Visualizing Co-Phylogenetic Reconciliations* 

Tiziana Calamoneri¹, Valentino Di Donato², 
Diego Mariottini², and Maurizio Patrignani²

¹ Computer Science Department, University of Rome “Sapienza”, Rome, Italy 
² Engineering Department, Roma Tre University, Rome, Italy

Abstract. We introduce a hybrid metaphor for the visualization of the 
reconciliations of co-phylogenetic trees, that are mappings among the 
nodes of two trees. The typical application is the visualization of the 
co-evolution of hosts and parasites in biology. Our strategy combines a 
space-filling and a node-link approach. Differently from traditional meth-
ods, it guarantees an unambiguous and ‘downward’ representation whenever the reconciliation is time-consistent (i.e., meaningful). We address 
the problem of the minimization of the number of crossings in the rep-
resentation, by giving a characterization of planar instances and by es-
establishing the complexity of the problem. Finally, we propose heuristics 
for computing representations with few crossings.

1 Introduction

Producing readable and compact representations of trees has a long tradition in 
the graph drawing research field. In addition to the standard node-link diagrams, 
which include layered trees, radial trees, hv-drawings, etc., trees can be visualized 
via the so-called space-filling metaphors, which include circular and rectangular 
treemaps, sunbursts, icicles, sunrays, icerays, etc. [14,15].

Unambiguous and effective representation of co-phylogenetic trees, that are 
pairs of phylogenetic trees with a mapping among their nodes, is needed in 
biological research. A phylogenetic tree is a full rooted binary tree (each node has 
zero or two children) representing the evolutionary relationships among related 
organisms. Biologists who study the co-evolution of species, such as hosts and 
parasites, start with a host phylogenetic tree \( H \), a parasite tree \( P \), and a mapping 
function \( \varphi \) (not necessarily injective nor surjective) from the leaves of \( P \) to 
the leaves of \( H \). The triple \( \langle H, P, \varphi \rangle \), called co-phylogenetic tree, is traditionally 
represented with a tanglegram drawing, that consists of a pair of plane trees 
whose leaves are connected by straight-line edges [2,3,4,9,10,12,16]. However, a 
tanglegram only represents the input of a more complex process that aims at 
computing a mapping \( \gamma \), called reconciliation, that extends \( \varphi \) and maps all the 
parasite nodes onto the host nodes.

* This research was partially supported by MIUR project “MODE – MOorphing graph 
Drawings Efficiently”, prot. 20157EFM5C_001 and by Sapienza University of Rome 
project “Combinatorial structures and algorithms for problems in co-phylogeny”.
Given $H$, $P$, and $\varphi$, a great number of different reconciliations are possible. Some of them can be discarded, since they are not consistent with time (i.e. they induce contradictory constraints on the periods of existence of the species associated to internal nodes). The remaining reconciliations are generally ranked based on some quality measure and only the optimal ones are considered. Even so, optimal reconciliations are so many that biologists have to perform a painstaking manual inspection to select those that are more compatible with their understanding of the evolutionary phenomena.

In this paper we propose a new and unambiguous metaphor to represent reconciliations of co-phylogenetic trees (Section 3). The main idea is that of representing $H$ in a suitable space-filling style and of using a traditional node-link style to represent $P$. This is the first representation guaranteeing the downward-ness of $P$ when time-consistent (i.e., meaningful) reconciliations are considered. In order to pursue readability, we study the number of crossings that are introduced in the drawing of tree $P$ (tree $H$ is always planar): on the one hand, in Section 4 we characterize planar reconciliations, on the other hand, we show in Section 5 that reducing the number of crossings in the representation of the reconciliations is NP-complete. Finally, we propose heuristics to produce drawings with few crossings (Section 6) and experimentally show their effectiveness and efficiency (Section 7). Details and full proofs can be found in [5].

2 Background

In this paper, whenever we mention a tree $T$, we implicitly assume that it is a full rooted binary tree with node set $\mathcal{V}(T)$ and arc set $\mathcal{A}(T)$, and that arcs are oriented away from the root $r(T)$ down to the set of leaves $\mathcal{V}_L(T) \subset \mathcal{V}(T)$ (see [5] for formal definitions). The lowest common ancestor of two nodes $u, v \in \mathcal{V}(T)$, denoted $lca(u, v)$, is the last common node of the two directed paths leading from $r(T)$ to $u$ and $v$. Two nodes $u$ and $v$ are comparable if $lca(u, v) \in \{u, v\}$, otherwise they are incomparable.

