A simple fixed parameter tractable algorithm for computing the hybridization number of two (not necessarily binary) trees

Teresa Piovesan, Steven Kelk

Abstract—Here we present a new fixed parameter tractable algorithm to compute the hybridization number r of two rooted, not necessarily binary phylogenetic trees on taxon set $\mathcal{X}$ in time $(6^r r!) \cdot poly(n)$, where $n = |\mathcal{X}|$. The novelty of this approach is its use of *terminals*, which are maximal elements of a natural partial order on $\mathcal{X}$, and several insights from the softwired clusters literature. This yields a surprisingly simple and practical bounded-search algorithm and offers an alternative perspective on the underlying combinatorial structure of the hybridization number problem.

Index Terms—phylogenetic network, fixed parameter tractability, nonbinary.

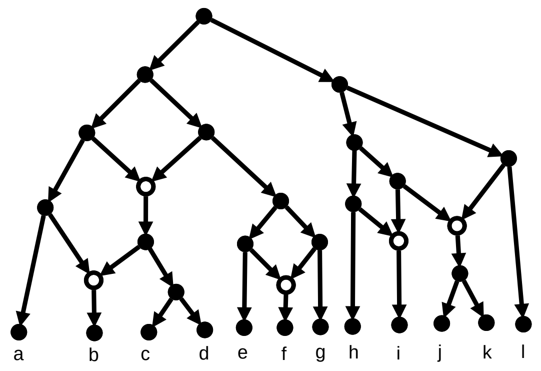

Fig. 1. An example of a (binary) rooted phylogenetic network on $\mathcal{X} = \{a, \ldots, l\}$. This network has five reticulation nodes, shown here unfilled.

1 Introduction

The rooted phylogenetic tree (henceforth, tree) is the traditional model for modelling the evolution of a set of species (or, more generally, *taxa*) $\mathcal{X}$ (see e.g. [9], [10], [24]). A rooted phylogenetic network (henceforth, network) is a generalisation from trees to directed acyclic graphs which allows reticulate evolutionary phenomena such as hybridization, recombination and horizontal gene transfer to be incorporated (see Figure 1). For detailed background information on networks we refer the reader to [12], [13], [14], [26], [22], [23].

One use of networks, motivated in particular by the need to merge a set of discordant gene trees into a species network [22], is the following. Given a set of trees $\mathcal{T}$, where each tree $T \in \mathcal{T}$ has the same set of taxa $\mathcal{X}$, construct a "most parsimonious" network which displays all the trees in $\mathcal{T}$. If we define "most parsimonious" to mean: has as few reticulation nodes (i.e. nodes with indegree two or higher) as possible, we obtain the *hybridization number* problem [2], [3]. There has been extensive research into perhaps the simplest possible variant of this problem; this is when $\mathcal{T}$ contains two binary (i.e. fully resolved) trees. Unfortunately, even this stylized version of the problem is computationally difficult; it is NP-hard and in a theoretical sense difficult to approximate well [5], [19]. On the other hand, there has been considerable progress in developing fixed parameter tractable (FPT) algorithms for the problem. Essentially, these are algorithms which can determine whether the hybridization number of two trees is at most r in time $f(r) \cdot poly(n)$, where $n = |\mathcal{X}|$ and $f(r)$ is a function that does not depend on n (see [8] for an introduction to fixed parameter tractability). The idea of such algorithms is that, by decoupling n and r, the running time of the algorithm tends to grow more slowly than algorithms with a running time of the form $O(n^{f(r)})$. The first such algorithms were described in [4], [7] and the current theoretical state-of-the art is an algorithm with running time $(3.18^r) \cdot poly(n)$ [29]. There are also a number of very fast software packages in existence that are wholly or partially based on insights from fixed parameter tractability [1], [6].

However, what if $\mathcal{T}$ contains more than two trees and/or contains trees that are not fully resolved? Algorithms to compute the hybridization number of such $\mathcal{T}$ are necessary, because this more accurately reflects the type of trees that emerge in applied phylogenetics [21]. In this article we are interested in the situation when $\mathcal{T}$ contains two not necessarily fully resolved trees

• T. Piovesan and S. Kelk are with the Deparment of Knowledge Engineering (DKE), Maastricht University, P.O. Box 616, 6200 MD Maastricht, The Netherlands. S. Kelk is corresponding author, e-mail: steven.kelk@maastrichtuniversity.nl

on $\mathcal{X}$. (We henceforth refer to such trees as *nonbinary*, noting that this classification includes binary trees as a special case). Given that this problem is a generalisation of the binary case, it inherits all the negative results from that case, but not necessarily the positive results. Indeed, there are far fewer *positive* results for nonbinary. A number of non-trivial technicalities arise because in the nonbinary case we only require that the network displays *some refinement* of each tree i.e. the image of the tree contained in the network can be more resolved than the original tree [20]. This is a natural and desirable definition given that biologists often use nodes with outdegree 3 or higher in trees to denote *uncertainty*, rather than a hard topological constraint.

Recently there have been two non-FPT algorithms implemented (both of which are available in the package DENDROSCOPE [15]) to solve the nonbinary problem in polynomial time when the hybridization number is bounded [11], [27]. The nonbinary problem is, furthermore, FPT. This was established in [20] using kernelization. Unfortunately, mainly due to the very idiosyncratic behaviour of *common chains* in the nonbinary case, the analysis given in [20] is rather long and complex, and the (weighted) kernel they describe is also rather large, containing at most $(89r)$ taxa; the size of the unweighted kernel is quadratic in r. As far as we are aware the algorithm in [20] has not been implemented.

In this article we present an alternative FPT algorithm for nonbinary trees that is based on bounded-search rather than kernelization, with running time $(6^r r!) \cdot poly(n)$. The resulting algorithm is extremely simple and amenable to implementation (it manages to completely avoid the concept of chains) and the analysis of correctness is comparatively straightforward. The algorithm builds heavily on a number of basic results from the *softwired cluster* literature [12], [13], in particular [18]. This literature concerns a slightly different methodology for constructing phylogenetic networks, but as observed in [26], [18] the optima of the models synchronise in the case of two input trees, allowing results and concepts from one methodology to be used in the other.

The simplicity of our new algorithm stems from a careful examination of a natural partial order (and its maximal elements, which we call *terminals*) on $\mathcal{X}$, which turns out to be closely linked to hybridization number. This partial order appeared earlier in [16] and [18] but was used in a slightly different way. Via the observations in [18] the earlier (and more general) results in [16] also imply an FPT algorithm via softwired clusters for the nonbinary case, but with an astronomical running time. The added value of the present article is that, by making heavy use of the fact that there are only two trees in the input, we are able to obtain a significantly simplified and optimized algorithm that can actually be used in practice.

For completeness we have implemented a prototype version of the algorithm, available upon request. However, perhaps the best use of the algorithm is to integrate it into existing, well-supported non-FPT algorithms for the nonbinary problem (such as the 2012 release of CASS [25], [27]) to bound their search space and to thus upgrade their status to FPT.

