Journal of Graph Algorithms and Applications
http://jgaa.info/ vol. 23, no. 4, pp. 715–743 (2019)
DOI: 10.7155/jgaa.00508

Treewidth of display graphs: bounds, brambles and applications

Remie Janssen[1] *Mark Jones*[1] *Steven Kelk*[2]
Georgios Stamoulis[2] *Taoyang Wu*[3]

[1]Delft Institute for Applied Mathematics,
Delft University of Technology, Netherlands.
[2]Department of Data Science and Knowledge Engineering,
Maastricht University, Netherlands.
[3]School of Computing Sciences,
University of East Anglia, United Kingdom.

Abstract

Phylogenetic trees and networks are leaf-labelled graphs used to model evolution. Display graphs are created by identifying common leaf labels in two or more phylogenetic trees or networks. The treewidth of such graphs is bounded as a function of many common dissimilarity measures between phylogenetic trees and this has been leveraged in fixed parameter tractability results. Here we further elucidate the properties of display graphs and their interaction with treewidth. We show that it is **NP**-hard to recognize display graphs, but that display graphs of bounded treewidth can be recognized in linear time. Next we show that if a phylogenetic network displays (i.e. topologically embeds) a phylogenetic tree, the treewidth of their display graph is bounded by a function of the treewidth of the original network (and also by various other parameters). In fact, using a bramble argument we show that this treewidth bound is sharp up to an additive term of 1. We leverage this bound to give an FPT algorithm, parameterized by treewidth, for determining whether a network displays a tree, which is an intensively-studied problem in the field. We conclude with a discussion on the future use of display graphs and treewidth in phylogenetics.

Submitted:	Reviewed:	Revised:	Accepted:
September 2018	April 2019	May 2019	August 2019
	Final:	Published:	
	September 2019	September 2019	

Article type: Regular paper
Communicated by: P. Mutzel

E-mail addresses: remiejanssen@gmail.com (Remie Janssen) markelliotlloyd@gmail.com (Mark Jones) steven.kelk@maastrichtuniversity.nl (Steven Kelk) georgios.stamoulis@maastrichtuniversity.nl (Georgios Stamoulis) Taoyang.Wu@uea.ac.uk (Taoyang Wu)

1 Introduction

A phylogenetic tree on a set of species (or, more abstractly, *taxa*) X is a tree whose leaves are bijectively labelled by X. The central idea of such structures is that internal nodes represent hypothetical ancestors of X [38]. In this way, the tree can be viewed as a summary of how X evolved over time. Here we focus on unrooted, binary trees: internal nodes all have degree 3, and there is no direction on the edges of the tree. This is not an onerous restriction, since many phylogenetic inference methods construct unrooted, binary trees. We refer the reader to [41, 18] for further background on phylogenetics.

In this article we study *display graphs*. Simply put, a display graph is obtained from two or more phylogenetic trees by identifying leaves with the same label [12, 42, 34]. Display graphs have attracted interest in recent years because of the phenomenon that, if two or more phylogenetic trees are (in some formal sense) "similar", the *treewidth* of their display graph is bounded by a function of various parameters. For example, by the number of trees that form the display graph [12], or by the Tree Bisection and Reconnect (TBR) distance of two trees [34, 1].

Treewidth is a well-known graph parameter which measures, at least in an algorithmic sense, how far an undirected graph is from being a tree: many **NP**-hard problems can be solved in polynomial or even linear time on graphs of bounded treewidth [5, 8, 9]. Display graphs thus form a bridge from phylogenetics into algorithmic graph theory. In particular, the bounds on the treewidth of display graphs have been exploited to give fixed parameter tractable algorithms for a number of **NP**-hard dissimilarity measures on phylogenetic trees [12, 34, 3, 19]. (See [15] for background on fixed parameter tractability). Display graphs have also turned out to be useful for speeding up the computation of certain "easy" parameters on phylogenetic trees [16], and the treewidth of the display graph itself has also been considered as a proxy for phylogenetic dissimilarity [33, 24].