A tanglegram $\langle T_1, T_2, G \rangle$ generalizes a co-phylogenetic tree and consists of two generic rooted trees $T_1$ and $T_2$ and a bipartite graph $G = (\mathcal{V}_L(T_1), \mathcal{V}_L(T_2), E)$ among their leaves. In a tanglegram drawing of $\langle T_1, T_2, G \rangle$: (i) tree $T_1$ is planarly drawn above a horizontal line $l_1$ with its arcs pointing downward and its leaves on $l_1$, (ii) tree $T_2$ is planarly drawn below a horizontal line $l_2$, parallel to $l_1$, with its arcs pointing upward and its leaves on $l_2$, and (iii) edges of $G$, called tangles, are straight-line segments drawn in the horizontal stripe bounded by $l_1$ and $l_2$. A decade-old literature is devoted to tanglegram drawings (see e.g. [2,3,4,9,10,12,16]). Finding a tanglegram drawing that minimizes the number of crossings among the edges in $E$ is known to be NP-complete, even if the trees are binary trees or if the graph $G$ is a matching [10].

A reconciliation of the co-phylogenetic tree $\langle H, P, \varphi \rangle$ is a mapping $\gamma : \mathcal{V}(P) \rightarrow \mathcal{V}(H)$ that satisfies the following properties: (i) for any $p \in \mathcal{V}_L(P)$, $\gamma(p) = \varphi(p)$, that is, $\gamma$ extends $\varphi$, (ii) for any arc $(p_i, p_j) \in \mathcal{A}(P)$, $lca(\gamma(p_i), \gamma(p_j)) \neq \gamma(p_j)$, that is, a child $p_j$ of $p_i$ cannot be mapped to an ancestor of $\gamma(p_i)$ and (iii)
for any \( p \in V \setminus V_L(P) \) with children \( p_1 \) and \( p_2 \), \( \text{lca}(\gamma(p), \gamma(p_1)) = \gamma(p) \) or \( \text{lca}(\gamma(p), \gamma(p_2)) = \gamma(p) \), that is, at least one of the two children is mapped in the subtree rooted at \( \gamma(p) \).

The set of all reconciliations of \( \langle H, P, \varphi \rangle \) is denoted \( \mathcal{R}(H, P, \varphi) \).

Four types of events may take place in a reconciliation (see formal definitions in [5]): co-speciation, when both the host and the parasite speciate; duplication, when the parasite speciates (but not the host) and both parasite children remain associated with the host; loss, when the host speciates but not the parasite, leading to the loss of the parasite in one of the two host children; and host-switch, when the parasite speciates and one child remains with the current host while the other child jumps to an incomparable host.

Each of the above events is usually associated with a penalty and the minimum cost reconciliations are searched (they can be computed with polynomial delay [8,21]). However, only reconciliations not violating obvious temporal constraints are of interest. A reconciliation \( \gamma \) is time-consistent if there exists a linear ordering \( \pi \) of the parasites \( V(P) \) such that: (i) for each arc \( (p_1, p_2) \in A(P) \), \( \pi(p_1) < \pi(p_2) \); (ii) for each pair \( p_1, p_2 \in V(P) \) such that \( \pi(p_1) < \pi(p_2) \), \( \gamma(p_2) \) is not a proper ancestor of \( \gamma(p_1) \). Recognizing time-consistent reconciliations is a polynomial task [18,8,21], while producing exclusively time-consistent reconciliations is NP-complete [13,19]. This is why usually time-inconsistent reconciliations are filtered out in a post-processing step [1].

The available tools to compute reconciliations adopt three main conventions to represent them. The simplest strategy, schematically represented in Fig. 1(a), represents the two trees by adopting the traditional node-link metaphor, where the nodes of \( P \) are drawn close to the nodes of \( H \) they are associated to. Unfortunately, when several parasite nodes are associated to the same host node, the drawing becomes cluttered and the attribution of parasite nodes to host nodes becomes unclear. Further, even if \( P \) was drawn without crossings (tree \( H \) always is), the overlapping of the two trees produces a high number of crossings (see [5]).

An alternative strategy (Fig. 1(b)) consists in representing \( H \) as a background shape, such that its nodes are shaded disks and its arcs are thick pipes, while \( P \) is contained in \( H \) and drawn in the traditional node-link style. This strategy is used, for example, by CophyTrees [1], the viewer associated with the Eucalypt tool [8]. The representation is particularly effective, as it is unambiguous and
crossings between the two trees are strongly reduced, but it is still cluttered when a parasite subtree has to be squeezed inside the reduced area of a host node (see [5]).

Finally, some visualization tools adopt the strategy of keeping the containment metaphor while only drawing thick arcs of $H$ and omitting host nodes (Fig. 1(c)). This produces a node-link drawing of the parasite tree drawn inside the pipes representing the host tree. Examples include Primete [17] and SylveX [6]—see [5]. Also this strategy is sometimes ambiguous, since it is unclear how to attribute parasites to hosts.

3 A new model for the visualization of reconciliations

Inspired by recent proposals of adopting space-filling techniques to represent biological networks [20], and with the aim of overcoming the limitations of existing visualization strategies, we introduce a new hybrid metaphor for the representation of reconciliations. A space-filling approach is used to represent $H$, while tree $P$ maintains the traditional node-link representation. The reconciliation is unambiguously conveyed by placing parasite nodes inside the regions associated with the hosts they are mapped to.