2 PRELIMINARIES

2.1 Trees, networks and clusters

Consider a set $\mathcal{X}$ of taxa. A *rooted phylogenetic network* (on $\mathcal{X}$), henceforth *network*, is a directed acyclic graph with a single node with indegree zero (the *root*), no nodes with both indegree and outdegree equal to 1, and leaves bijectively labeled by $\mathcal{X}$. The indegree of a node v is denoted $\delta^-(v)$ and v is called a *reticulation* if $\delta^-(v) \geq 2$, otherwise it is called a *tree* node. An edge (u, v) is called a *reticulation edge* if its target node v is a reticulation. When counting reticulations in a network, we count reticulations with more than two incoming edges more than once because, biologically, these reticulations represent several reticulate evolutionary events. Therefore, we formally define the *reticulation number* of a network $N = (V, E)$ as

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

A *rooted phylogenetic tree* on $\mathcal{X}$, henceforth *tree*, is simply a network that has reticulation number zero. We say that a network N on $\mathcal{X}$ *displays* a tree T if T can be obtained from N by performing a series of node and edge deletions and eventually by suppressing nodes with both indegree and outdegree equal to 1 (see Figure 2). We assume without loss of generality that each reticulation has outdegree at least one. Consequently, each leaf has indegree one. We say that a network is *binary* if every reticulation node has indegree 2 and outdegree 1 and every tree node that is not a leaf has outdegree 2.

Proper subsets of $\mathcal{X}$ are called *clusters*, and a cluster C is a *singleton* if $|C| = 1$. We say that an edge (u, v) of a tree *represents* a cluster $C \subset \mathcal{X}$ if C is the set of taxa descendants of v. A tree T represents a cluster C if it contains an edge that represents C. For example, the tree in Figure 2(a) represents $\{c, d, e\}$ but not $\{d, e, f\}$. We say that N represents C "in the softwired sense" if N displays some tree T on $\mathcal{X}$ such that T represents C. In this article we only consider the softwired notion of cluster representation and henceforth assume this implicitly. A network represents a set of clusters $\mathcal{C}$ if it represents every cluster in $\mathcal{C}$ (and possibly more). For a set $\mathcal{C}$ of clusters on $\mathcal{X}$ we define $r(\mathcal{C})$ as $\min\{r(N) | N$ represents $\mathcal{C}\}$, we refer to this as the *reticulation number* of $\mathcal{C}$. We say that two clusters $C_1, C_2 \subset \mathcal{X}$ are *compatible* if either $C_1 \cap C_2 = \emptyset$ or $C_1 \subseteq C_2$ or $C_2 \subseteq C_1$, and *incompatible* otherwise. A set of clusters $\mathcal{C}$ is compatible if all clusters in $\mathcal{C}$ are mutually compatible.

2.2 The equivalence of (maximal) common pendant subtrees and (maximal) ST-sets

Let T be a tree on $\mathcal{X}$. We write $Cl(T)$ to denote the set of clusters represented by edges of T, and for a set of

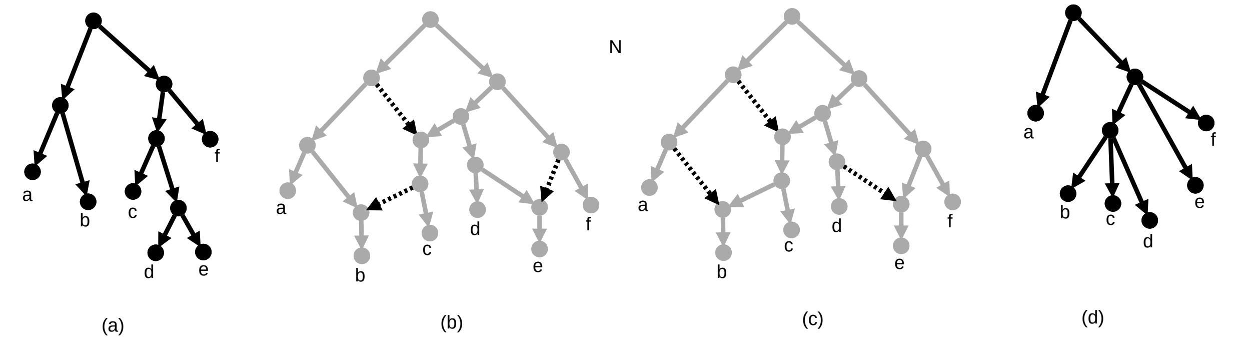

Fig. 2. In (b) we see that N displays the tree in (a), and in (c) we see that N displays a binary refinement of the tree in (d). The dotted edges denote the reticulation edges that should be deleted to obtain the required tree.

trees $\mathcal{T}$ on $\mathcal{X}$ we write $Cl(\mathcal{T}) = \cup_{T \in \mathcal{T}} Cl(T)$. We say that a (binary) tree T' on $\mathcal{X}$ is a (binary) *refinement* of T if $Cl(T) \subseteq Cl(T')$ (see Figure 2). We say two trees T_1 and T_2 on $\mathcal{X}$ have a *common refinement* if there exists a tree T' on $\mathcal{X}$ such that $Cl(T_1) \cup Cl(T_2) \subseteq Cl(T')$, where the last condition is equivalent to saying that the set of clusters $Cl(T_1) \cup Cl(T_2)$ is compatible. We say that a tree T^* on $\mathcal{X}^* \subseteq \mathcal{X}$ is a *pendant subtree* of T if there is a refinement T' of T such that $\mathcal{X}^* \in Cl(T')$. Note that this definition does not depend on the topology of T^* so we can equivalently say that $\mathcal{X}^*$ is a pendant subtree of T. A pendant subtree $\mathcal{X}^*$ is *non-trivial* if $|\mathcal{X}^*| > 1$. Given two trees T_1, T_2 on $\mathcal{X}$ we say that $\mathcal{X}^* \subseteq \mathcal{X}$ is a *common* pendant subtree if $\mathcal{X}^*$ is a pendant subtree of both T_1 and T_2 and $T_1|\mathcal{X}^*$ and $T_2|\mathcal{X}^*$ have a common refinement. (As usual $T|\mathcal{X}'$ for $\mathcal{X}' \subseteq \mathcal{X}$ refers to the tree obtained by suppressing nodes with indegree and outdegree equal to 1 in the minimal subtree of T that connects all elements of $\mathcal{X}'$). Note that our definition of common pendant subtree is consistent with [20], which we follow.

Given a set $S \subseteq \mathcal{X}$ of taxa, we use $\mathcal{C} \setminus S$ to denote the result of removing all elements of S from each cluster in $\mathcal{C}$ and we use $\mathcal{C}|S$ to denote $\mathcal{C} \setminus (\mathcal{X} \setminus S)$ (the restriction of $\mathcal{C}$ to S). Following [18], we say that a set $S \subseteq \mathcal{X}$ is an *ST-set* with respect to $\mathcal{C}$, if S is compatible with all clusters in $\mathcal{C}$ and any two clusters $C_1, C_2 \in \mathcal{C}|S$ are compatible. An ST-set S is *maximal* if there is no ST-set T with $S \subset T$. The maximal ST-sets are unique, partition $\mathcal{X}$ and can be computed in polynomial time [18].