The purpose of this article is to further investigate, and algorithmically exploit, properties of the display graphs formed not only by trees, but also by trees and *networks*. To the best of our knowledge this is the first time tree-network display graphs have been considered. In the first part of the article, we list some basic properties of display graphs, and then address the problem of *recognizing* them, a problem posed in [33]. Specifically: given a cubic graph G, do there exist two unrooted binary phylogenetic trees T_1, T_2 on the same set of taxa X such that G is the display graph $D(T_1, T_2)$ of T_1 and T_2 (after suppression of degree-2 nodes)? We prove that the problem is **NP**-hard, by providing an equivalence with the **NP**-hard TreeArboricity problem [13]. On the positive side, we prove that if G has bounded treewidth then this question can be answered in linear time. For this purpose we use Courcelle's Theorem [14, 2]. This well-known meta-theorem states, essentially, that graph properties which can be expressed as a bounded-length fragment of Monadic Second Order Logic (MSOL) can be solved in linear time on graphs of bounded treewidth. We provide such an expression for recognizing display graphs.

In the second, longer part of the article, we turn our attention to display graphs formed by merging an unrooted binary phylogenetic tree T with an unrooted binary phylogenetic *network* N, both on the same set of taxa X. The latter is simply an undirected graph where internal nodes have degree 3 and leaves, as usual, are bijectively labelled by X. Unlike trees, networks do not need to be acyclic. We emphasize that unrooted phylogenetic networks (as defined here and in e.g. [23, 44, 21, 40]) should be viewed as undirected analogues of rooted phylogenetic networks, which correspond to directed graphs [29]. This is to distinguish them from *split* networks which are phylogenetic data-visualisation tools and which have a very different phylogenetic interpretation; these are sometimes also referred to as "unrooted" networks [36].

Display graphs involving networks are relevant because of the growing number of optimization problems, traditionally posed on rooted trees and networks, which are now being mapped to the unrooted setting (see e.g. [31, 44, 27, 21]). We prove that, if N *displays* T - i.e., N contains a topological embedding of T - the treewidth of their display graph is at most $2tw(N) + 1$, where $tw(N)$ is the treewidth of the network N. We also give alternative upper bounds for the treewidth of the display graph of N and T expressed in terms of a parameter more familiar to the phylogenetics community. Specifically, we give (tight) bounds in terms of the *level* of the original network N [23] (which automatically implies bounds in terms of the weaker parameter *reticulation number)*. Briefly, the level of a network N is simply the maximum, ranging over all biconnected components of N, of the number of edges in the biconnected component minus the number of edges that a spanning tree for that component has. Following [34] we use these upper bounds to give a compact MSOL-based fixed-parameter tractable algorithm for the **NP**-hard problem of determining whether an unrooted network N displays T, under various parameterizations. This problem, particularly in the rooted setting, continues to attract significant interest in the phylogenetics literature (see [26, 44, 45] for relevant references). The parameterization in terms of treewidth is potentially interesting since, as we point out, the treewidth of N can be significantly lower than the level or reticulation number of N.

The question arises whether the bound $2tw(N)+1$ can be strengthened. We show that, up to the additive $+1$ term, this bound is essentially sharp. We do this by providing an infinite family of networks N with corresponding trees T such that T is displayed by N and whereby the treewidth of the display graph is at least twice the treewidth of N. To derive the lower bound on treewidth we crucially use *brambles* [39].

In the final part of the article we reflect on the potential future use of display graphs and treewidth in phylogenetics, and list a number of open problems.

2 Preliminaries

An *unrooted binary phylogenetic tree* T on a set of leaf labels (known as *taxa*) X is an undirected tree where all internal vertices have degree three and the

leaves are bijectively labeled by X. When it is understood from the context we will often drop the prefix "unrooted binary phylogenetic" for brevity. Similarly, an *unrooted binary phylogenetic network* N on a set of leaf labels X is a simple, connected, undirected graph that has $|X|$ degree-1 vertices that are bijectively labeled by X and any other vertex has degree 3. See Figure 1 for a simple example of a tree T and a network N.