More specifically, the representation of tree $H$ is a variant of a representation known in the literature with the name of icicle [11]. An icicle is a space-filling representation of hierarchical information in which nodes are represented by rectangles and arcs are represented by the contact of rectangles, such that the bottom side of the rectangle representing a node touches the top sides of the rectangles representing its children (see Fig. 2(a)). In our model, in order to contain parasite subtrees of different depths, we allow rectangles of different height. Also we force all leaves of $H$ (i.e. present-day hosts) to share the same bottom line that intuitively represents current time.

Formally, an $HP$-drawing $\Gamma(\gamma)$ of $\gamma \in \mathcal{R}(H, P, \varphi)$ is the simultaneous representation of $H$ and $P$ as follows. Tree $H$ is represented in a space-filling fashion such that: (1) nodes of $H$ are represented by internally disjoint rectangles that cover the drawing area; the rectangle corresponding to the root of $H$ covers the top border of the drawing area while the rectangles corresponding to the leaves of $H$ touch the bottom border of the drawing area with their bottom sides; and
(2) arcs of \( H \) are represented by the vertical contact of rectangles, the upper rectangle being the parent and the lower rectangle being the child. Conversely, tree \( P \) is represented in a node-link style such that: (1) each node \( p \in V(P) \) is drawn as a point in the plane inside the representation of the rectangle corresponding to node \( \gamma(p) \); and (2) each arc \( (p_1, p_2) \in A(P) \) is drawn as a vertical segment if \( p_1 \) and \( p_2 \) have the same \( x \)-coordinate; otherwise, it is drawn as a horizontal segment followed by a vertical segment.

It can be assumed that an HP-drawing only uses integer coordinates. In particular the corners of the rectangles representing the nodes of \( H \) could exclusively use even coordinates and the nodes of \( P \) could exclusively use odd coordinates.

Graphically, since the icicle represents a binary tree, we give the rectangles a slanted shape in order to ease the visual recognition of the two children of each node (see Fig. 2(b)). Also, the bend of an arc of \( P \) is a small circular arc.

We say that HP-drawing \( \Gamma(\gamma) \) is planar if no pair of arcs of \( P \) intersect except, possibly, at a common endpoint, and that it is downward if, for each arc \( (p_1, p_2) \in A(P) \), parasite \( p_1 \) has a \( y \)-coordinate greater than that of parasite \( p_2 \).

4 Planar instances and reconciliations

In this section we characterize the reconciliations that can be planarly drawn, showing that a time-consistent reconciliation is planar if and only if the corresponding co-phylogenetic tree admits a planar tanglegram drawing.

**Theorem 1.** Given a co-phylogenetic tree \( \langle H, P, \varphi \rangle \), the following statements are equivalent: (1) \( \langle H, P, \varphi \rangle \) admits a planar tanglegram drawing \( \Delta \). (2) Every time-consistent reconciliation \( \gamma \in \mathcal{R}(H, P, \varphi) \) admits a planar downward HP-drawing \( \Gamma(\gamma) \).

**Sketch of proof.** First, we prove that (2) implies (1). Consider a planar drawing \( \Gamma(\gamma) \) of \( \gamma \in \mathcal{R}(H, P, \varphi) \) and let \( l \) be the horizontal line passing through the bottom border of \( \Gamma(\gamma) \). Observe that the leaves of \( P \) lie above \( l \). Construct a tanglegram drawing \( \Delta \) of \( \langle H, P, \varphi \rangle \) as follows: (a) Draw \( H \) by placing each node \( h \in V(H) \) in the center of the rectangle representing \( h \) in \( \Gamma(\gamma) \) and by representing each arc \( a \in A(H) \) as a suitable curve between its incident nodes; (b) draw \( P \) in \( \Delta \) as a mirrored drawing with respect to \( l \) of the drawing of \( P \) in \( \Gamma(\gamma) \); (c) connect each leaf \( p \in L(P) \) to the host \( \gamma(p) \) with a straight-line segment. It is immediate that \( \Delta \) is a tanglegram drawing of \( \langle H, P, \varphi \rangle \) and that it is planar whenever \( \Gamma(\gamma) \) is.

Proving that (1) implies (2) is more laborious. Let \( \Delta \) be a planar tanglegram drawing of \( \langle H, P, \varphi \rangle \). We construct a drawing \( \Gamma(\gamma) \) of the given time-consistent reconciliation \( \gamma \in \mathcal{R}(H, P, \varphi) \) as follows. First, insert into the arcs of \( P \) dummy nodes of degree two to represent losses, obtaining a new tree \( P' \). Since \( \gamma \) is time-consistent, consider any ordering \( \pi' \) of \( V(P') \) consistent with \( H \). Remove from \( \pi' \) the leaves of \( P \) and renumber the remaining nodes obtaining a new ordering \( \pi \) from 1 to \( |V(P')| - |L(P')| \). Regarding \( y \)-coordinates: all the leaves of \( P' \) have \( y \)-coordinate 1, that is, they are placed at the bottom of the drawing, while
each internal node \( p \in V(P') \setminus V_L(P') \) has \( y \)-coordinate \( 2\pi(p) + 1 \). Regarding \( x \)-coordinates: each leaf \( p \in V_L(P) \) has \( x \)-coordinate \( 2\sigma(p) + 1 \), where \( \sigma(p) \) is the left-to-right order of the leaves of \( T_2 \) in \( \Delta \). The \( x \)-coordinate of an internal node \( p \) of \( P \) is copied from one of its children \( p_1 \) or \( p_2 \), arbitrarily chosen if none of them is connected by a host-switch, the one (always present) that is not connected by a host-switch otherwise.