In [18, Lemma 6] it is proven that, if $\mathcal{C} = Cl(T_1) \cup Cl(T_2)$, and $\mathcal{X}^*$ is an ST-set of $\mathcal{C}$, then for each $i \in \{1, 2\}$, there exists a node v_i of T_i such that $\mathcal{X}^*$ is exactly equal to the union of the clusters represented by some (not necessarily strict) subset of the edges outgoing from v_i. From this it follows that $\mathcal{X}^*$ is a (maximal) ST-set of $Cl(T_1) \cup Cl(T_2)$ if and only if $\mathcal{X}^*$ is a (maximal) common pendant subtree of T_1 and T_2. We will make heavy use of this equivalence and use the concepts interchangeably. In particular, all the maximal ST-sets of $\mathcal{C} = Cl(T_1) \cup Cl(T_2)$ are singletons (in which case we say $\mathcal{C}$ is *ST-collapsed* [18]) if and only if T_1 and T_2 have no non-trivial common pendant subtrees. A related operation is to create an ST-collapsed set of clusters by *collapsing* all maximal ST-sets into single taxa as shown in Figure 3. Collapsing maximal ST-sets does not change the reticulation number of the set of clusters (because there always exists an optimal network in which the maximal ST-sets are "pendant" [18, Corollary 11]).

2.3 The special case of (clusters obtained from) two trees

Given two trees $\mathcal{T} = \{T_1, T_2\}$ on $\mathcal{X}$, we (again following [20]) define $h(\mathcal{T})$ (the *hybridization number* of $\mathcal{T}$) as the smallest value of $r(N)$ ranging over all networks N on $\mathcal{X}$ such that N displays a binary refinement of T_1 and a binary refinement of T_2. In [18, Observation 9] we note that the emphasis on *binary* refinements does not sacrifice generality. Furthermore, from [26, Lemma 2] we may assume without loss of generality that in the definition of $h(\mathcal{T})$, N can be restricted to being binary. Observe that, if $\mathcal{T}$ is an arbitrary set of trees on $\mathcal{X}$, $r(Cl(\mathcal{T})) \leq h(\mathcal{T})$. This holds because if a network displays a (refinement of a) tree T then it certainly also represents all the clusters in $Cl(T)$. For $|\mathcal{T}| > 2$ this inequality can be strict [26]. However, in [18, Lemma 12] it is proven that if $\mathcal{T} = \{T_1, T_2\}$ are two trees on $\mathcal{X}$, and $\mathcal{C} = Cl(\mathcal{T})$, then $r(\mathcal{C}) = h(\mathcal{T})$. Unfortunately, even in this special case, if N represents all clusters in $\mathcal{C}$ it does not necessarily display (binary refinements of) T_1 and T_2 [26]. Fortunately a polynomial-time, reticulation-number preserving transformation is possible, which we describe later in Section 3.

3 The structure of optimal solutions

We begin with some simple results which formalize the idea that, when $\mathcal{T}$ contains exactly two trees, the problem has "optimal substructure" i.e. optimal solutions can be constructed from arbitrary optimal solutions for well-chosen subproblems. We begin with a focus on clusters, but then explicitly link this to trees in Lemma 2 and Corollary 1.

Observation 1. *Let $\mathcal{C} = Cl(\mathcal{T})$ be a set of clusters on $\mathcal{X}$, where $\mathcal{T} = \{T_1, T_2\}$ is a set of two trees on $\mathcal{X}$ with no non-trivial common pendant subtrees, and $r(\mathcal{C}) \geq 1$. Then there exists $x \in \mathcal{X}$ such that $r(\mathcal{C} \setminus \{x\}) < r(\mathcal{C})$.*

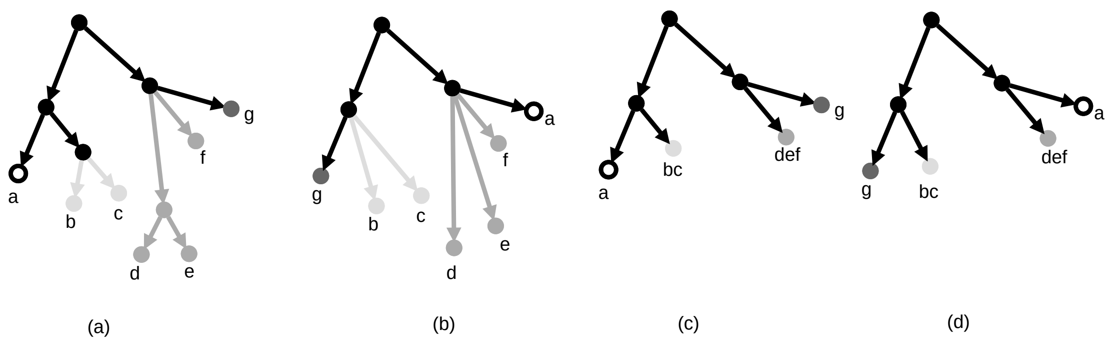

Fig. 3. The two trees shown in (a) and (b) have maximal ST-sets (i.e. maximal common pendant subtrees) $\{a\}, \{b,c\}, \{d,e,f\}, \{g\}$, shown in greyscale. In (c) and (d) we show the result of collapsing the maximal ST-sets in (a) and (b) (respectively) into single taxa.

Proof: Consider without loss of generality a binary network N which represents $\mathcal{C}$, where $r(N) = r(\mathcal{C})$. By acyclicity N contains at least one *Subtree Below a Reticulation* (SBR) [18], i.e. a node u with indegree 1 whose parent is a reticulation, and such that no reticulation can be reached by a directed path from u. Let $\mathcal{X}'$ be the set of taxa reachable from u by directed paths. $\mathcal{X}'$ is an ST-set, so $|\mathcal{X}'| = 1$ (because $\mathcal{C}$ is ST-collapsed). Let x be the single taxon in $\mathcal{X}'$. Deleting x and its reticulation parent from N (and tidying up the resulting network in the usual fashion[1]) creates a network N' on $\mathcal{X} \setminus \{x\}$ with $r(N') < r(N)$ that represents $\mathcal{C} \setminus \{x\}$. □

Lemma 1. *Let $\mathcal{C} = Cl(\mathcal{T})$ be a set of clusters on $\mathcal{X}$, where $\mathcal{T} = \{T_1, T_2\}$ is a set of two trees on $\mathcal{X}$ with no non-trivial common pendant subtrees, and $r(\mathcal{C}) \geq 1$. Then for each $x \in \mathcal{X}$ it holds that $r(\mathcal{C}) - 1 \leq r(\mathcal{C} \setminus \{x\}) \leq r(\mathcal{C})$.*

Proof: The second $\leq$ is immediate because removing a taxon from a cluster set cannot raise the reticulation number of the cluster set. The first $\leq$ holds because in [18, Lemma 10] it is shown how, given *any* network N' on $\mathcal{X} \setminus \{x\}$ that represents $\mathcal{C} \setminus \{x\}$, we can extend N' to obtain a network N on $\mathcal{X}$ that represents $\mathcal{C}$ such that $r(N) \leq r(N') + 1$. □

