008; Charleston, 1998; Charleston and Perkins, 2003; Lee and Stock, 2010; Machado *et al.*, 2005; Merkel *et al.*, 2010). One way to visualize similarities and differences is to draw two phylogenetic trees as rooted trees side by side and to draw lines (which we will call *connectors*) between taxa that correspond to each other in the two trees, see Figure 1. Such a depiction is called a *tanglegram* and different variations of the problem of computing an *optimal* tanglegram have been studied in the literature.

For example, a number of articles (Bansal *et al.*, 2009; Böcker *et al.*, 2009; Buchin *et al.*, 2009; Fernau *et al.*, 2005; Nöllenburg *et al.*, 2009; Venkatachalam *et al.*, 2010) consider the One-Tree Crossing Minimization (OTCM) and the Two-Tree Crossing Minimization (TTCM) problems that both aim at minimizing the number of crossings between connectors. In the former problem, the layout of one of the trees is fixed and that of the other is mutable whereas in the latter formulation the layout of both trees are allowed to be changed. For binary trees, OTCM is solvable in $O(n\log n)$ time (Venkatachalam *et al.*, 2010), while TTCM is NP-complete (Fernau *et al.*, 2010). In Dwyer and Schreiber (2004), the authors describe a 'seesaw' heuristic for the TTCM problem for

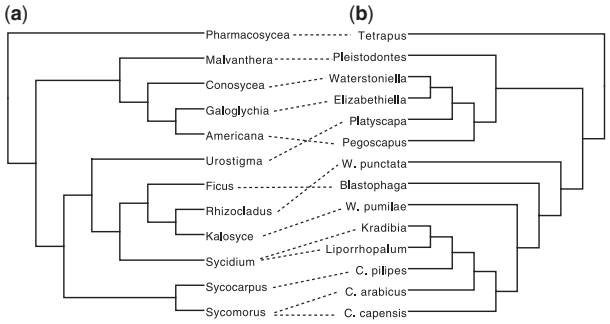

Fig. 1. A tanglegram between a phylogeny of sections of *Ficus* (**a**) and that of their associated genera of pollinating wasps (**b**). Adapted from (Machado *et al.*, 2005).

binary (or bifurcating) trees, which operates by repeatedly solving the OTCM problem, each time switching the roles of the two trees. A branch-and-bound approach for binary trees that works in $O(n^3)$ time and gives a 2-approximation for complete binary trees is presented in Buchin *et al.* (2009). A generalization of the algorithm to unbalanced binary trees is described in Nöllenburg *et al.* (2009), though in this case the approximation factor does not hold. In addition, this article gives an ILP formulation and an exact branch-and-bound algorithm for binary trees, where the latter has a worst-case running time of $O(n^2+n\cdot 2^{2n})$. Other approaches use *fixed-parameter tractability* (FPT) parameterized by the number k of connector crossings (Fernau *et al.*, 2005). A generalization to non-binary trees is discussed in Venkatachalam *et al.* (2010). The only algorithm that is able to compute tanglegrams for binary trees with many-to-many connections is described in Bansal *et al.* (2009). Their algorithm requires $O(k\log^2 k/\log\log k)$ time (where k is the number of connectors) in the case that one tree is fixed. Additionally, they present some alternating and local search strategies for the TTCM problem.

While evolutionary histories are usually described by rooted phylogenetic trees, in some cases rooted phylogenetic networks may provide a more accurate evolutionary scenario, especially when mechanisms such as horizontal gene transfer, hybridization, recombination, reassortment or incomplete lineage sorting have played a role in shaping the history. There is currently much research on the development of computational methods for computing rooted phylogenetic networks, for an overview see Huson and Scornavacca (2011a); Huson *et al.* (2011).

The goal of this article is to introduce the concept of a tanglegram for rooted phylogenetic networks and to provide a useful heuristic for computing such tanglegrams. Unlike trees, rooted phylogenetic networks are not necessarily planar and so the definition of an *optimal* tanglegram is not immediately obvious for them. Our heuristic does not require that the networks are bifurcating or

*To whom correspondence should be addressed.

© The Author(s) 2011. Published by Oxford University Press.
This is an Open Access article distributed under the terms of the Creative Commons Attribution Non-Commercial License (http://creativecommons.org/licenses/by-nc/2.5), which permits unrestricted non-commercial use, distribution, and reproduction in any medium, provided the original work is properly cited.

Downloaded from bioinformatics.oxfordjournals.org at Istituto di Neurologia/ on June 17, 2011

bicombining, or that they both contain the same set of taxa. It can also be used to compare two trees, or to compare a tree and a network, of course. Moreover, the heuristic can also be employed to align the order of leaves for a whole collection of rooted trees or networks. The algorithm is implemented in our Java program Dendroscope 3 (Huson and Scornavacca, 2011b; Huson *et al.*, 2007), which is freely available from www.dendroscope.org.

2 BASICS

Throughout this article, we follow the terminology and notation defined in Huson *et al.* (2011) and assume that the reader is familiar with graphs and related terminology. In this section we introduce some basic concepts and results.

2.1 Phylogenetic trees and networks

Let $\mathcal{X}$ be a set of taxa. An *unrooted phylogenetic network* N on $\mathcal{X}$ is an unrooted, connected graph whose leaves are bijectively labeled by the taxa in $\mathcal{X}$. A *rooted DAG* is defined as a directed graph that is free of directed cycles and contains precisely one node with indegree zero, called the root. A *rooted phylogenetic network* N on $\mathcal{X}$ is a *rooted DAG* whose set of leaves is bijectively labeled by the taxa in $\mathcal{X}$. A node v in N is said to be a *tree node* if its indegree is less than two and a *reticulate node* otherwise. An edge of N is called a *tree edge*, unless it leads to a reticulate node, in which case it is called a *reticulate edge*. A *(rooted) phylogenetic tree* is a (rooted) phylogenetic network for which it is not possible to delete any edge without producing a graph that is not connected.

2.2 Clusters, splits and split networks

Let N be a rooted phylogenetic network on $\mathcal{X}$. Any tree edge e in N defines a *cluster* $C(e)$ which is given by the set of all taxa that label leaves that lie below e in N. We use $\mathcal{C}(N)$ to denote the set of all clusters obtainable from N in this way.

Let $\mathcal{X}$ be a set of taxa. A *split* $A\,|\,B$ on $\mathcal{X}$ is a partition of $\mathcal{X}$ into two non-empty sets. Any edge e in an unrooted phylogenetic tree T on $\mathcal{X}$ defines a split $S(e)=A\,|\,B$, where A and B are all taxa that label nodes that lie on one side or the other side of the edge, respectively. A split $A\,|\,B$ is said to *separate* two taxa x and y if they are contained in different parts of the split, that is, if $|\{x,y\}\cap A|=|\{x,y\}\cap B|=1$ holds. The restriction of a split S to a set of taxa $\mathcal{X}'$, denoted by $S|_{\mathcal{X}'}$ is defined as the split $S'=A'|B'$, where $A'=A\cap\mathcal{X}'$ and $B'=B\cap\mathcal{X}'$.

Let Σ be a set of splits on $\mathcal{X}$. A *weighting* of Σ is given by a map ω that assigns a non-negative weight $\omega(S)$ to each split $S\in\Sigma$. Let Σ be a set of weighted splits on $\mathcal{X}$. We define a distance matrix of *split distances* $D(\Sigma)=\{d_{xy}\}$ on $\mathcal{X}$ by setting $d_{xy}=\sum_{S\in\Sigma(x,y)}\omega(S)$, where $\Sigma(x,y)$ denotes the set of all splits in Σ that separate the taxa x and y. If Σ is unweighted, then we use this definition with $\omega(S)=1$ for all S.