Let \( h \) be a node of \( V(H) \); rectangle \( R_h \), representing \( h \) in \( \Gamma \), has the minimum width that is sufficient to span all the parasites contained in the subtree \( T_h(H) \) of \( H \) rooted at \( h \) (hence, it spans the interval \([x_{min} - 1, x_{max} + 1]\), where \( x_{min} \) and \( x_{max} \) are the minimum and maximum \( x \)-coordinates of a parasite contained in \( T_h(H) \), respectively). The top border of \( R_h \) has \( y \)-coordinate \( y_{min} - 1 \), where \( y_{min} \) is the minimum \( y \)-coordinate of a parasite node contained in the parent of \( h \). The bottom border of \( R_h \) is \( y_{min} - 1 \), where \( y_{min} \) is the minimum \( y \)-coordinate of a parasite node contained in \( h \).

The proof concludes by showing that the obtained representation \( \Gamma(\gamma) \) is planar and downward \[5\].

We remark that a statement analogous to the one of Theorem \[1\] can be proved also for the visualization strategy schematically represented in Fig. \[1\]b) and adopted, for example, by CophyTrees \[1\].

The algorithm we actually implemented, called PlanarDraw, is a refinement of the one described in the proof of Theorem \[1\]. It assigns to the parent parasite an \( x \)-coordinate that is intermediate between those of the children whenever both children are not host-switches and it produces a more compact representation with respect to the \( y \)-axis (see Figs. \[2\]b) and \[3\]).
5 Minimizing the number of crossings

In this section we focus on non-planar instances and prove that computing an HP-drawing of a reconciliation with the minimum number of crossings is NP-complete. Given a reconciliation $\gamma \in \mathcal{R}(H, P, \varphi)$ and a constant $k$, we consider the decision problem RECONCILIATION LAYOUT (RL) that asks whether there exists an HP-drawing of $\gamma$ that has at most $k$ crossings. We prove that RL is NP-hard by reducing to it the NP-complete problem TWO-TREES CROSSING MINIMIZATION (TTCM) [10]. The input of TTCM consists of two binary trees $T_1$ and $T_2$, whose leaf sets are in one-to-one correspondence, and a constant $k$. The question is whether $T_1$ and $T_2$ admit a tanglegram drawing with at most $k$ crossings among the tangles. In [4] it is shown that TTCM remains NP-complete even if the input trees are two complete binary trees of height $h$ (hence, with $2^h$ leaves). We reduce this latter variant to RL.

**Theorem 2.** Problem RL is NP-complete.

*Sketch of proof.* Problem RL is in NP by exploring all possible HP-drawings of $\gamma$. Let $I_{TTCM} = (T_1, T_2, \psi, k)$ be an instance of TTCM, where $T_1$ and $T_2$ are complete binary trees of height $h$, $\psi$ is a one-to-one mapping between $V_L(T_1)$ and $V_L(T_2)$, and $k$ is a constant. We show how to build an equivalent instance $I_{RL} = (\gamma \in \mathcal{R}(H, P, \varphi), k')$ of RL.

First we introduce a gadget, called ‘sewing tree’, that will help in the definition of our instance. A sewing tree is a subtree of the parasite tree whose nodes are alternatively assigned to two host leaves $h_1$ and $h_2$ as follows. A single node $p_0$ with $\gamma(p_0) = h_2$ is a sewing tree $S_0$ of size 0 and root $p_0$. Let $S_m$ be a sewing tree of size $m$ and root $p_m$ such that $\gamma(p_m) = h_2$ (\gamma(p_m) = h_1, respectively). In order to obtain $S_{m+1}$ we add a node $p_{m+1}$ with $\gamma(p_{m+1}) = h_1$ ($\gamma(p_{m+1}) = h_2$, respectively) and two children, $p_m$ and $p'_m$, with $\gamma(p'_m) = h_1$ ($\gamma(p'_m) = h_2$, respectively). See Fig. 4 for examples of sewing trees. Intuitively, a sewing tree has the purpose of making costly from the point of view of the number of crossings the insertion of a host node $h_3$ between hosts $h_1$ and $h_2$, whenever $h_3$ contains several vertical arcs of $P$ towards leaves of the subtree rooted at $h_3$.