We recall the following definition from [18]. For a set of clusters $\mathcal{C}$ on $\mathcal{X}$, we call $(S_1, S_2, ..., S_p)$ $(p \geq 0)$ an *ST-set tree sequence of length* p if S_1 is a ST-set of $\mathcal{C}$, S_2 is a ST-set of $\mathcal{C} \setminus S_1$, S_3 is a ST-set of $\mathcal{C} \setminus S_1 \setminus S_2$ (and so on) and if all the clusters in $\mathcal{C} \setminus S_1 \setminus \ldots \setminus S_p$ are mutually compatible i.e. can be represented by a tree. If $\mathcal{C} = Cl(\mathcal{T})$ where $\mathcal{T} = \{T_1, T_2\}$ are two trees on $\mathcal{X}$, then $r(\mathcal{C})$ is exactly equal to the minimum length of an ST-set tree sequence for $\mathcal{C}$ [18, Corollary 9]. Essentially, the ST-set tree sequence describes an order in which common pendant subtrees can be iteratively pruned from T_1 and T_2 to obtain a common tree T. As an example, the two trees in Figure 3(a) and (b) have a minimum-length ST-set tree sequence $(\{b,c\}, \{d,e,f\})$, and the hybridization number of these two trees is indeed 2.

1. Specifically, for as long as necessary applying the following tidying-up operations until they are no longer needed: deleting any node with outdegree zero that is not labelled by an element of $\mathcal{X}$; suppressing all nodes with indegree and outdegree both equal to 1; replacing multi-edges with single edges; deleting nodes with indegree 0 and outdegree 1 [18].

Observation 1 and Lemma 1 show that, in an ST-collapsed cluster set, there *always* exists at least one taxon x such that $r(\mathcal{C} \setminus \{x\}) = r(\mathcal{C}) - 1$, and that this is the best possible decrease in reticulation number. If we somehow locate such an x (it does *not* matter which one), construct $\mathcal{C} \setminus \{x\}$, compute its maximal ST-sets, collapse them, and then repeat this until we obtain a compatible set of clusters, we are actually constructing a minimum-length ST-set tree sequence $(S_1, \ldots, S_{r(\mathcal{C})})$ of $\mathcal{C}$. (Note that the actual S_i can easily be obtained by reversing any collapsing operations). Such a sequence not only tells us $r(\mathcal{C})$, it also instructs us how to construct in polynomial time a network N which represents all the clusters in $\mathcal{C}$ such that $r(\mathcal{C}) = r(N)$ [18, Theorem 3]. Less obviously, it also tells us how to construct a network N with $r(N) = r(C) = h(\mathcal{T})$ which displays the two trees that $\mathcal{C}$ came from:

Lemma 2. *Let $\mathcal{C} = Cl(\mathcal{T})$ be a set of clusters on $\mathcal{X}$, where $\mathcal{T} = \{T_1, T_2\}$ is a set of two trees on $\mathcal{X}$. Let $(S_1, \ldots, S_p)$ be an ST-set tree sequence of $\mathcal{C}$. Then in polynomial time we can construct a network N that displays binary refinements of T_1 and T_2 such that $r(N) = p$.*

Proof: (Figure 4 shows a slightly stylized example of the following). Let $\mathcal{X}_0 = \mathcal{X}$ and let $\mathcal{X}_i = \mathcal{X}_{i-1} \setminus S_i$, for $1 \leq i \leq p$. Define $\mathcal{C}_i = \mathcal{C}|\mathcal{X}_i$, for $0 \leq i \leq p$. By assumption, the clusters in $\mathcal{C}_p$ can be represented by a tree. This is equivalent to saying that $T_1|\mathcal{X}_p$ and $T_2|\mathcal{X}_p$ have a common refinement. We construct in polynomial time an arbitrary binary tree T on $\mathcal{X}_p$ that displays these clusters; T will also be a common binary refinement of $T_1|\mathcal{X}_p$ and $T_2|\mathcal{X}_p$. Let $T = N_p$. We now show how to construct a network N_{i-1} that displays binary refinements of $T_1|\mathcal{X}_{i-1}$ and $T_2|\mathcal{X}_{i-1}$, given an arbitrary network N_i that displays binary refinements of $T_1|\mathcal{X}_i$ and $T_2|\mathcal{X}_i$, for $1 \leq i \leq p$. By definition, S_i is an ST-set of $\mathcal{C}_{i-1}$. S_i thus corresponds to a common pendant subtree of $T_1|\mathcal{X}_{i-1}$ and $T_2|\mathcal{X}_{i-1}$, and indeed $T_1|\mathcal{X}_i$ and $T_2|\mathcal{X}_i$ are exactly the trees obtained by pruning S_i from $T_1|\mathcal{X}_{i-1}$ and $T_2|\mathcal{X}_{i-1}$. So, reversing this pruning means that $T_1|\mathcal{X}_{i-1}$ and $T_2|\mathcal{X}_{i-1}$ can be obtained from $T_1|\mathcal{X}_i$ and $T_2|\mathcal{X}_i$ (respectively) by re-grafting S_i at a particular vertex or edge. Specifically, let T^* be an arbitrary binary tree that represents $\mathcal{C}_{i-1}|S_i$, this will also be a common binary refinement of the common pendant subtree S_i. Now, N_{i-1} can be obtained from N_i by extending the images of $T_1|\mathcal{X}_i$ and $T_2|\mathcal{X}_i$ inside N_i as follows: we introduce T^* below a new reticulation and attach this reticulation at (or, if necessary, slightly above) the two aforementioned re-grafting points. There are some small technicalities (such as the need for a "dummy root" [18]) but we omit these details. □

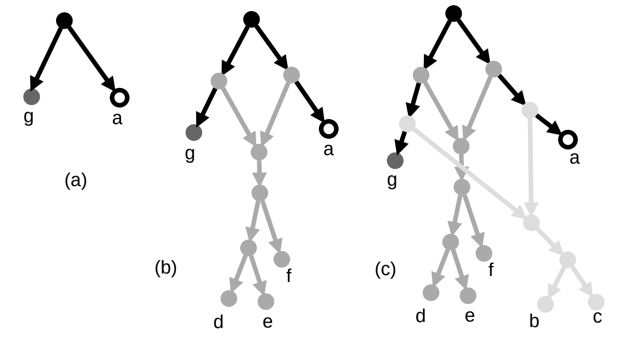

Fig. 4. A demonstration of the construction described in Lemma 2. The trees in Figure 3(a) and (b) have a minimum-length ST-set tree sequence $(\{b,c\},\{d,e,f\})$ and here we show how to construct a network N with $r(N)=2$ that displays binary refinements of both these trees, by re-introducing the elements of the ST-set tree sequence in reverse order.