Definition 2.1 (Compatibility). Two splits $A_1\,|\,B_1$ and $A_2\,|\,B_2$ are *compatible* if at least one of the sets $A_1\cap A_2$, $A_1\cap B_2$, $B_1\cap A_2$ or $B_1\cap B_2$ is empty. Otherwise they are called *incompatible*. A collection of splits Σ is *compatible* if and only if all splits are pairwise compatible.

A set of compatible splits can always be represented by a tree (Buneman, 1971). More generally, any set of splits Σ can always be represented by a *split network* N (Dress and Huson, 2004). This is an unrooted phylogenetic network with the property that every split $S=A\,|\,B$ in Σ *is represented* in N by a set of parallel edges, such that deleting all edges in the set will result in exactly two connected components, one labeled by the taxa in A and the other labeled by B.

A class of splits of particular interest are the *circular splits*.

Definition 2.2 (Circular splits). A set of splits Σ on $\mathcal{X}$ is called *circular*, if there exists a linear ordering $\pi=(x_1,\ldots,x_n)$ of the elements of $\mathcal{X}$ *for* Σ such that each split $S\in\Sigma$ is *interval-realizable*, that is, has the form

$$S=\{x_p,x_{p+1},\ldots,x_q\}\,|\,(\mathcal{X}\setminus\{x_p,x_{p+1},\ldots,x_q\}),$$

for appropriately chosen $1<p\leq q\leq n$.

Such an ordering $\pi=(x_1,\ldots,x_n)$ is called a circular ordering for Σ, as it holds that $(x_1,\ldots,x_n)$ is a circular ordering for Σ, if and only if the inverse ordering $(x_n,x_{n-1},\ldots,x_1)$ and $(x_2,x_3,\ldots,x_n,x_1)$ both are.

Circular splits are of particular interest because any set of circular splits can always be represented by a split network that can be drawn in the plane such that no two edges intersect and all labeled nodes lie on the outside of the network, see Figure 2.

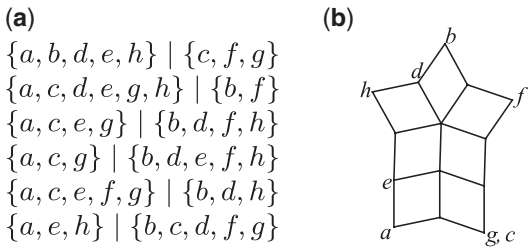

Fig. 2. (**a**) A set of six circular splits Σ on $\mathcal{X}=\{a,b,\ldots,h\}$. A circular ordering is given by (a,g,c,f,b,d,h,e). (**b**) An outer-labeled planar split network representing Σ.

2.3 The Neighbor-Net algorithm

Given a distance matrix D on $\mathcal{X}$, the Neighbor-Net algorithm (Bryant and Moulton, 2004) computes a circular ordering π of $\mathcal{X}$ from D and then a set of weighted splits Σ that are interval-realizable with respect to π. A distance matrix D on $\mathcal{X}$ is said to be *circular* if and only if there exists a set of circular weighted splits Σ on $\mathcal{X}$ such that $D(\Sigma)=D$. The following result asserts the consistency of the Neighbor-Net method:

Theorem 2.3 [Neighbor-Net consistency (Bryant *et al.*, 2007)]. Given a circular distance matrix D on $\mathcal{X}$, the Neighbor-Net algorithm produces a circular ordering π and a set of weighted splits Σ that are interval-realizable with respect to π such that $D=D(\Sigma)$.

3 TANGLEGRAMS FOR ROOTED PHYLOGENETIC NETWORKS

In this section, we first develop the concept of a tanglegram for rooted phylogenetic networks and then define what we mean by an *optimal* tanglegram. We then develop a heuristic for computing an optimal tanglegram for two rooted phylogenetic networks.

Let N be a rooted phylogenetic network on $\mathcal{X}$. A *topological embedding* of N is given by a map that assigns to each node v in

Downloaded from bioinformatics.oxfordjournals.org at Istituto di Neurologia/ on June 17, 2011

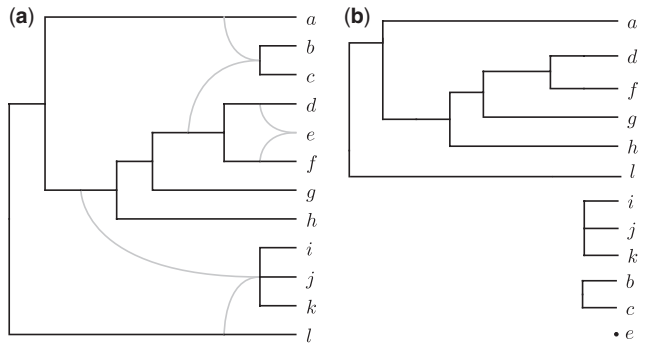

Fig. 3. (**a**) A phylogenetic network N. (**b**) A concrete drawing τ of the forest $\mathcal{F}(N)$. This drawing induces a partial order of the leaves such that $a <_\tau d <_\tau f <_\tau g <_\tau h <_\tau l$, $i <_\tau j <_\tau k$, and $b <_\tau c$.

N an ordering $(v_1, v_2, ..., v_k)$ of the set of its children. Note that any concrete drawing θ of N induces such a topological embedding that is given by the order in which edges leave a node. Moreover, such a drawing defines a total order on $\mathcal{X}$, which we will denote by π_θ.

Note that deletion of all reticulate edges in N produces a forest or collection of trees, $\{T_1, \ldots, T_k\}$, which we denote by $\mathcal{F}(N)$. For any tree T_i in $\mathcal{F}(N)$, let $\mathcal{X}_i$ denote the set of taxa that label leaves of T_i. Note that T_i is not necessarily a phylogenetic tree because some (or even all) of its leaves may be unlabeled.

A topological embedding τ for $\mathcal{F}(N)$ is given by specifying a topological embedding for each tree T_i in $\mathcal{F}(N)$. Note that τ induces a total ordering of the taxon set $\mathcal{X}_i$ for each T_i in $\mathcal{F}(N)$. While τ determines a partial ordering $<_\tau$ of $\mathcal{X}$, it does not specify a total ordering of $\mathcal{X}$ because the trees of $\mathcal{F}(N)$ are not ordered.

DEFINITION 3.1 (Non-interleaving order). Let N be a rooted phylogenetic network on $\mathcal{X}$ and let τ be a topological embedding of $\mathcal{F}(N)$. A total order π on $\mathcal{X}$ is called *non-interleaving* with respect to τ if for any two taxa $a <_\pi b$, we have:

1. If $a, b \in \mathcal{X}_i$ for some tree T_i, then $a <_\tau b$;
2. If $a \in \mathcal{X}_i$ and $b \in \mathcal{X}_j$ (with $i \neq j$), then there exists no taxa $c \in \mathcal{X}_i$ and $d \in \mathcal{X}_j$ such that $a <_\pi b <_\pi c <_\pi d$.

For example, for the phylogenetic network N on $\mathcal{X} = \{a, ..., l\}$ and the concrete drawing θ of the forest $\mathcal{F}(N)$ in Figure 3, both $(a,b,c,d,e,f,g,h,i,j,k,l)$ and $(a,d,f,g,h,l,i,j,k,b,c,e)$ are non-interleaving total orders on $\mathcal{X}$ w.r.t. θ, while $(a,d,f,g,h,i,j,l,$ $k,b,c,e)$ is not because it violates condition (2) of Definition 3.1.