Nodes $r(H), h_1, h_2, \ldots, h_8$ of the host tree $H$ and their relationships are depicted in Fig. 5. Rooted at $h_5$ and $h_8$ we have two complete binary trees of
Fig. 5: The construction of the instance of RL starting from an instance of TTCM with $h = 2$. Filled green and pink are the subtrees of $H$ whose embeddings correspond to the embeddings of $T_1$ and $T_2$, respectively.

height $h$. Intuitively, these two subtrees of $H$ correspond to $T_1$ and $T_2$, respectively (they are drawn filled green and filled pink in Fig. 5). Hence, the leaves $l_{1,1}, l_{1,2}, \ldots, l_{1,2^h}$ of $T_1$ are associated to the leaves $h_{1,1}, h_{1,2}, \ldots, h_{1,2^h}$ of the subtree rooted at $h_3$, and, similarly, the leaves $l_{2,1}, l_{2,2}, \ldots, l_{2,2^h}$ of $T_2$ are associated to the leaves $h_{2,1}, h_{2,2}, \ldots, h_{2,2^h}$ of the subtree rooted at $h_8$.

The root $r(P)$ of $P$ has $\gamma(r(P)) = r(H)$. One child of $r(P)$ is the root of a sewing tree between $h_3$ and $h_6$. The other child $p_1$, with $\gamma(p_1) = r(H)$, has one child that is the root of a sewing tree between $h_3$ and $h_7$, and one child $p_2$, with $\gamma(p_2) = h_2$. Parasite $p_2$ is the root of a complete binary tree $T_h$ of height $h$, whose internal nodes are assigned to $h_2$, while the leaves are assigned to $h_3$. Each one of the $2^h$ leaves of $T_h$ is associated with a tangle of the instance $I_{TTCM}$. Namely, suppose $e = (l_{i,j})$ is a tangle edge in the instance $I_{TTCM}$. Then, an arbitrary leaf $p_e$ of $T_h$ is associated with $e$. Node $p_e$ has children $p_{1,i}$, with $\gamma(p_{1,i}) = h_{1,i}$, and $p'_{1,i}$, with $\gamma(p'_{1,i}) = h_3$. Node $p_{1,i}'$, in turn, has children $p_{2,j}$, with $\gamma(p_{2,j}) = h_{2,j}$, and $p''_{1,i}$, with $\gamma(p''_{1,i}) = h_3$. Finally, we pose $k' = k + 2^h \cdot (2^h - 1)$.

The proof concludes by showing that instance $I_{TTCM}$ is a yes instance of TTCM if and only if instance $I_{RL}$ is a yes instance of RL.

Since in the proof of Theorem 2 a key role is played by host-switch arcs, one could wonder whether an instance without host-switches is always planar. This is not the case: for any non-planar time-consistent reconciliation $\gamma \in R(H, P, \varphi)$, there exists a time-consistent reconciliation $\gamma_r \in R(H, P, \varphi)$ that maps all internal nodes of $P$ to $r(H)$ and that has no host-switch. If the absence of host-
switches could guarantee planarity, $\gamma_r$ would be planar and, by Theorem 1, also $\gamma$ would be planar, leading to a contradiction. Indeed, it is not difficult to construct reconciliations without host-switches and not planar [5].

6 Heuristics for drawing reconciliations with few crossings

Theorem 2 shows that a drawing of a reconciliation with the minimum number of crossings cannot be efficiently found. For this reason, we propose two heuristics aiming at producing HP-drawings with few crossings (Fig. 6 shows two examples of non-planar HP-drawings produced by the heuristics). In the following we will briefly describe them.

6.1 Heuristic SearchMaximalPlanar

This heuristic is based on the strategy of first drawing a large planar sub-instance and then adding non-planar arcs. We hence construct a maximal planar subgraph $G_{pl}$ of tanglegram $\langle H, P, \varphi \rangle$ by adding to it one by one the following objects: (i) all nodes of $H$ and of $P$; (ii) all arcs of $H$; (iii) edge $(r(H), r(P))$; (iv) for each $l_p \in V_L(P)$, edge $(l_p, \varphi(l_p))$; (v) for each $p \in V(P) \setminus V_L(P)$, edge $(p, p')$, where $p'$ is any child of $p$ that is not a host-switch, while the arc from $p$ to the sibling of $p'$ is added to a set $\text{missingArCs}$; (vi) all arcs from $\text{missingArCs}$ that is possible to add without introducing crossings (all arcs that have not been inserted in $G_{pl}$ are stored in a set of $\text{non-planarArCs}$). A planar embedding of the graph $G_{pl}$ is used as input for Algorithm PlanarDraw so obtaining a planar drawing of part of reconciliation $\gamma$; arcs in $\text{non-planarArCs}$ are added in a post-processing step.