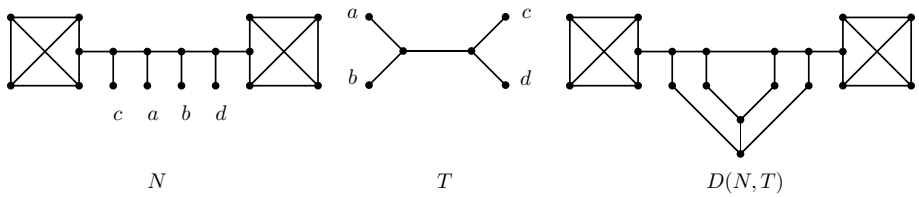

Figure 1: Left: An unrooted binary phylogenetic network N and an unrooted binary phylogenetic tree T, both on the same set of taxa $\{a, b, c, d\}$. Right: the display graph $D(N, T)$ formed from N and T. As observed later in Section 4.1, the network N does not display the tree T but the treewidth of their display graph is equal to the treewidth of N, which is equal to 3. (Note also that, if in T the positions of b and c are swapped, then N does display T but both the network and the new display graph will still have treewidth 3.)

The *reticulation number* $r(N)$ of a network $N = (V, E)$ is defined as $r(N) := |E| - (|V| - 1)$, i.e., the number of edges we need to delete from N in order to obtain a tree that spans V. A network N with $r(N) = 0$ is simply an unrooted phylogenetic tree. Note that in graph theory the value $|E| - (|V| - 1)$ of a connected graph is sometimes called the *cyclomatic number* of the graph [17].

For a given network N we define its *level*, denoted $\ell(N)$, as the minimum reticulation number ranging over all biconnected components of N. To be consistent with the phylogenetics literature we say that N is a "level-k network" if $\ell(N) \leq k$ (which means that they are "almost k-trees" [7]). A level-0 phylogenetic network is simply a phylogenetic tree. Many **NP**-hard problems in phylogenetics that involve phylogenetic networks as input or output can be solved in polynomial time if the network has bounded level (or bounded reticulation number) [32, 20, 10].

We now formally define the main object of study in this article, namely the *display graph*:

Definition 1 *Let $T_1 = (V_1 \cup X, E_1), T_2 = (V_2 \cup X, E_2)$ be two trees, both on the same set of leaf labels X. The* display graph *of T_1, T_2, denoted by $D(T_1, T_2)$, is formed by* identifying *vertices with the same leaf label and forming the disjoint union of these two trees, i.e., $D(T_1, T_2) = (V_1 \cup V_2 \cup X, E_1 \cup E_2)$.*

Although the more general definition of display graph encountered in the literature allows the display graph to be formed by more than two trees, not necessarily on the same set of taxa (see e.g. [12]), here we will focus exclusively

on the above, more restricted definition which is enough for our purposes. We note that, by construction, a display graph is always biconnected.

Note that a display graph is a labeled graph: the set X bijectively labels the degree-2 nodes in the graph. In some parts of the article the labels X and the degree-2 vertices are not important (because, modulo some trivial exceptions, degree-2 vertices do not impact upon the treewidth of a graph), and in such cases we work with *suppressed* display graphs. Such a graph is obtained by erasing the labels X and repeatedly *suppressing* degree-2 nodes. Here suppressing (also known as dissolving) a degree-2 vertex v means introducing a new edge between the two neighbours of v, and deleting v and its two incident edges. A suppressed display graph is always cubic (when $|X| \geq 3$). The act of suppressing degree-2 nodes can potentially create multi-edges. It is easy to see that this happens if and only if the two trees contain one or more common *cherries*. A cherry is a size-2 subset of taxa $\{x, y\}$ that have a common parent, and a cherry is common on two trees if it exists in both of them.

The definition of a display graph formed by a tree T and a network N, both on X, is completely analogous to the definition for two trees, and is denoted as $D(N, T)$.