What is the relevance of this definition? We want to be able to draw a rooted phylogenetic network N in such a way that we preserve the given topological embedding τ of its forest and also that we place all leaves of the network along a line in the order specified by π and the root occurs on the outside of the drawing. The non-interleaving property ensures that this can be done in such a way that no two tree edges cross.

3.1 Definition of a tanglegram for networks

Let N_1 and N_2 be two rooted phylogenetic networks on taxon sets $\mathcal{X}_1$ and $\mathcal{X}_2$, respectively. We will use $M \subseteq \mathcal{X}_1 \times \mathcal{X}_2$ to denote a set of *connectors* between $\mathcal{X}_1$ and $\mathcal{X}_2$. If the two networks are on the same taxon set, then M is the set of *identity connectors* that connects each taxon to itself, which we will denote by M_{Id} below for emphasis. In the case of a host-parasite comparison, M will pair hosts and parasites.

Unlike trees, rooted phylogenetic networks are not necessarily planar and so the definition of a tanglegram is not immediately obvious for them:

DEFINITION 3.2 (Tanglegram for networks). Let N_1 and N_2 be two rooted phylogenetic networks on $\mathcal{X}_1$ and $\mathcal{X}_2$, respectively, and let M be a set of connectors between $\mathcal{X}_1$ and $\mathcal{X}_2$. A *tanglegram* Z for N_1, N_2 and M is specified by a system $(N_1, \mathcal{X}_1, \tau_1, \pi_1, N_2, \mathcal{X}_2, \tau_2, \pi_2, M)$ where τ_i is a topological embedding of $\mathcal{F}(N_i)$ and π_i is a non-interleaving total order of $\mathcal{X}_i$ with respect to τ_i, for $i = 1, 2$.

Let N be a rooted phylogenetic network on $\mathcal{X}$. Consider a concrete drawing θ of N in the plane. We call θ a *rooted outer-labeled (tree-)planar embedding* if all taxon labels are placed on a line, the root node occurs on the outside of the embedded network and no two (tree) edges cross. If N possesses a non-interleaving order π of $\mathcal{X}$ with respect to a topological embedding τ for $\mathcal{F}(N)$, then there exists a rooted outer-labeled tree-planar embedding for N in which the taxa appear along a line in the order specified by π. As mentioned above, the non-interleaving property of π ensures that we can lineup appropriate embeddings of all the trees in $\mathcal{F}(N)$ in the order induced by π.

A *drawing* of a tanglegram $Z = (N_1, \mathcal{X}_1, \tau_1, \pi_1, N_2, \mathcal{X}_2, \tau_2, \pi_2, M)$ consists of a rooted outer-labeled tree-planar embedding of both N_1 and N_2, together with a set of lines representing the connectors between $\mathcal{X}_1$ and $\mathcal{X}_2$. Such a drawing of Z can be obtained in the following steps. First draw all trees in $\mathcal{F}(N_1)$ and $\mathcal{F}(N_2)$ in such a way that the two orderings τ_1 and τ_2 are respected and all the leaves of N_1 and N_2 are lined up in the order specified by π_1 and π_2. Second, add all reticulate edges to the diagram. These two steps can always be done in such a way that no two tree edges cross and that the roots of N_1 and N_2 occur on the outside of the drawing, due to the fact that both π_1 and π_2 are non-interleaving. Finally, draw lines between the leaves of the two networks so as to connect taxa as specified in M.

Note that, if N_1 and N_2 are two trees, $F(N_1)$ and $F(N_2)$ coincide with N_1 and N_2, respectively. This means that, in this case, a tanglegram between N_1 and N_2 define univocally the rooted outer-labeled tree-planar embeddings of N_1 and N_2. Giving an embedding, drawing a tree is straightforward (see Huson *et al.*, 2011), Chapter 13). Therefore, our definition of a tanglegram for two rooted phylogenetic networks generalizes the definition of a tanglegram for two rooted phylogenetic trees, and so, in particular, the problem of computing an optimal tanglegram for networks is NP-complete (Fernau *et al.*, 2010).

Let N be a rooted phylogenetic network on $\mathcal{X}$, let τ be a topological embedding of $\mathcal{F}(N)$ and let π be a non-interleaving total order on $\mathcal{X}$. We define the *reticulation crossing number* as the minimum number of crossings involving reticulation edges in any drawing of N respecting τ and π. An *optimal* tanglegram can now be defined as follows:

DEFINITION 3.3 (Optimal tanglegram). Let N_1 and N_2 be two rooted phylogenetic networks on $\mathcal{X}_1$ and $\mathcal{X}_2$, respectively, and let M be a set of connectors between $\mathcal{X}_1$ and $\mathcal{X}_2$. A tanglegram $Z = (N_1, \mathcal{X}_1, \tau_1, \pi_1, N_2, \mathcal{X}_2, \tau_2, \pi_2, M)$ is called *optimal* if the crossings between connectors in M is minimized by τ_1 and τ_2 and, among the

Downloaded from bioinformatics.oxfordjournals.org at Istituto di Neurologia/ on June 17, 2011

tanglegrams minimizing this value, it can be drawn so as to minimize the sum of reticulation crossing numbers for N_1 and N_2.

Let N_1 and N_2 be two rooted phylogenetic networks on $\mathcal{X}_1$ and $\mathcal{X}_2$, respectively, and let M be a set of connectors between $\mathcal{X}_1$ and $\mathcal{X}_2$. For two linear orderings π_1 and π_2 of $\mathcal{X}_1$ and $\mathcal{X}_2$, the number of crossings $Cr(\pi_1,\pi_2,M)$ among connectors in M can be calculated as $|\{(a,b)\in M\times M \mid a=(p,q), b=(x,y) \text{ with } (p<_\pi x \wedge q>_\pi y)\vee(p>_\pi x \wedge q<_\pi y)\}|$. A heuristic that computes the reticulate crossing number for a drawing of a rooted phylogenetic network is described in (Huson, 2009).

3.2 Neighbor-net heuristic for tanglegrams

Let N_1 and N_2 be two rooted phylogenetic networks on $\mathcal{X}_1$ and $\mathcal{X}_2$, respectively, and let M be a set of connectors between $\mathcal{X}_1$ and $\mathcal{X}_2$. For the purposes of this paper, we will assume that M pairs the leaves of the two trees in a one-to-one manner. (Note, however, that the example shown in Figure 1 does not have this property.) If M pairs two distinct taxa a and b, as for example in a host/parasite study, then we identify the labels a and b with each other while computing the tanglegram, but then distinguish between the two labels when drawing the tanglegram.

In the following, we present a heuristic for obtaining an optimal tanglegram for N_1 and N_2. We call this the *NN-tanglegram* approach.

In our approach, we first compute a distance matrix H on the total set of taxa $\mathcal{X}=\mathcal{X}_1\cup\mathcal{X}_2$ that reflects the topology of the two networks N_1 and N_2, then apply the Neighbor-Net algorithm to H to obtain an ordering π of $\mathcal{X}$, and finally construct a tanglegram for N_1 and N_2 based on π. In the following, we will assume that both networks contain a single leaf connected to the root of the network that is labeled by a special taxon ρ, which we will refer to as a *formal outgroup*.