6.2 Heuristic ShortenHostSwitch

This heuristic is based on the observation that ‘long’ host-switch arcs are more likely to cause crossings than ‘short’ ones. Hence, this heuristic searches for an
embedding of $H$ that reduces the distance between the end-nodes of host-switch arcs of $P$. To do this, as a preliminary step, ShortenHostSwitch chooses the embedding of $H$ with a preorder traversal as follows. Let $v \in \mathcal{V}(H)$ be the current node of the traversal. Consider the set of nodes of $H$ that are ancestors or descendants of $v$. The removal of this set would leave two connected components, one on the left, denoted $V_{v,\text{left}}(H) \subseteq \mathcal{V}(H)$ and one on the right, denoted $V_{v,\text{right}}(H) \subseteq \mathcal{V}(H)$. Denote by $V_{v,\text{left}}(P)$ ($V_{v,\text{right}}(P)$, respectively) the set of parasite nodes mapped to some node in $V_{v,\text{left}}(H)$ ($V_{v,\text{right}}(H)$, respectively). Moreover, denote by $V_p(P) \subseteq \mathcal{V}(P)$ the set of the parasite nodes mapped to the subtree of $H$ rooted at $v$.

If $v \in \mathcal{V}_L(H)$ no embedding choice has to be taken for $v$. Otherwise let $v_1$ and $v_2$ be its children. For $i \in \{1, 2\}$ and $X \in \{\text{left, right}\}$ compute the number $h_{v_i,X}$ of the host-switch arcs from $V_{v_i}(P)$ to $V_{v,X}(P)$ or vice versa. If $h_{1,\text{right}} + h_{2,\text{left}} > h_{2,\text{right}} + h_{1,\text{left}}$ then $v_1$ is embedded as the right child and $v_2$ as the left child of $v$, otherwise $v_2$ will be the right child and $v_1$ the left child.

Observe that the sets $V_{v,\text{left}}(P)$ and $V_{v,\text{right}}(P)$ can be efficiently computed while descending $H$. Namely, we start with $V_{r(H),\text{left}}(P) = V_{r(H),\text{right}}(P) = \emptyset$ and, supposing $v_l$ and $v_r$ are chosen to be the left and right children of $v$, respectively, we set $V_{v_l,\text{left}}(P) = V_{v,\text{left}}(P)$, $V_{v_l,\text{right}}(P) = V_{v,\text{right}}(P) \cup V_v$, $V_{v_r,\text{left}}(P) = V_{v,\text{left}}(P) \cup V_{v_l}$, and $V_{v_r,\text{right}}(P) = V_{v,\text{right}}(P)$.

It remains to describe how ShortenHostSwitch places parasite nodes inside the representation of host nodes. First, we temporarily assign to each node $p \in \mathcal{V}(P)$ the lower $x$– and $y$–coordinates inside $\gamma(p)$ (observe that all nodes mapped to the same host are overlapped). For the leaves $\mathcal{V}(P)$ the temporary $y$-coordinate is definitive and only the $x$-coordinate has to be decided. We order the parasite leaves $p_1, p_2, \ldots, p_k$ inside each host leaf $v_k$ as follows. We divide the leaves into two sets $L_{v,\text{left}}(P)$ and $L_{v,\text{right}}(P)$, where $L_{v,\text{left}}(P)$ contains the leaves associated with $v$ that have a parent with lower $x$-coordinates and $L_{v,\text{right}}(P)$ contains the remaining leaves associated with $v$. We order the set $L_{v,\text{left}}(P)$ ($L_{v,\text{right}}(P)$, respectively) ascending (descending, respectively) based on the $y$-coordinates of their parents. We place the set $L_{v,\text{left}}(P)$ and then the $L_{v,\text{right}}(P)$ inside $v$ according to their orderings. Once the leaves of $P$ have been placed, the remaining internal nodes of $P$ are placed according to the same algorithm used for planar instances by PlanarDraw.

7 Experimental evaluation

We collected standard co-phylogenetic tree instances from the domain literature. Table 1 shows their properties.

Since reconciliations obtained from planar co-phylogenetic trees are always planar, we restricted our experiments to non-planar instances. In order to obtain a datasuite of reconciliations we used the Eucalypt tool [8] to produce the set of minimum-cost reconciliations of each instance with costs 0, 2, 1, and 3 for co-speciation, duplication, loss, and host-switch, respectively. We configured the
| Instance                  | Acronym | # hosts | # par. | Planar |
|---------------------------|---------|---------|--------|--------|
| Caryophyllaceae & Microbotryum [21] | CM      | 35      | 39     | No     |
| Stinkbugs & Bacteria [21]  | SB      | 27      | 23     | Yes    |
| Encyrtidae & Coccidae [8] | EC      | 13      | 19     | Yes    |
| Fishs & Dactylogyrus [8]   | FD      | 39      | 101    | No     |
| Gopher & Lice [8]          | GL      | 15      | 19     | No     |
| Seabirds & Chewing Lice [8]| SC      | 21      | 27     | No     |
| Rodents & Hantaviruses [8] | RH      | 67      | 83     | No     |
| Smut Fungi & Caryophill. plants [8] | SFC | 29 | 31 | No |
| Pelican & Lice (ML) [8]    | PML     | 35      | 35     | Yes    |
| Pelican & Lice (MP) [8]    | PMP     | 35      | 35     | Yes    |
| Rodents & Pinworms [8]     | RP      | 25      | 25     | No     |
| Primates & Pinworms [8]    | PP      | 71      | 81     | No     |
| COG2085 [8]                | COG2085 | 199    | 87     | No     |
| COG4965 [8]                | COG4965 | 199    | 59     | No     |
| COG3715 [8]                | COG3715 | 199    | 79     | No     |
| COG4964 [8]                | COG4964 | 199    | 53     | No     |

Table 1: The co-phylogenetic trees used to generate the datasuite.