Corollary 1. *Let $\mathcal{C} = Cl(\mathcal{T})$ be a set of clusters on $\mathcal{X}$, where $\mathcal{T} = \{T_1, T_2\}$ is a set of two trees on $\mathcal{X}$. Let N be a network on $\mathcal{X}$ that represents all the clusters in $\mathcal{C}$. Then in polynomial time we can construct a network N' that displays binary refinements of T_1 and T_2 such that $r(N') \leq r(N)$.*

Proof: If N is a tree we can simply take a binary refinement of N and we are done. Otherwise, N contains at least one SBR. The taxa in an SBR form an ST-set. So if we identify an SBR of N (which can easily be done in polynomial time), remove it (and tidy up in the usual fashion), and repeat this until we obtain a tree, we obtain an ST-set tree sequence of length at most $r(N)$. (It will be less than $r(N)$ if removing some SBR causes more than one reticulation to disappear from the network when tidying up). This dismantling of N is described in more detail in [18, Lemma 7]. We can then apply Lemma 2 to construct the network. □

Lemma 2 and Corollary 1 allow us for the remainder of the article to focus only on clusters.

4 Terminals

As we have seen, computing $r(\mathcal{C})$ (and an accompanying optimal network) essentially boils down to repeatedly identifying some taxon x such that $r(\mathcal{C} \setminus \{x\}) = r(\mathcal{C}) - 1$. The key to attaining fixed parameter tractability is to construct a "small" $\mathcal{X}' \subseteq \mathcal{X}$ which is guaranteed to contain at least one such taxon x. This brings us to the following concept.

Given a cluster set $\mathcal{C}$ and $x, y \in \mathcal{X}$, we write $x \to_{\mathcal{C}} y$ if and only if every non-singleton cluster in $\mathcal{C}$ containing x, also contains y[2]. We say that a taxon $x \in \mathcal{X}$ is a *terminal* if there does not exist $x' \in \mathcal{X}$ such that $x \neq x'$ and $x \to_{\mathcal{C}} x'$.

Observation 2. *Let $\mathcal{C}$ be an ST-collapsed set of clusters on $\mathcal{X}$ such that $r(\mathcal{C}) \geq 1$. Then the relation $\to_{\mathcal{C}}$ is a partial order on $\mathcal{X}$, the terminals are the maximal elements of the partial order and each non-singleton cluster of $\mathcal{C}$ contains at least one terminal.*

Proof: The relation $\to_{\mathcal{C}}$ is clearly reflexive and transitive. To see that it is anti-symmetric, suppose there exist two elements $x \neq y \in \mathcal{X}$ such that $x \to_{\mathcal{C}} y$ and $y \to_{\mathcal{C}} x$. Then we have that, for every non-singleton cluster $C \in \mathcal{C}$, $C \cap \{x,y\}$ is either equal to $\emptyset$ or $\{x,y\}$ i.e. C is compatible with $\{x,y\}$. Furthermore, the only clusters that can possibly be in $\mathcal{C}|\{x,y\}$ are $\{x\},\{y\}$ and $\{x,y\}$ and these are all mutually compatible. So $\{x,y\}$ is an ST-set, contradicting the fact that $\mathcal{C}$ is ST-collapsed. Hence $\to_{\mathcal{C}}$ is a partial order. The fact that the terminals are the maximal elements of the partial order then follows immediately from their definition. Finally, observe that a non-singleton cluster C must contain at least one terminal, because if it does not then the relation $\to_{\mathcal{C}}$ induces a cycle on some subset of C, contradicting the aforementioned anti-symmetry property. □

Let T be a phylogenetic tree on $\mathcal{X}$. For a vertex u of T we define $\mathcal{X}(u) \subseteq \mathcal{X}$ to be the set of all taxa that can be reached from u by directed paths. For a taxon $x \in \mathcal{X}$ we define $W^T(x)$, the *witness set* for x in T, as $\mathcal{X}(u) \setminus \{x\}$, where u is the parent of x. A critical property of $W^T(x)$ is that, for any non-singleton cluster $C \in Cl(T)$ that contains x, $W^T(x) \subseteq C$ [18].

Observation 3. *Let $\mathcal{C} = Cl(\mathcal{T})$ be a set of clusters on $\mathcal{X}$, where $\mathcal{T} = \{T_1, T_2\}$ is a set of two trees on $\mathcal{X}$ with no non-trivial common pendant subtrees, and $r(\mathcal{C}) \geq 1$. Then for any $x \in \mathcal{X}$ the following statements are equivalent: (1) x is a terminal of $\mathcal{C}$; (2) there exist incompatible clusters $C_1, C_2 \in \mathcal{C}$ such that $C_1 \cap C_2 = \{x\}$; (3) $W^{T_1}(x) \cap W^{T_2}(x) = \emptyset$.*

Proof: We first prove that (2) implies (1). For $x' \notin C_1 \cup C_2$ it holds that $x \not\to_{\mathcal{C}} x'$, because $x \in C_1$ but $x' \notin C_1$. For $x' \in C_1 \setminus C_2$ it cannot hold that $x \to_{\mathcal{C}} x'$, because $x \in C_2$ but $x' \notin C_2$, and this holds symmetrically for $x' \in C_2 \setminus C_1$. Hence x is a terminal. We now show that (1) implies (3). Suppose (3) does not hold. Then there exists some taxon $x' \in W^{T_1}(x) \cap W^{T_2}(x)$. So every non-singleton cluster in $\mathcal{C}$ that contains x also contains x', irrespective of whether

2. Note that, if a taxon x appears in only one cluster, $\{x\}$, then (vacuously) $x \to_{\mathcal{C}} y$ for all $y \neq x$.

the cluster came from T_1 or T_2. But then $x \rightarrow_{\mathcal{C}} x'$, so (1) does not hold. Hence (1) implies (3). Finally, we show that (3) implies (2). Note that (3) implies that in both T_1 and T_2 the parent of x is *not* the root. If this was not so, then (wlog) $W^{T_1}(x) = \mathcal{X} \setminus \{x\}$, and combining this with the fact that $W^{T_1}(x), W^{T_2}(x) \neq \emptyset$ would contradict (3). Hence $W^{T_1}(x) \cup \{x\} \in Cl(T_1)$ and $W^{T_2}(x) \cup \{x\} \in Cl(T_2)$, from which (2) follows. □

For two nodes $u \neq v$ in a network we define a *tree path from u to v* as a directed path that starts at u and ends at v such that all interior nodes of the path are tree nodes. This definition includes the possibility that u and/or v are reticulation nodes, this will be clear from the specific context. Observe that if $x \neq y$ are taxa in a network N that represents a set of clusters $\mathcal{C}$ and there is a tree path from the parent of x to y, then $x \rightarrow_{\mathcal{C}} y$. The set of nodes *reachable by a tree path from u* is the set of all $v \neq u$ such that there is a tree path from u to v.