To compute a distance matrix on $\mathcal{X}$, we need to construct a set of splits for each of the two networks. We describe the process for the network N_1. For each tree edge e in N_1 let $C_1(e)$ be the set of all taxa that label a leaf that lies below e in N_1. We define the split associated with e as $C(e)\,|\,\mathcal{X}_1\setminus C_1(e)$. Let $\Sigma(N_1)$ denote the set of all splits obtained in this way. We compute $\Sigma(N_2)$ similarly and we then define $\Sigma=\Sigma(N_1)|_{X_1\cap X_2}\cup\Sigma(N_2)|_{X_1\cap X_2}$. (The restriction to $\mathcal{X}_1\cap\mathcal{X}_2$ ensures the applicability of Theorem 3.5 if $\mathcal{X}_1\neq\mathcal{X}_2$).

We obtain the distance matrix H on $\mathcal{X}$ by setting the distance between two taxa x and y equal to the number of splits that separate the two taxa, where any split that occurs both for N_1 and N_2 is counted twice. In other words, we set $H=D(\Sigma(N_1)|_{X_1\cap X_2})+D(\Sigma(N_2)|_{X_1\cap X_2})$.

In the simulation study in Section 5.1, we will also present the results when using $H'=D_{\text{path}}(N_1)+D_{\text{path}}(N_2)$ as distance matrix, where $D_{\text{path}}(N_i)=\{d_{xy}\}$ such that d_{xy} is the length of the shortest path between x and y in N_i. As we will see, this variant actually performs better on networks than using splits-based distances.

We apply the Neighbor-Net algorithm to the distance matrix H (or H' in the case that the shortest path distance matrix is used) so as to obtain a circular ordering $\zeta=(x_1,\ldots,x_n)$ of $\mathcal{X}$. The ordering ζ is computed in this way because this ensures that the *NN-tanglegram* method returns the optimal solution under optimal conditions (see Theorem 3.5). Let i denote the position of the formal outgroup taxon ρ in ζ. We obtain a linear ordering π of $\mathcal{X}$ by breaking the ordering ζ at position i, that is, by setting $\pi=(x_i,x_{i+1},\ldots,x_n,x_1,\ldots,x_{i-1})$. Given the ordering π of $\mathcal{X}$, we now have to compute two embeddings τ_1 and τ_2 for the forests $\mathcal{F}(N_1)$ and $\mathcal{F}(N_2)$ such that π is non-interleaving with respect to τ_1 and τ_2. Given a rooted phylogenetic network N and a node u of N, we use $\mathcal{X}_N(u)$ to denote the set of taxa that label the leaves below u.

To compute τ_1, we first delete all reticulate edges in N_1 to produce the forest $\mathcal{F}(N_1)$. Then, for each $T^*\in\mathcal{F}(N_1)$, we determine a topological embedding τ^* that minimizes the number of crossings among connectors $Cr(\zeta^*,\pi,M_{\text{Id}})$, where ζ^* is the ordering of $\mathcal{X}^*$ induced by the embedding τ^*. This optimization is easily solved in a bottom-up traversal of each tree in $\mathcal{F}(N_1)$. Note that the place that is assigned to a node v such that $\mathcal{X}_{N_1}(v)=\emptyset$ or $\mathcal{X}_{N_1}(v)|_{\mathcal{X}_1\cap\mathcal{X}_2}=\emptyset$ in the the topological embedding of its parent is not relevant for the computation of the number of crossings among connectors $Cr(\zeta^*,\pi,M_{\text{Id}})$ and so can be chosen arbitrarily. The set of topological embeddings for all trees T^* in $\mathcal{F}(N_1)$ constitutes τ_1. To obtain π_1 from τ_1, we add the taxa of $\mathcal{X}_1$ to π_1 one by one, in such a way that π_1 remains non-interleaving w.r.t. τ_1 and the value of $Cr(\pi_1,\pi,M_{\text{Id}})$ is minimized. The ordering τ_2 is computed in exactly the same way but using the network N_2 instead of N_1.

Let $\mathcal{C}(N)$ be the set of clusters associated with N. We say that $\mathcal{C}(N)$ is *interval-realizable* with respect to $\pi=(x_1,x_2,\ldots,x_n)$ if each cluster C in $\mathcal{C}(N)$ has the form $\{x_p,x_{p+1},\ldots,x_q\}$, for appropriately chosen $1\leq p\leq q\leq n$. We have the following result:

LEMMA 3.4 (Interval realizability). Let N be a phylogenetic network on $\mathcal{X}$. If N has a rooted outer-labeled planar embedding θ and π_θ is the linear order on $\mathcal{X}$ that is defined by θ, then $\mathcal{C}(N)$ is interval-realizable with respect to π_θ.

This lemma is used to prove the following theorem:

THEOREM 3.5 (Zero crossings solution). Let N_1 and N_2 be two rooted phylogenetic networks on $\mathcal{X}_1$ and $\mathcal{X}_2$, respectively and let M be a set of connectors between $\mathcal{X}_1$ and $\mathcal{X}_2$. If a planar drawing exists for the tanglegram of N_1, N_2 and M, then the NN-tanglegram heuristic will find a solution with zero crossings among connectors.

The proof of both results can be found in Appendix A. Note that Theorem 3.5 ensures that, if an optimal tanglegram with cost zero exists for two trees, then our algorithm will find it, because a tree does not contain any reticulate edges.

This is not true for networks. Indeed, in this case the Neighbor-Net algorithm may have more than one optimal solution. Theorem 3.5 ensures that any linear ordering π computed as described in Section 3.2 can be realized with zero crossings among connectors, but it does not guarantee that the resulting drawing will have zero crossings involving reticulate edges. For example, for the two networks in Figure 4, both orders (a,b,c,d) and (b,c,d,a) are circular with respect to $H=D(\Sigma(N_1))+D(\Sigma(N_2))$ and can be obtained from the distance matrix H using Neighbor-Net. Both orderings give a solution with zero crossings among connectors; yet, while a planar drawing for (a,b,c,d) exists [see Fig. 4(a)], a drawing respecting the ordering (b,c,d,a) will contain some crossings involving reticulate edges and thus fail to be optimal [see Fig. 4(b)]. However, if all optimal solutions of Neighbor-Net given H can be considered, then the NN-tanglegram approach will find the solution with cost zero. In such a case, our algorithm can be used to solve the *planar layout* (Lozano *et al.*, 2008) or *drawability* problem (Fernau *et al.*, 2010) for two networks [solved in linear time for two binary trees in Fernau *et al.* (2010)].

Downloaded from bioinformatics.oxfordjournals.org at Istituto di Neurologia/ on June 17, 2011

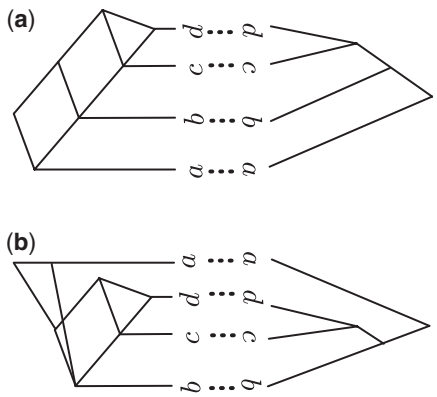

Fig. 4. A pair of networks for which our approach may fail to find the optimal solution. (**a**) An optimal ordering and (**b**) an ordering that needs to be drawn with at least one crossing.