We implemented the two heuristics SearchMaximalPlanar and Shorten-HostSwitch in JavaScript (but we used Python for accessing the file system and the GDToolkit library [7] for testing planarity) and run the experiments on a Linux laptop with 7.7 GiB RAM and quad-core i5-4210U 1.70 GHz processor.

Table 2 shows the results of the experiments. Planar instances SB, EC, PML, and PMP were not used to generate reconciliations. Also, instances COG3715 and COG4965 did not produce any time-consistent reconciliation. For all the other phylogenetic-trees, the second column of Table 2 shows the number of reconciliations computed by Eucalypt (we bounded to 100 the reconciliations of RH, COG2085, and COG4965). Table 2 is vertically divided into three sections, each devoted to a different heuristics. Each section shows the minimum, maximum, and average number of crossings and the average computation time for the HP-drawings produced by the heuristics on the reconciliations obtained for the phylogenetic-tree specified in the first column.

The section labeled SearchMaximalPlanar* shows the results of SearchMaximalPlanar where we computed the embedding of tree $H$ (which is the most expensive algorithmic step) once for all the reconciliations of the same instance. Hence, differently from the other two sections, the computation times reported in this section refer to the sum of computation times for all the reconciliations obtained from the same instance.

From Table 2 it appears that heuristic SearchMaximalPlanar is much slower than ShortenHostSwitch. This could have been predicted, since SearchMaximalPlanar runs a planarity test several times. However, the gain in terms of cross-
The results of the experiments are shown in Table 2:

| Inst. | #Rec. | #Crossings | Avg | #Crossings | Avg | #Crossings | Avg | ms |
|-------|-------|------------|-----|------------|-----|------------|-----|----|
| CM    | 64    | 30         | 15  | 21         | 0.5 | 21         | 13  | 17 | 644 |
| FD    | 80    | 84         | 55  | 69         | 1   | 108        | 74  | 92 | 7289 |
| GL    | 2     | 1          | 1   | 1          | 0   | 1          | 1   | 1  | 180 |
| PP    | 72    | 6          | 2   | 3          | 1   | 4          | 3   | 3  | 4840 |
| RH    | 100   | 11         | 11  | 2          | 11  | 9          | 10  | 10 | 1710 |
| RP    | 1     | 4          | 2   | 3          | 1   | 3          | 3   | 3  | 737  |
| SC    | 1     | 6          | 6   | 6          | 0   | 4          | 4   | 4  | 499  |
| SFC   | 16    | 22         | 11  | 17         | 0   | 16         | 12  | 13 | 412  |
| COG2085|100  | 80         | 58  | 70         | 7   | 95         | 84  | 89 | 20540 |
| COG4965|100  | 125        | 79  | 97         | 8   | 68         | 57  | 60 | 9901  |

Table 2: The results of the experiments.

ings is questionable. Although there are instances where SearchMaximalPlanar appears to outperform ShortenHostSwitch (for example, CM, PP) this is hardly a general trend. We conclude that aiming at planarity is not the right strategy for minimizing crossings in this particular application context.

The strategy of computing the embedding of $H$ once for all reconciliations of the same co-phylogenetic tree (central section labeled SearchMaximalPlanar∗ of Table 2) seems to be extremely effective in reducing computation times. For example, on instance COG2085, where this heuristics needed 20.5 seconds, it actually used 205 msec per reconciliation, about 11% of the time needed by SearchMaximalPlanar, at the cost of very few additional crossings.

8 Conclusions and Future Work

This paper introduces a new and intriguing simultaneous visualization problem, i.e. producing readable drawings of the reconciliations of co-phylogenetic trees. Also, a new metaphor is proposed that takes advantage both of the space-filling and of the node-link visualization paradigms. We believe that such a hybrid strategy could be effective for the simultaneous visualization needs of several application domains.

As future work, we would like to address the problem of visually exploring and analyzing sets of reconciliations of the same co-phylogenetic tree, which is precisely the task that several researchers in the biological field need to perform. Heuristic SearchMaximalPlanar∗ is a first step in this direction, since it maintains the mental map of the user by fixing the drawing of $H$. Finally, we would like to adapt heuristics for the reduction of the crossings of tanglegram drawings, such as those in [3,12,16], to our problem and we would like to perform user tests to assess the effectiveness of the proposed metaphor.
Acknowledgments

We thank Riccardo Paparozzi for first experiments on the visualization of co-phylogenetic trees. Moreover, we are grateful to Marie-France Sagot and Blerina Sinaimeri for proposing us the problem and for the interesting discussions.