Let N be a phylogenetic network and T a phylogenetic tree, both on a common taxon set X. Then we say that N *displays* T (or T is displayed by N) if there exists a subtree N' of N that is a *subdivision* of T, that is, T can be obtained by a series of *edge contractions* on N'. Here the contraction of an edge $\{u, v\}$ means deleting the edge and identifying u and v. We say that N' is an *image* of T. We observe that every vertex of T is mapped to a vertex of N', and that edges of T map to paths in N' (perhaps consisting of only a single edge) leading us to the following observation (see also [12]):

Observation 1 *If an unrooted binary phylogenetic network N displays an unrooted binary phylogenetic tree T, both on the same set of leaf labels X, then there exists a subtree N' of N and a surjective function f from $V(N')$ to $V(T)$ such that:*

(1) $f(\ell) = \ell, \forall \ell \in X$,

(2) the subsets of $V(N')$ induced by $f^{-1}(v)$, where $v \in V(T)$, are mutually disjoint, and each such subset induces a connected subtree of $V(N')$,

(3) For each edge $\{u, v\}$ in T, there exists a unique edge $\{\alpha, \beta\}$ in N' with $f(\alpha) = u$ and $f(\beta) = v$.

This observation will be crucial when we study the treewidth of $D(N, T)$ as a function of several parameters (including the treewidth) of N. We note that, since the leaves of N, T are bijectively labeled by X, we can write with a slight abuse of notation $\ell \in X$ to also refer to the unique leaf vertex with label ℓ, and this is what we do in (1) above.

We now move on to define the concept of the *treewidth* of an undirected graph:

Definition 2 *Given an undirected graph $G = (V, E)$, a* tree decomposition *of G is a pair $(\mathcal{B}, \mathcal{T})$ where $\mathcal{B} = \{B_1, \ldots, B_q\}$ is a multiset of* bags *such that $B_i \subseteq V(G)$ and $\mathcal{T}$ is a tree whose q nodes are in bijection with $\mathcal{B}$ (and, as such, we can assume that the q nodes of $\mathcal{T}$ are exactly the q bags of $\mathcal{B}$), satisfying the following three properties:*

(tw1) $\cup_{i=1}^{q} B_i = V(G)$*;*

(tw2) $\forall e = \{u, v\} \in E(G), \exists B_i \in \mathcal{B}$ *s.t.* $\{u, v\} \subseteq B_i$*;*

(tw3) running intersection property: $\forall v \in V(G)$ *all the bags B_i that contain v form a connected subtree of $\mathcal{T}$.*

The width *of $(\mathcal{B}, \mathcal{T})$ is equal to* $\max_{i=1}^{q} |B_i| - 1$*. The* treewidth *of G, denoted by $tw(G)$, is the smallest width among all possible tree decompositions of G. A tree decomposition achieving the smallest possible width for a given graph G is called optimal. For brevity, we often simply write $\mathcal{T}$ as shorthand for a tree decomposition.*

If an undirected graph H can be obtained from a graph G by deleting vertices and edges and contracting edges, then H is a *minor* of G. It is well known that, if H is a *minor* of a graph G, then $tw(H) \leq tw(G)$ [17].

In [33] it was shown that the treewidth of the display graph of two trees can be, in the worst case, linear in the number of the vertices in the trees. In this article we will explore the relation of the treewidth of a display graph formed by a phylogenetic network and a tree displayed by that network, and the treewidth (or other parameters) of the network itself.