3.3 Minimizing the reticulation crossing number and drawing the tanglegram

The previous section describes a heuristic for computing an optimal tanglegram. This heuristic aims at minimizing the crossings among connectors but it does not try to minimize the reticulation crossing numbers for N_1 and N_2. For example, if $\mathcal{X}_1 \neq \mathcal{X}_2$ then multiple choices of π_1 and π_2 can exist that minimize the number of crossing among connectors. In such a case, one of them is chosen at random, whether or not it happens to minimize the sum of reticulation crossing numbers for N_1 and N_2 (see Definition 3.3). In a forthcoming paper (Scornavacca and Huson, 2011), we will describe a heuristic for minimizing this value and a method for computing a concrete drawing of the optimized tanglegram that tries to minimize the crossing involving reticulation edges (as implemented in Dendroscope 3).

4 ALIGNMENT OF PHYLOGENETIC NETWORKS

The heuristic for calculating an optimal tanglegram for rooted phylogenetic networks that we describe in Section 3.2 is easily extended to a set of more than two rooted phylogenetic networks or trees, simply by computing the matrix H based on splits from all the networks or trees under consideration. This is implemented in our program Dendroscope 3 and can be used to 'align' the taxa when viewing a whole collection of networks or trees simultaneously.

5 VALIDATION

To validate the approach, we first report on a simulation study that we have undertaken and then apply the algorithm to a published dataset to illustrate how the algorithm may be used in practice.

5.1 Simulation study

In the first part of the simulation study, we compared our implementation with the best available software for computing tanglegrams of trees. In more detail, we compared against the *bb-1st-sol* algorithm, as it is the best performing algorithm of the five presented in Nöllenburg *et al.* (2009), and against the *lh* algorithm, as it has similar performance to the other algorithms described in Bansal *et al.* (2009), while being faster. Since these algorithms only accept binary trees, we first restricted our attention to binary trees. We compared these two published methods against our *NN-tanglegram* heuristic and two variants of it, which we will refer to as *NN-tanglegram+1S* and *NN-tanglegram+5S*. These two variants first compute π_1 and π_2 as described in Section 3.2, then define π as the order π_1 restricted to the common taxa and finally use π to compute the new orders π_1 and π_2 as described in Section 3.2. *NN-tanglegram+5S* executes this step 5 times, alternating π_1 and π_2 to compute the new π. Since the *NN-tanglegram* heuristic performs similarly on trees when using the splits-based distance H or the shortest path distance matrix H', we will present the results only for the former variant. All runs were executed on a 2.53 GHz processor with 4 GB of RAM.

For the first dataset, which we will refer to as D_1, we created 6 random binary trees on the same taxon set for five different sizes, namely on 20, 60, 100, 140 and 180 taxa. This dataset contains 15 instances to solve for each taxon set. Each instance was formulated as an ILP (integer linear program) as described in Nöllenburg *et al.* (2009) and then solved using lpSolve (freely available from lpsolve.sourceforge.net/5.5/). The number of replicates considered for each parameter setting was limited by the long running time of the ILP solver. In the second dataset, D_2, we created 10 random binary trees for each of the 5 sizes listed above, ensuring in each case that a tanglegram with zero crossings among connectors exists. This dataset contains 45 instances to solve for each taxon set.

For each tanglegram and each method, we computed the *performance ratio* (PR), that is, the ratio $(cn+1)/(cn_{opt}+1)$, where cn and cn_{opt} are the computed and the optimum number of crossings among connectors, respectively. The performance ratio values and the average running time for each method are shown in Figure 5.

For both datasets, the best-performing method is *bb-1st-sol*, having the lowest PR values and the lowest average running time. Note that this method is guaranteed to find a solution with zero crossings, if one exists (Nöllenburg *et al.*, 2009), just like our *NN-tanglegram* heuristic [see Fig. 5(d)]. However, the method is restricted to binary trees, unlike our method, which also applies to multifurcating trees. On the first dataset *lh* appears to perform well and its PR values are comparable with those of *bb-1st-sol*. However, when the number of instances per taxon set is increased (as in dataset D_2), this method can perform very badly for some instances [see Fig. 5(d)], although the average PR values remain low. Moreover, the average running time of *lh* is unacceptably high for use in an interactive visualization tool [on average >100 s when the cardinality of the taxon set is 180, see Fig. 5(a,c)].

Our new method, although designed for the general case of networks, also performs well for binary trees, while the average running time is low. Comparing the performance of *NN-tanglegram* with *NN-tanglegram+1S* and *NN-tanglegram+5S*, we can see that the *swapping* step, as expected, improves the PR values but increases the average running time. However, swapping one time (as done in *NN-tanglegram+1S*) is highly recommended because the achieved improvement of the PR values is worth the small increase in running time. In our implementation, the user can choose how many times to swap or can abort the swapping procedure after a given amount of time.

In the second part of the simulation, we studied the performance of our methods on two different network datasets. For both

Downloaded from bioinformatics.oxfordjournals.org at Istituto di Neurologia/ on June 17, 2011