References

1. CophyTrees – viewer associated with [8], http://eucalypt.gforge.inria.fr/viewer.html
2. Bansal, M.S., Chang, W.C., Eulenstein, O., Fernández-Baca, D.: Generalized binary tanglegrams: Algorithms and applications. In: Rajasekaran, S. (ed.) Bioinformatics and Computational Biology, Lecture Notes in Computer Science, vol. 5462, pp. 114–125. Springer Berlin Heidelberg (2009)
3. Böcker, S., Hüffner, F., Truß, A., Wahlström, M.: A faster fixed-parameter approach to drawing binary tanglegrams. In: Chen, J., Fomin, F.V. (eds.) IWPEC 2009, LNCS, vol. 5917, pp. 38-49. Springer (2009)
4. Buchin, K., Buchin, M., Byrka, J., Nölleburg, M., Okamoto, Y., Silveira, R.I., Wolff, A.: Drawing (complete) binary tanglegrams. Algorithmica 62(1-2), 309–332 (2012)
5. Calamoneri, T., Di Donato, V., Mariottini, D., Patrignani, M.: Visualizing co-phylogenetic reconciliations. Tech. Report arXiv:1708.09691, Cornell University (2017), http://arxiv.org/abs/1708.09691
6. Chevenet, F., Doyon, J.P., Scornavacca, C., Jacox, E., Jousselin, E., Berry, V.: SylvX: a viewer for phylogenetic tree reconciliations. Bioinformatics 32(4), 608–610 (2016)
7. Di Battista, G., Didimo, W.: Gdtoolkit. In: Tamassia, R. (ed.) Handbook on Graph Drawing and Visualization., pp. 571–597. Chapman and Hall/CRC (2013)
8. Donati, B., Baudet, C., Sinaimeri, B., Crescenzi, P., Sagot, M.F.: EUCALYPT: Efficient tree reconciliation enumerator. Algorithms for Molecular Biology 10(3) (2015)
9. Dwyer, T., Schreiber, F.: Optimal leaf ordering for two and a half dimensional phylogenetic tree visualisation. In: Proceedings of the 2004 Australasian symposium on Information Visualisation - Volume 35. pp. 109–115. APVis ’04, Australian Computer Society, Inc., Darlinghurst, Australia, Australia (2004)
10. Fernau, H., Kaufmann, M., Poths, M.: Comparing trees via crossing minimization. J. Comput. Syst. Sci. 76(7), 593–608 (2010)
11. Kruskal, J.B., Landwehr, J.M.: Icicle plots: Better displays for hierarchical clustering. The American Statistician 37(2), 162–168 (1983)
12. Nölleburg, M., Völker, M., Wolff, A., Holten, D.: Drawing binary tanglegrams: An experimental evaluation. In: Finocchi, I., Hershberger, J. (eds.) ALENEX 2009. pp. 106–119. SIAM (2009)
13. Ovadia, Y., Fielder, D., Conow, C., Libeskind-Hadas, R.: The co phylogeny reconstruction problem is NP-complete. J. Comput. Biol. 18(1), 59–65 (jan 2011)
14. Rusu, A.: Tree drawing algorithms. In: Tamassia, R. (ed.) Handbook on Graph Drawing and Visualization., pp. 155–192. Chapman and Hall/CRC (2013)
15. Schulz, H.J.: Treevis.net: A tree visualization reference. IEEE Computer Graphics and Applications 31(6), 11–15 (Nov 2011)
16. Scornavacca, C., Zickmann, F., Huson, D.H.: Tanglegrams for rooted phylogenetic trees and networks. Bioinformatics 13(27), i248–56 (jul 2011)
17. Sennblad, B., Schreil, E., Sonnhammer, A.C.B., Lagergren, J., Arvestad, L.: primetv: a viewer for reconciled trees. BMC Bioinformatics 8(1), 148 (2007)
18. Stolzer, M., Lai, H., Xu, M., Sathaye, D., Vernot, B., Durand, D.: Inferring duplications, losses, transfers and incomplete lineage sorting with nonbinary species trees. Bioinformatics 28(18), i409–i415 (2012)
19. Tofigh, A., Hallett, M.T., Lagergren, J.: Simultaneous identification of duplications and lateral gene transfers. IEEE/ACM Trans. Comput. Biology Bioinform. 8(2), 517–535 (2011)
20. Tollis, I.G., Kakoulis, K.G.: Algorithms for visualizing phylogenetic networks. In: Hu, Y., Nöllenburg, M. (eds.) Graph Drawing and Network Visualization, GD 2016. LNCS, vol. 9801, pp. 183–195. Springer (2016)
21. Wieseke, N., Hartmann, T., Bernt, M., Middendorf, M.: Cophylogenetic reconciliation with ILP. IEEE/ACM Trans Comput Biol Bioinform 12(6), 1227–1235 (Nov-Dec 2015)