Lemma 3. *Let $\mathcal{C}$ be an ST-collapsed set of clusters on $\mathcal{X}$ such that $r(\mathcal{C}) \geq 1$. Then $\mathcal{C}$ has at most $3 \cdot r(\mathcal{C})$ terminals.*

Proof: Let N be a network on $\mathcal{X}$ such that N represents $\mathcal{C}$ and $r(N) = r(\mathcal{C})$. Without loss of generality we can assume N is binary. For each $x \in \mathcal{X}$, exactly one of the following conditions holds: (1) the parent of x in N is a reticulation; (2) the parent of x in N is not a reticulation but there is a directed path from the parent of x in N to a reticulation. To see this observe that if neither condition holds then N contains an edge (u, v) such that at least two taxa, but no reticulations, are reachable by directed paths from v. But then $\mathcal{C}$ contains a non-singleton ST-set, contradiction. Let $R(N)$ be the reticulation nodes in N. Let $\Omega(\mathcal{C}) \subseteq \mathcal{X}$ denote the set of terminals of $\mathcal{C}$. We describe a function $F : \Omega(\mathcal{C}) \rightarrow R(N)$ such that each reticulation is mapped to at most 3 times, from which the result follows. For each terminal x for which condition (1) holds, $F(x) = p(x)$, where $p(x)$ is the parent of x. For each terminal x for which condition (2) holds, choose a reticulation r such that there is a tree path from $p(x)$ to r, and set $F(x) = r$. Note that there cannot ever be a tree path from $p(x)$ to y if $x \neq y$ are both terminals, because this would mean $x \rightarrow_{\mathcal{C}} y$. Now, it follows that a reticulation can be mapped to (in F) in at most 3 ways: from a terminal immediately below it and from one terminal per incoming edge. □

Corollary 2. *Let $\mathcal{C}$ be an ST-collapsed set of clusters on $\mathcal{X}$ such that $r(\mathcal{C}) \geq 1$. Any subset of terminals with cardinality $2 \cdot r(\mathcal{C}) + 1$ or higher, contains at least one taxon x such that $r(\mathcal{C} \setminus \{x\}) < r(\mathcal{C})$.*

Proof: From the proof of Lemma 3 we observe that in any subset of $2 \cdot r(\mathcal{C}) + 1$ terminals, there exists at least one taxon x for which condition (1) holds. Hence x is an SBR and (as argued in Observation 1) $r(\mathcal{C} \setminus \{x\}) < r(\mathcal{C})$. □

5 Main result

For a reticulation r in a network N, let $\mathcal{X}^t(r)$ be the set of all taxa that can be reached by tree paths from r. For example, if we label the reticulations in the network in Figure 2 r_1, r_2, r_3, from left to right, $\mathcal{X}^t(r_1) = \{b\}, \mathcal{X}^t(r_2) = \{c\}$ and $\mathcal{X}^t(r_3) = \{e\}$. The following lemma shows that an optimal network cannot contain a reticulation r such that $\mathcal{X}^t(r) = \emptyset$.

Lemma 4. *Let $\mathcal{C} = Cl(\mathcal{T})$ be a set of clusters on $\mathcal{X}$, where $\mathcal{T} = \{T_1, T_2\}$ is a set of two trees on $\mathcal{X}$ with no non-trivial common pendant subtrees, and $r(\mathcal{C}) \geq 1$. Let N be a network on $\mathcal{X}$ that represents $\mathcal{C}$ and let r be a reticulation of N such that $\mathcal{X}^t(r) = \emptyset$. Then $r(\mathcal{C}) < r(N)$.*

Proof: Let $R^t(r)$ be the set of reticulations in N reachable by tree paths from r. Now, consider the technique described in the proof of Corollary 1 for dismantling N by removing one SBR at a time. All reticulations in $R^t(r)$ will be pruned away at an iteration that is earlier than or equal to the iteration in which r is pruned away. Moreover, due to the fact that $X^t(r) = \emptyset$ - that is, there are no taxa "sandwiched" between r and $R^t(r)$ - there definitely exists $r' \in R^t(r)$ such that r' and r both vanish in the same iteration. But this means that the technique produces an ST-set tree sequence of length strictly less than $r(N)$, which (by Lemma 2, or [18, Theorem 3]) implies the existence of a network N' that represents $\mathcal{C}$ such that $r(N') < r(N)$. □

Corollary 3. *Let $\mathcal{C} = Cl(\mathcal{T})$ be a set of clusters on $\mathcal{X}$, where $\mathcal{T} = \{T_1, T_2\}$ is a set of two trees on $\mathcal{X}$ with no non-trivial common pendant subtrees, and $r(\mathcal{C}) \geq 1$. Let N be a network on $\mathcal{X}$ that represents $\mathcal{C}$ such that $r(N) = r(\mathcal{C})$ and let r be a reticulation of N such that $\mathcal{X}^t(r) = \{x\}$ for some $x \in \mathcal{X}$. Then $r(\mathcal{C} \setminus \{x\}) = r(\mathcal{C}) - 1$.*

Proof: If x is an SBR the result is immediate. Otherwise, if x is deleted from N, then a network N' is obtained such that N' represents $\mathcal{C} \setminus \{x\}$ and, in N', $\mathcal{X}^t(r) = \emptyset$. By Lemma 4, $r(\mathcal{C} \setminus \{x\}) < r(N')$. The result follows because $r(N') = r(N) = r(\mathcal{C})$. □

For a network N, we say that a *switching* of N is obtained by, for each reticulation node, deleting all but one of its incoming edges. The grey subtrees in Figure 2 are switchings. A network N on $\mathcal{X}$ displays a tree T on $\mathcal{X}$ if and only if there is a switching T_N of N such that T can be obtained from T_N by suppressing nodes with indegree and outdegree equal to one (and if necessary deleting nodes with indegree 0 and outdegree 1). Hence, each switching is the "image" in N of some tree displayed by N. Indeed, the following definitions are entirely consistent with the definition of cluster representation given in Section 2. Given a network N and a switching T_N of N, we say that an edge (u, v) of N represents a cluster C w.r.t. T_N if (u, v) is an edge of T_N and C is the set of taxa descendants of v in T_N. It is natural to define that an edge (u, v) of N represents a cluster C if there exists some switching T_N of N such that (u,v) represents C w.r.t T_N.

We say that a cluster $C \in \mathcal{C}$ is *minimal* if it is a non-singleton cluster such that there does not exist a non-singleton cluster $C' \in \mathcal{C}$ with $C' \subset C$.

Lemma 5. *Let $\mathcal{C} = Cl(\mathcal{T})$ be a set of clusters on $\mathcal{X}$, where $\mathcal{T} = \{T_1, T_2\}$ is a set of two trees on $\mathcal{X}$ with no non-trivial common pendant subtrees, and $r(\mathcal{C}) \geq 1$. There exists a minimal cluster $C \in \mathcal{C}$ such that, for at least $|C|-1$ of the taxa x in C, $r(\mathcal{C} \setminus \{x\}) = r(\mathcal{C}) - 1$.*

Proof: Let N be a binary network that represents $\mathcal{C}$ such that $r(\mathcal{C}) = r(N)$. Let $e = (u,v)$ be an edge of N that represents some non-singleton cluster of $\mathcal{C}$ such that there does not exist another edge $e^* = (u^*, v^*)$ reachable from e with this property (where reachable here means: there is a directed path from v to u^*). Hence e is a "lowest" edge that represents a non-singleton cluster. Let $C \in \mathcal{C}$ be a non-singleton cluster represented by e. We will prove that at least $|C|-1$ taxa x in C have the property $r(\mathcal{C} \setminus \{x\}) = r(\mathcal{C}) - 1$. Observe that this property will then automatically also hold for all non-singleton clusters $C' \subset C$, in particular minimal C', from which the claim will follow.