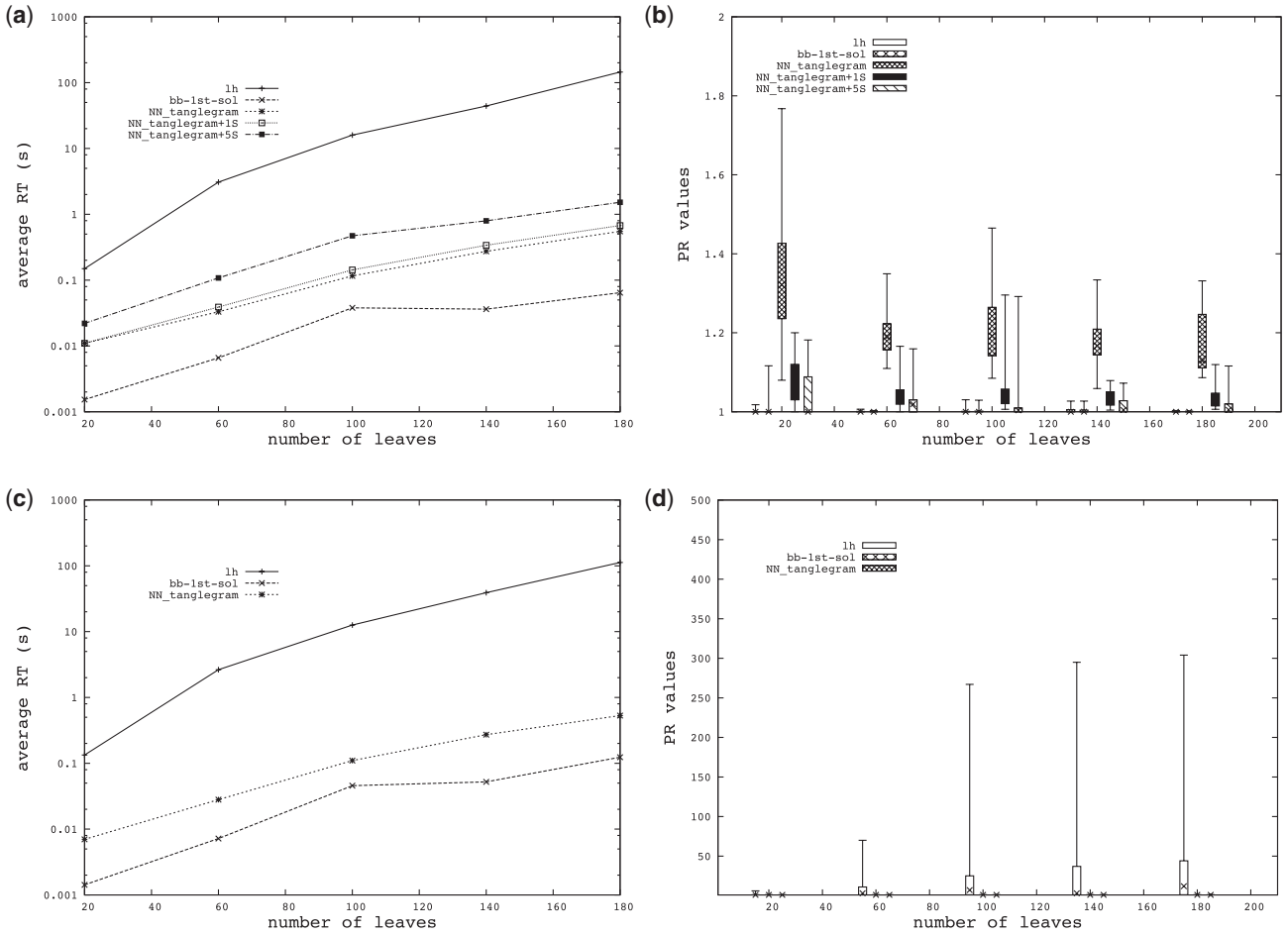

Fig. 5. (**a**) Average running time (RT) and (**b**) performance ratio values (PR) for dataset D_1. (**c**) Average running time (RT) and (**d**) performance ratio values (PR) for dataset D_2.

datasets (D_3 and D_4), we created 15 random binary networks (not necessarily bicombining) on the same taxon set for 5 different sizes, namely on 20, 60, 100, 140 and 180 taxa, ensuring in each case that a tanglegram with zero crossings among connectors exists. This leads to 105 instances to solve for each taxon set. The two datasets differ by the probability to add a reticulate edge between two nodes, which is higher for D_4. (This implies that D_4 on average contains more reticulations than D_3 and thus is a more complicated dataset than the latter.)

For both datasets, we compared the performance of *NN-tanglegram* with *NN-tanglegram+1S* and *NN-tanglegram+5S*. Since the *NN-tanglegram* heuristic performs a lot better on networks when using the shortest path distance matrix H' rather than the splits-based distance H [see Figure 6(b), *NN-tanglegram+1S* vs *NN-tanglegram_C+1S*], we will discuss the results only for the former variant.

As expected, the PR values and average running times are higher than for the binary tree datasets but still acceptably low for use in an interactive visualization tool [see Fig. 6]. The PR values increase both with the number of leaves and the number of reticulations in the networks [see Fig. 6(b,d)]. The pattern of relations among *NN-tanglegram*, *NN-tanglegram+1S* and *NN-tanglegram+5S* is the same than for the tree datasets (i.e. the swapping step improves the PR values but increases the average running time).

In general, although the average PR values remain low, the methods can produce tanglegrams with high numbers of crossings among connectors for some instances. Note that, when the number of leaves in the networks is large, the crossing number can easily be very high. For example, if the two networks under consideration have 180 leaves each and if only one taxon is incorrectly placed at the two different ends of the networks, then the crossing number will be at least 179. Nevertheless, in this case the tanglegram may still be useful for visualizing similarities and differences among the two networks.

5.2 Application to published data

Persicaria is a genus of plants in the family Polygonaceae. In (Kim and Donoghue, 2008), the authors present evidence of hybrid speciations within this genus using cpDNA regions and nuclear ITS sequences. The strict consensus tree from the most parsimonious (MP) trees and the maximum likelihood (ML) tree were computed and drawn superposed on each other for both the cpDNA and nuclear ITS datasets. A tanglegram between the two superposed drawings

Downloaded from bioinformatics.oxfordjournals.org at Istituto di Neurologia/ on June 17, 2011

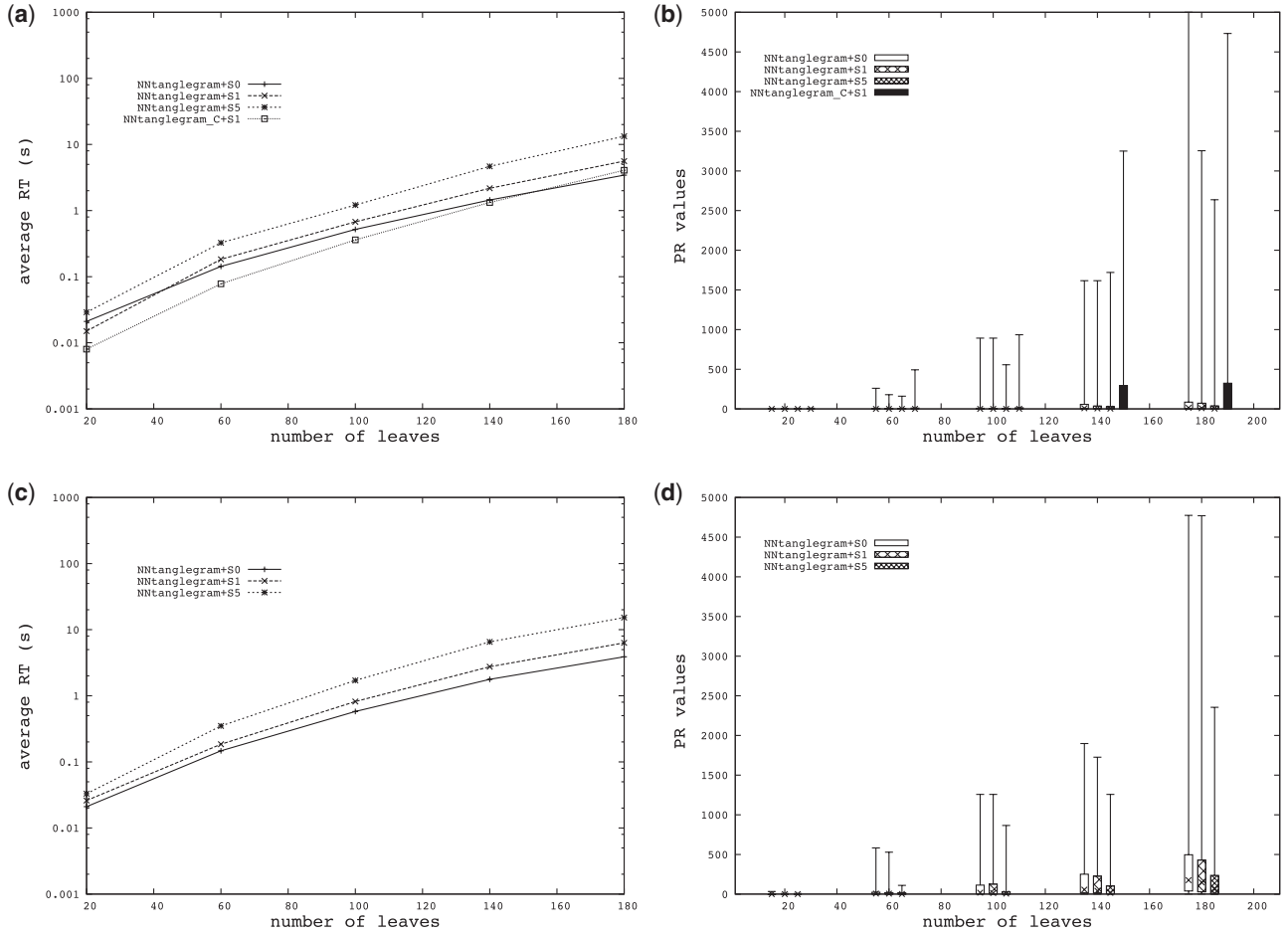

Fig. 6. (**a**) Average running time (RT) and (**b**) performance ratio values (PR) for datasets D_3. (**c**) Average running time (RT) and (**d**) performance ratio values (PR) for datasets D_4.