Finally, we define the *bramble* parameter of a graph, a parameter closely related to treewidth that is very useful when proving lower bounds on treewidth. Given a graph G and two subgraphs S_1, S_2 of it, we say that S_1 and S_2 *touch* if $V(S_1) \cap V(S_2) \neq \emptyset$, *or* some edge of G has one endpoint in S_1 and the other in S_2. A *bramble* B of G is a set of connected subgraphs of G that pairwise touch. A (sub)set $H \subseteq V(G)$ is a *hitting set* of a bramble B of G if H intersects every element of B. The *order* of B is the minimum size of such a hitting set and the bramble number of G, denoted by $br(G)$, is the maximum order of a bramble of G, among all possible brambles. The usefulness of brambles comes from the following result, due to Seymour & Thomas, relating the treewidth of a graph G to its bramble number:

Theorem 1 ([39]) *For any graph G we have that $tw(G) = br(G) - 1$.*

3 Recognizing display graphs of pairs of trees

We consider the DisplayGraph decision problem, posed in [33]:

Input: A biconnected, cubic, simple graph $G = (V, E)$.

Goal: Find two unrooted binary trees T_1, T_2, on the same set of taxa X, such that the suppressed display graph $D(T_1, T_2)$ of these two trees is isomorphic to G, if they exist.

Note that in this formulation we can assume without any loss of generality that T_1 and T_2 do not have common cherries.

Here we will argue that the DisplayGraph problem is **NP**-hard by providing an equivalence between the DisplayGraph problem and the **NP**-hard TreeArboricity problem [13] which is defined as follows:

Input: A simple, undirected graph $G = (V, E)$.
Goal: Find the smallest positive integer k such that there exists a partition $(V_1, \ldots, V_k)$ of V such that each part of the partition induces a tree, i.e., the subgraph induced by the vertices in V_i, denoted by $G|_{V_i}$ is a tree for $i \in [k]$ (such a partition is called a *tree partition*). This k is the *Tree Arboricity* of G, also denoted as $ta(G)$.

We emphasize that unlike some closely related variants of the problem (for example VertexArboricity [37]), it is not permitted that a $G|_{V_i}$ induces a forest consisting of two or more components.

Chang et al. [13] discuss the decision version of the TreeArboricity problem with $k = 2$ (i.e., is $ta(G) \leq 2$?) and show that it is **NP**-complete when restricted to simple, cubic, 3-connected planar graphs. The following lemma binds their problem to ours.

Lemma 1 *Given a simple, connected, cubic graph G as input to the* TreeArboricity *decision problem, G is a "yes" instance for the* TreeArboricity *problem with $k = 2$ if and only if G is a suppressed display graph $D(T_1, T_2)$ of two binary phylogenetic trees T_1, T_2 on a common set of taxa X.*

Proof: Given such T_1, T_2 then the partition of the set of vertices into two sets V_1, V_2 is simply $V_i = V(T_i) \setminus X$. We exclude the taxa X since, when we form the display graph $D(T_1, T_2)$, these will become degree-2 vertices which are subsequently suppressed. On the other hand, given a bipartition V_1, V_2 of G, we can form the two phylogenetic trees T_1, T_2 on a common set of taxa X whose display graph is isomorphic to G as follows. First of all, by definition, $G|_{V_1}, G|_{V_2}$ are trees. Since G is connected and cubic, every leaf vertex v in one bipartition, say $G|_{V_1}$, has exactly 2 neighbor vertices u_1, u_2 in $G|_{V_2}$ (i.e., $\{u_1, u_2\} \subseteq V_2$). Subdivide each of the edges $\{v, u_1\}, \{v, u_2\}$ with a new vertex in X (i.e., for $i = 1, 2$, replace edge $\{v, u_i\}$ with the two edges $\{v, w_i\}, \{w_i, u_i\}$, where each w_i is a newly introduced vertex, and include $w_i \in X$ which is initially empty). The points of subdivisions of these "crossing" edges (having one vertex in each bipartition) are the taxa X of the new trees. Repeat the process on the remaining leaf vertices from $G|_{V_2}$. The same argumentation will also take care of the remaining degree-2 vertices in each of $G|_{V_1}$ and $G|_{V_2}$. To complete the proof, we need to show that the number of the degree-1 plus the degree-2 vertices in $G|_{V_1}, G|_{V_2}$ are equal, such that the two constructed trees are binary phylogenetic trees. Indeed, this will follow because G is cubic and

connected and a "yes" instance to the TreeArboricity problem. Specifically, each edge not entirely in $G|_{V_i}$ must have one endpoint in each bipartition. Thus, if we define for every vertex $v \in V_i$ its "missing" degree in each tree as $\mu(v) = 3 - \deg(v)$ (where here $\deg(v)$ refers to the degree of v in $G|_{V_i}$), then we see that $\sum_{v \in V_1} \mu(v) = \sum_{u \in V_2} \mu(u)$ i.e., both constructed trees T_1, T_2 are binary and, by construction, on the same set of taxa X. □