By definition $e = (u,v)$ is an edge of some switching T_N of N such that C is equal to the set of taxa descendants of v in T_N. Fix any such T_N. Observe firstly that if there is a directed path in T_N from v to some reticulation r, then $\mathcal{X}^t(r) \subseteq C$. The next statement is critical. Suppose there is a tree node v' which is reachable in T_N by a directed path from v. Suppose furthermore that, in T_N, the set of all taxa $\mathcal{X}'$ reachable from v' by tree paths (in T_N) has cardinality exactly 2. We show that this situation cannot actually happen. To see this, let $\{y,z\}$ be the taxa in $\mathcal{X}'$. By assumption $\{y,z\}$ is not an ST-set, because $\mathcal{C}$ is ST-collapsed. Hence there must exist a non-singleton cluster $C^* \in \mathcal{C}$ such that without loss of generality $C^* \cap \{y,z\} = \{y\}$. Now, C^* must be represented by some edge $e'' = (u'', v'')$ of N. Moreover, e'' must lie somewhere on the tree path from v' to y in T_N. However, u'' is then reachable by a directed path from v, contradicting our claim that e was "lowest". So such an $\mathcal{X}'$ does not exist. Now, suppose that r is a reticulation in T_N such that (1) r can be reached in T_N by a directed path from v, (2) two or more taxa can be reached in T_N from r by tree paths. Due to the fact that N is binary, there must exist a tree node v' reachable in T_N by a tree path from r, such that $\{x,y\}$ are the only two taxa reachable from v' by tree paths in T_N. We have already concluded, however, that this is not possible. Hence we can infer that, if r is a reticulation in T_N such that r can be reached by a directed path from v, $|\mathcal{X}^t(r)| = 1$. This, in turn, means that with one possible exception (because there can be at most one taxon in C reachable in T_N from v by a tree path) each taxon $x \in C$ is such that $\mathcal{X}^t(r) = \{x\}$ for some r i.e. x is either an SBR or is the unique taxon "sandwiched" between several reticulations. By Corollary 3 we are done. □

An immediate consequence of Lemma 5 is that if we could identify minimal cluster C, it would be sufficient to restrict our attention to an arbitrary size-2 subset of it: we could still be sure that at least one of the the taxa x is such that $r(\mathcal{C} \setminus \{x\}) = r(\mathcal{C}) - 1$. This is the motivation behind the following theorem.

Theorem 1. *Let $\mathcal{C} = Cl(\mathcal{T})$ be a set of clusters on $\mathcal{X}$, where $\mathcal{T} = \{T_1, T_2\}$ is a set of two trees on $\mathcal{X}$ with no non-trivial common pendant subtrees, and $r(\mathcal{C}) \geq 1$. Let $\mathcal{X}' \subseteq \mathcal{X}$ be the set constructed as follows. If there are strictly more than $2 \cdot r(\mathcal{C})$ terminals in $\mathcal{C}$, let $\mathcal{X}'$ be an arbitrary subset of the terminals of cardinality $2 \cdot r(\mathcal{C}) + 1$. Otherwise, for each minimal cluster $C \in \mathcal{C}$, put two arbitrary taxa from C in $\mathcal{X}'$, of which at least one is a terminal. Then $|\mathcal{X}'| \leq 6 \cdot r(\mathcal{C})$ and there exists $x \in \mathcal{X}'$ such that $r(\mathcal{C} \setminus \{x\}) = r(\mathcal{C}) - 1$.*

Proof: The first way of constructing $\mathcal{X}'$ is correct by Corollary 2. Let us then assume that there are at most $2 \cdot r(\mathcal{C})$ terminals. Recall that each (minimal) cluster contains at least one terminal, by Observation 2. A terminal can appear in at most one minimal cluster from T_1, and at most one minimal cluster from T_2. Consider the following mapping from $\mathcal{X}'$ to itself. Map each terminal to itself. For each non-terminal $y \in \mathcal{X}'$, map y (arbitrarily) to a terminal $x \in \mathcal{X}'$ such that x and y are both in some minimal cluster of $\mathcal{C}$. In this mapping, a terminal can be mapped onto at most 3 times (i.e. from itself and at most two non-terminals). Hence $|\mathcal{X}'| \leq 6 \cdot r(\mathcal{C})$. □

6 The algorithm

We have described the algorithm non-determinstically (see Algorithm 1) to keep the exposition as clear as possible. The correctness of the algorithm is primarily a consequence of Lemma 5 and Corollary 2. If we let $r = r(\mathcal{C})$, the running time is at most $(6^r r!) \cdot r \cdot poly(n)$ where $n = |\mathcal{X}|$. The single r term comes from line 2. The $(6^r r!)$ term is a consequence of Theorem 1; $|\mathcal{X}'|$ never rises above $6r$, and each iteration of the main loop is assumed to reduce the reticulation number by 1, giving a running time of at most $(6r)(6(r-1))(6(r-2))\ldots = 6^r r!$. The $poly(n)$ term includes operations such as computing terminals, locating minimal clusters and collapsing maximal ST-sets; the first two operations are clearly polynomial-time because $\mathcal{C}(\mathcal{T}) \leq 4(n-1)$ (which follows from the fact that a tree on n taxa contains at most $2(n-1)$ edges). In fact, the most time-consuming operation inside the $poly(n)$ term is collapsing maximal ST-sets (i.e. maximal common pendant subtrees). In [18, Lemma 5] a naive $O(n^4)$ algorithm is given for this although with intelligent use of data structures and exploiting the fact that $\mathcal{C}$ comes from two trees $O(n^2)$ is certainly possible without too much effort. Finally, we note that the single r term can be absorbed, if necessary, into the $poly(n)$ term to give $(6^r r!) \cdot poly(n)$, because (trivially) $r \leq n$.

Algorithm 1

```
1: Input: Two trees $\mathcal{T} = \{T_1, T_2\}$ on the same set of taxa $\mathcal{X}$.
2: Output: A network $N$ that displays binary refinements of $T_1$ and $T_2$ such that $r(N) = h(\mathcal{T})$.
3: set $\mathcal{C} := Cl(\mathcal{T})$
4: guess $r = h(\mathcal{T}) = r(\mathcal{C})$
5: for $i := r$ downto $1$ do
6:     collapse all maximal ST-sets (i.e. maximal common pendant subtrees) in $\mathcal{C}$ to obtain a set of clusters $\mathcal{C}'$
7:     if $\mathcal{C}'$ contains more than $2i$ terminals then
8:         set $\mathcal{X}'$ to be an arbitrary size $2i+1$ subset of the terminals
9:     else
10:        construct $\mathcal{X}'$ by taking two taxa from each minimal cluster of $\mathcal{C}'$, such that at least one of each pair is a terminal
11:    end if
12:    guess an element $x \in \mathcal{X}'$ such that $r(\mathcal{C}' \setminus \{x\}) = r(\mathcal{C}') - 1$ and record that $x_{r-i+1} := x$
13:    set $\mathcal{C} := \mathcal{C}' \setminus \{x\}$
14: end for
15: convert the sequence $(x_1, \dots, x_r)$ into the ST-set tree sequence $\mathcal{S} = (S_1, \dots, S_r)$ of $\mathcal{C}$ by decollapsing taxa
16: use $\mathcal{S}$ to construct a binary network $N$ with $r(N) = h(\mathcal{T})$ that displays binary refinements of $T_1$ and $T_2$ (see Lemma 2).
```

7 Future work

Computing the hybridization number of more than two trees remains a challenging problem. In [28] a kernelization-based FPT algorithm is given that works for any number of binary trees, and [17] describes a bounded-search FPT algorithm that works for "well-bounded" sets of nonbinary trees. Both types of FPT algorithm face the problem that for three or more trees it no longer seems sufficient to guess *any* taxon whose removal lowers the hybridization number; the topology of the optimal network becomes far more important. This remains the major obstacle to obtaining efficient algorithms for three or more trees: the only explicit attempt to address this so far is the brute-force enumeration of network topologies described in [17].