(one for the cpDNA dataset and nuclear ITS dataset, respectively) with crossing number among connectors 244 was shown. Here, instead of superposing the drawing of the strict consensus MP tree and the ML tree, we show both trees embedded in a network. The tanglegram between the network obtained by combining the strict consensus MP tree and the ML tree for the cpDNA dataset and the one obtained by the nuclear ITS dataset is shown in Figure 7. This tanglegram is much clearer than the original representation (see Fig. 1 of Kim and Donoghue, 2008).

6 CONCLUSIONS

Tanglegrams are a useful tool for comparing rooted phylogenetic trees. In this article, we have extended them to rooted phylogenetic networks and have described a practical approach to their computation. The simulation study proves that our new method, although designed for the general case of networks, also performs well for binary trees, while the average running time stays low. Moreover, the performance of our method on networks is good enough for use in an interactive visualization tool. Our implementation in the popular tree-drawing program Dendroscope will make tanglegrams for trees and networks easily accessible to biologists and other users.

ACKNOWLEDGEMENTS

The authors would like to thank Mukul S. Bansal and Martin Nöllenburg for providing them with an implementation of their methods.

Conflict of Interest: none declared.

REFERENCES

Bansal,M.S. *et al*. (2009) Generalized binary tanglegrams: Algorithms and applications. In *BICoB '09: Proceedings of the 1st International Conference on Bioinformatics and Computational Biology*. Springer, Berlin, Heidelberg, pp. 114–125.

Böcker,S. *et al*. (2009) A faster fixed-parameter approach to drawing binary tanglegrams. In *4th International Workshop of Parameterized and Exact Computation*, Vol. 5917 of *LNSC*, Springer, pp. 38–49.

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

Bryant,D. *et al*. (2007) Consistency of the Neighbor-Net algorithm. *Algorithms Mol. Biol.*, **2**, 8.

Downloaded from bioinformatics.oxfordjournals.org at Istituto di Neurologia/ on June 17, 2011

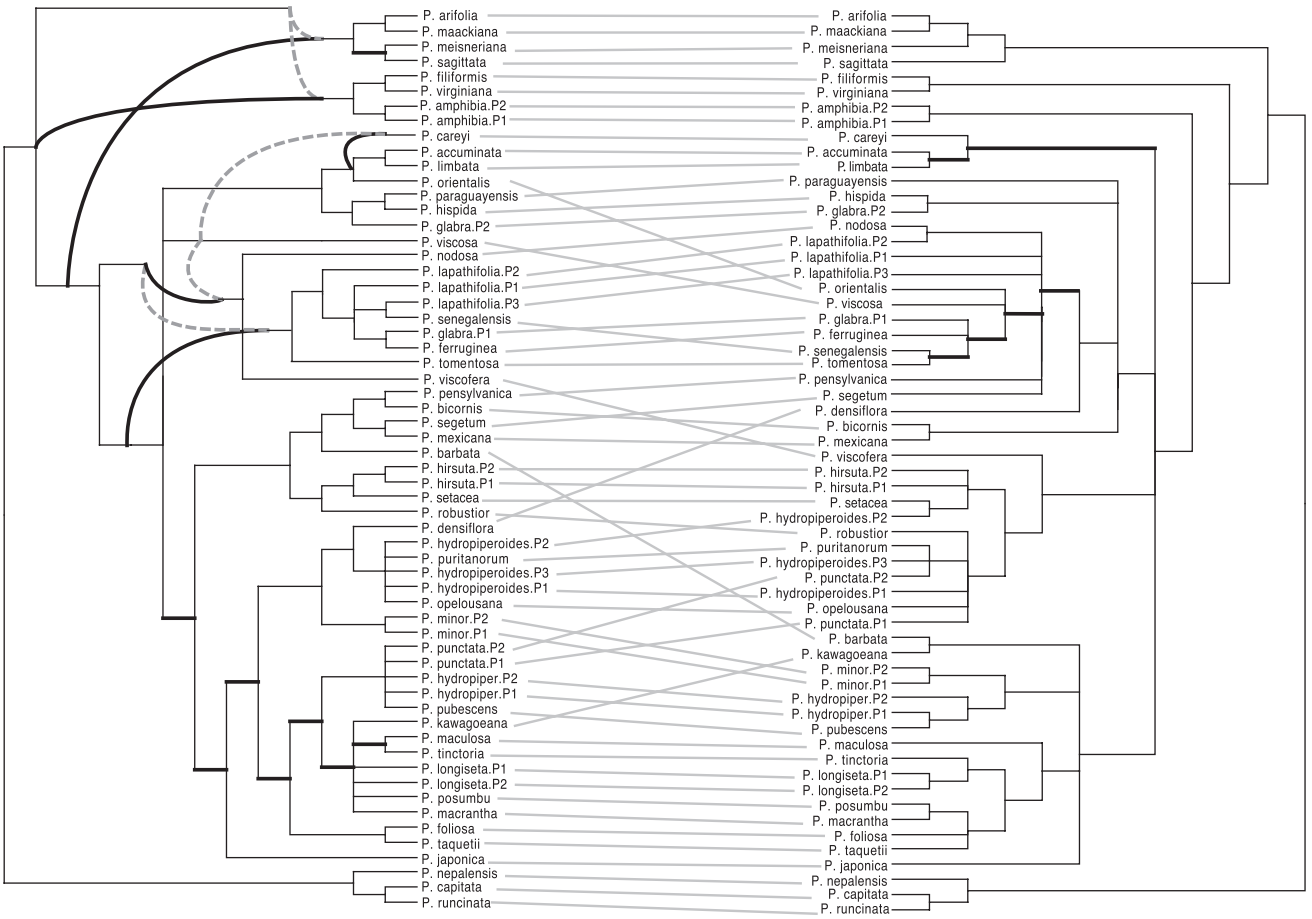

Fig. 7. A tanglegram between the networks obtained by analyzing the nrITS dataset (left) and the cpDNA dataset (right). For each dataset, the edges present only in the ML tree or in the strict consensus MP tree, are shown as bold black lines, or bold dotted gray lines, respectively. Data from (Kim and Donoghue, 2008).

Buchin,K. *et al*. (2009) Drawing (complete) binary tanglegrams: Hardness, approximation, fixed-parameter tractability. In *Proceedings of the 16th International Symposium on Graph Drawing*, Vol. 5417 of *LNCS*, Springer, pp. 324–335.

Buneman,P. (1971) The recovery of trees from measures of dissimilarity. In Hodson,F.R. *et al*. (eds) *Mathematics in the Archaeological and Historical Sciences*. Edinburgh University Press, Edinburgh, UK, pp. 387–395.

Burt,A. and Trivers,R. (2008) *Genes in Conflict: The Biology of Selfish Genetic Elements*. 1st edn. Belknap Press of Harvard University Press, Harvard, USA.

Charleston,M. (1998) Jungles: a new solution to the host/parasite phylogeny reconciliation problem. *Math. Biosci.*, **149**, 191–223.

Charleston,M.A. and Perkins,S.L. (2003) Lizards, malaria, and jungles in the caribbean. In Page,R.D.M. (ed) *Tangled Trees: Phylogeny, Cospeciation, and Coevolution*. The University of Chicago Press, Chicago, USA, pp. 65–92.