8 Acknowledgements

We gratefully acknowledge Jean Derks and Nela Lekic for their helpful comments concerning an earlier version of this article. We also thank Simone Linz and Leo van Iersel for useful discussions.

References

[1] B. Albrecht, C. Scornavacca, A. Cenci, and D.H. Huson. Fast computation of minimum hybridization networks. *Bioinformatics*, 28(2):191–197, 2012.

[2] M. Baroni, S. Grünewald, V. Moulton, and C. Semple. Bounding the number of hybridisation events for a consistent evolutionary history. *Journal of Mathematical Biology*, 51:171–182, 2005.

[3] M. Baroni, C. Semple, and M. Steel. A framework for representing reticulate evolution. *Annals of Combinatorics*, 8:391–408, 2004.

[4] M. Bordewich, S. Linz, K. St. John, and C. Semple. A reduction algorithm for computing the hybridization number of two trees. *Evolutionary Bioinformatics*, 3:86–98, 2007.

[5] M. Bordewich and C. Semple. Computing the minimum number of hybridization events for a consistent evolutionary history. *Discrete Applied Mathematics*, 155(8):914–928, 2007.

[6] Z-Z. Chen and L. Wang. Hybridnet: a tool for constructing hybridization networks. *Bioinformatics*, 26(22):2912–2913, 2010.

[7] J. Collins, S. Linz, and C. Semple. Quantifying hybridization in realistic time. *Journal of Computational Biology*, 18:1305–1318, 2011.

[8] J. Flum and M. Grohe. *Parameterized Complexity Theory*. Springer, 2006.

[9] O. Gascuel, editor. *Mathematics of Evolution and Phylogeny*. Oxford University Press, Inc., 2005.

[10] O. Gascuel and M. Steel, editors. *Reconstructing Evolution: New Mathematical and Computational Advances*. Oxford University Press, USA, 2007.

[11] D. H. Huson and S. Linz. Computing minimum hybridization networks from real phylogenetic trees. Submitted, 2012.

[12] D. H. Huson, R. Rupp, V. Berry, P. Gambette, and C. Paul. Computing galled networks from real data. *Bioinformatics*, 25(12):i85–i93, 2009.

[13] D. H. Huson, R. Rupp, and C. Scornavacca. *Phylogenetic Networks: Concepts, Algorithms and Applications*. Cambridge University Press, 2010.

[14] D. H. Huson and C Scornavacca. A survey of combinatorial methods for phylogenetic networks. *Genome Biology and Evolution*, 3:23–35, 2011.

[15] D.H. Huson and C. Scornavacca. Dendroscope 3: An interactive tool for rooted phylogenetic trees and networks. *Systematic Biology*, 2012.

[16] S. M. Kelk and C. Scornavacca. Constructing minimal phylogenetic networks from softwired clusters is fixed parameter tractable, 2011. Submitted, prelimary version available at http://arxiv.org/abs/1108.3653.

[17] S. M. Kelk and C. Scornavacca. Towards the fixed parameter tractability of constructing minimal phylogenetic networks from arbitrary sets of nonbinary trees, 2012. Submitted, preliminary version http://arxiv.org/abs/1207.7034.

[18] S. M. Kelk, C. Scornavacca, and L. J. J. van Iersel. On the elusiveness of clusters. *IEEE/ACM Trans. Comput. Biology Bioinform.*, 9(2):517–534, 2012.

[19] S. M. Kelk, L. J. J. van Iersel, S. Linz, N. Lekic, C. Scornavacca, and L. Stougie. Cycle killer... qu'est-ce que c'est? on the comparative approximability of hybridization number and directed feedback vertex set, 2012. To appear in *SIAM Journal on Discrete Mathematics*.

[20] S. Linz and C. Semple. Hybridization in non-binary trees. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 6(1):30–45, 2009.

[21] D. A. Morrison. *An introduction to phylogenetic networks*. RJR Productions, 2011. Available from http://www.rjr-productions.org/Networks/.

[22] L. Nakhleh. *The Problem Solving Handbook for Computational Biology and Bioinformatics*, chapter Evolutionary phylogenetic networks: models and issues. Springer, 2009.

[23] C. Semple. *Reconstructing Evolution - New Mathematical and Computational Advances*, chapter Hybridization Networks. Oxford University Press, 2007.

[24] C. Semple and M. Steel. *Phylogenetics*. Oxford University Press, 2003.

[25] L. J. J. van Iersel. New version of phylogenetic network software, 2012. Webpage, accessed 25th July 2012: http://phylonetworks.blogspot.nl/2012/06/new-version-of-phylogenetic-networks.html.

[26] L. J. J. van Iersel and S. M. Kelk. When two trees go to war. *Journal of Theoretical Biology*, 269(1):245–255, 2011.

[27] L. J. J. van Iersel, S. M. Kelk, R. Rupp, and D. H. Huson. Phylogenetic networks do not need to be complex: Using fewer reticulations to represent conflicting clusters. *Bioinformatics*, 26:i124–i131, 2010. Special issue: Proceedings of Intelligent Systems for

Molecular Biology 2010 (ISMB2010), 10th-13th September 2010, Boston USA.
[28] L. J. J. van Iersel and S. Linz. A quadratic kernel for computing the hybrization number of multiple trees, 2012. Submitted, preliminary version http://arxiv.org/abs/1203.4067.
[29] C. Whidden, R. G. Beiko, and N. Zeh. Fixed-parameter and approximation algorithms for maximum agreement forests. Submitted, preliminary version arXiv:1108.2664v1 [q-bio.PE].

Teresa Piovesan Teresa Piovesan received her master degree in 2012 from Maastricht University in The Netherlands. She is now working as a PhD student at the Centrum voor Wiskunde en Informatica (CWI) in Amsterdam, The Netherlands. Her research is focused on exploring the links between quantum information theory and classical computation using optimization tools.

Steven Kelk Steven M. Kelk is currently assistant professor at the Department of Knowledge Engineering (DKE) at Maastricht University in The Netherlands, where he works on applications of combinatorial optimization and discrete mathematics to computational biology. His main research interests are approximation algorithms, phylogenetic networks, graph theory and fixed parameter tractability.