Dress,A.W.M. and Huson,D.H. (2004) Constructing splits graphs. *IEEE/ACM Trans. Comput. Biol. Bioinformatics*, **1**, 109–115.

Dwyer,T. and Schreiber,F. (2004) Optimal leaf ordering for two and a half dimensional phylogenetic tree visualisation. In *Australasian Symposium on Information Visualisation*, Vol. 35 of *CRPIT*, ACS, pp. 109–115.

Fernau,H. *et al*. (2005) Comparing trees via crossing minimization. In *The 25th Conference on Foundations of Software Technology and Theoretical Computer Science*, Vol. 3821 of *LNSC*, Springer, pp. 457–469.

Fernau,H. *et al*. (2010) Comparing trees via crossing minimization. *J. Comput. Syst.*, **76**, 593–608.

Huson,D.H. (2009) Drawing rooted phylogenetic networks. *IEEE/ACM Trans. Comput. Biol. Bioinformatics*, **6**, 103–109.

Huson,D.H. and Scornavacca,C. (2011a) A survey of combinatorial methods for phylogenetic networks. *Genome Biol. Evol.*, **3**, 23–35.

Huson,D.H. and Scornavacca,C. (2011b) Dendroscope 3 - a program for computing and drawing rooted phylogenetic trees and networks. In preparation, software available from: www.dendroscope.org.

Huson,D.H. *et al*. (2007) Dendroscope: an interactive viewer for large phylogenetic trees. *BMC Bioinformatics*, **8**, 460.

Huson,D.H. *et al*. (2011) *Phylogenetic Networks: Concepts, Algorithms and Applications*. Cambridge University Press, Cambridge, UK.

Kim,S.-T. and Donoghue,M.J. (2008) Incongruence between cpDNA and nrITS trees indicates extensive hybridization within eupersicaria (polygonaceae). *Am. J. Bot.*, **95**, 1122–1135.

Lee,M.-M. and Stock,S. (2010) A multilocus approach to assessing co-evolutionary relationships between *Steinernema* spp. (nematoda: Steinernematidae) and their bacterial symbionts *Xenorhabdus* spp. (γ-proteobacteria: Enterobacteriaceae). *Syst. Parasitol.*, **77**, 1–12.

Lozano,A. *et al*. (2008) Seeded tree alignment. *IEEE/ACM Trans. Comput. Biol. Bioinformatics*, **5**, 503–513.

Machado,C.A. *et al*. (2005) Critical review of host specificity and its coevolutionary implications in the fig/fig-wasp mutualism. *Proc. Natl Acad. Sci. USA*, **102** (Suppl. 1), 6558–6565.

Merkel,V. *et al*. (2010) Distribution and phylogeny of immunoglobulin-binding protein G in Shiga toxin-producing *Escherichia coli* and its association with adherence phenotypes. *Infect. Immun.*, **78**, 3625–3636.

Nöllenburg,M. *et al*. (2009) Drawing binary tanglegrams: an experimental evaluation. In *Proceedings of the Eleventh Workshop on Algorithm Engineering and Experiments (ALENEX)*, SIAM, pp. 106–119.

Scornavacca,C. and Huson,D. (2011) Drawing phylogenetic networks with constraints on the order of taxa. In preparation.

Downloaded from bioinformatics.oxfordjournals.org at Istituto di Neurologia/ on June 17, 2011

Venkatachalam,B. *et al*. (2010) Untangling tanglegrams: comparing trees by their drawings. *IEEE/ACM Trans. Comput. Biol. Bioinformatics*, **7**, 588–597.

APPENDIX A

A.1 Proof of Lemma 3.4

To obtain a contradiction, assume that $\mathcal{C}(N)$ is not interval-realizable with respect to an order π and there exists a rooted outer-labeled planar embedding θ such that $\pi_\theta = \pi$. Let C be a cluster in $\mathcal{C}(N)$ that is not interval-realizable with respect to π. Then there exist three taxa $a, b, c \in \mathcal{X}$ such that $a, b \in C$, $c \notin C$ and $a <_\pi c <_\pi b$. Let v be the target node of a tree edge in N that represents C and let p_1 and p_2 two paths connecting v to the leaves labeled by a and b, respectively. By definition of a rooted phylogenetic network, there exists a direct path p_3 connecting c to the root node ρ. Since any p_3 cannot include v, the Jordan curve theorem implies that p_3 has to cross p_1 or p_2, a contradiction.

A.2 Proof of Theorem 3.5

For ease of exposition, assume that $\mathcal{X}_1 = \mathcal{X}_2$ holds. Since both N_1 and N_2 can be represented by rooted outer-labeled planar graphs, it follows from Lemma 3.4 that $\mathcal{C}(N_1)$ and $\mathcal{C}(N_2)$ are interval-realizable for some orders π_1 and π_2 of $\mathcal{X}$, respectively. From the definition of $\Sigma(\cdot)$ we have that both $\Sigma(N_1)$ and $\Sigma(N_2)$ fulfill Definition 2.2 with respect to π_1 and π_2 and thus are circular split systems. Thus, by definition, $D(\Sigma(N_1))$, $D(\Sigma(N_2))$ and $D(\Sigma(N_1)) + D(\Sigma(N_2))$ are circular. It follows, from the consistency of Neighbor-Net (Theorem 2.3), that the split set $\Sigma(N_1) \cup \Sigma(N_2)$ is circular with respect to the circular ordering π computed by the Neighbor-Net algorithm. Note that also $\Sigma(N_1)$ and $\Sigma(N_2)$ are circular with respect to π and thus the linear ordering π is by definition a circular ordering for $\Sigma(N_1)$ and $\Sigma(N_2)$ too. The definitions of $\Sigma(\cdot)$ and of π imply that the sets $\mathcal{C}(N_1)$ and $\mathcal{C}(N_2)$ are interval-realizable with respect to π.

It remains to be established that the ordering π_1 (similar for π_2) returned by the Neighbor-Net tanglegram heuristic is such that $\pi_1 = \pi$. To establish this, we have to show that, if $\mathcal{C}(N_1)$ is interval-realizable with respect to π, then there exists an embedding τ_1 of $\mathcal{F}(N_1)$ such that π is a non-interleaving order w.r.t. τ_1. Note that, for each $T^* \in \mathcal{F}(N_1)$ on a taxon set $\mathcal{X}^*$, the cluster set $\mathcal{C}(T^*)$ is a subset if $\mathcal{C}(N_1)$ and thus is interval-realizable with respect to π; second, $\mathcal{C}(T^*)$ is compatible. This implies that we can construct an embedding τ^* of T^* with $\pi_{\tau^*}(T^*) = \pi|_{\mathcal{X}^*}$ that can be drawn in such a way that no two tree edges cross. Thus, the set of topological embeddings for all trees T^* in $\mathcal{F}(N)$ constitutes a topological embedding τ_1 for $\mathcal{F}(N_1)$ such that π satisfies condition (1) of Definition 3.1 w.r.t. τ_1. But π also satisfies condition (2) of Definition 3.1 w.r.t. τ_1, otherwise $\mathcal{C}(N_1)$ would not be interval-realizable with respect to π. From these observations, it follows that we can construct an embedding τ_1 for the forest $\mathcal{F}(N_1)$ such that π is a non-interleaving order w.r.t. τ_1 as described in Section 3.2. This concludes the proof.

Downloaded from bioinformatics.oxfordjournals.org at Istituto di Neurologia/ on June 17, 2011