Theorem 2 DisplayGraph *is* ***NP****-complete.*

Proof: The DisplayGraph problem is easily seen to be in **NP**: a certificate can be the two trees T_1, T_2 that form the graph G. We only need to check that $D(T_1, T_2)$, after suppressing degree-2 vertices, is isomorphic to G, something that can be done in polynomial time since the graph isomorphism problem is polynomial-time solvable for graphs of bounded degree [35, 25]. For hardness, Chang et al. [13] prove that the decision version of the TreeArboricity problem with $k = 2$ is **NP**-complete when restricted to simple, cubic, 3-connected planar graphs. Thus, let G be a simple, cubic, 3-connected planar graph that is input to the TreeArboricity problem. A 3-connected graph is vacuously also a biconnected graph, so G is a valid input to the DisplayGraph problem. The result follows because of the if and only if relationship described in Lemma 1. □

3.1 The fixed parameter tractability of recognizing display graphs of bounded treewidth

Let $G = (V, E)$ be a simple, biconnected cubic graph. We will use Courcelle's Theorem to test whether G is a suppressed display graph. This will show that the question can be settled in time $O(f(tw(G)) \cdot |V|)$ where f is a function that depends only on the treewidth of G. Specifically, when G has bounded treewidth this will yield a linear time algorithm. The constant-length MSOL formulation simply tests whether $ta(G) \leq 2$. (Clearly, $ta(G) \geq 2$ because G is not acyclic). The MSOL formulation (and an introduction to MSOL proofs) is given in the appendix.

Theorem 3 *Suppressed display graphs can be recognized in linear time on graphs of bounded treewidth.*

Proof: This is a consequence of the correctness of the MSOL formulation described in Appendix A.2 and the equivalence stated in Lemma 1. □

4 Display graphs formed from trees and networks

In this section we will consider the display graph formed by an unrooted binary phylogenetic network $N = (V, E)$ and an unrooted binary phylogenetic tree T

both on the same set of taxa X. We will show upper and lower bounds on the treewidth of $D(N,T)$ in terms of the treewidth $tw(N)$ of N and the level $\ell(N)$ of N (and thus also the reticulation number $r(N)$ of N). We will also show how these upper bounds can be leveraged algorithmically to give FPT results for deciding whether a given network N displays a given tree T.

4.1 Treewidth upper bounds

We first relate the treewidth of the display graph with the treewidth of the network N.

Lemma 2 *Let $N = (V,E)$ be an unrooted binary phylogenetic network and T an unrooted binary phylogenetic tree, both on X, where $|X| \geq 3$. If N displays T, then $tw(D(N,T)) \leq 2tw(N) + 1$.*

Proof: Since N displays T, we fix a subgraph N' of N that is a subdivision of T and a surjective function f from $V(N')$ to $V(T)$ as defined in Observation 1 (in Section 2). Informally, f maps taxa to taxa, and degree-3 vertices of N' to the corresponding vertex of T. Vertices of degree-2 in N' form paths between degree-3 vertices, and between degree-3 vertices and taxa. Concerning f, each edge $\{u,v\}$ of T corresponds to a path in N' (whose interior vertices, if they exist, have degree 2) whereby part of the path lies inside $f^{-1}(u)$, part lies in $f^{-1}(v)$, and exactly one edge of the path has an endpoint in $f^{-1}(u)$ and an endpoint in $f^{-1}(v)$.

Now, consider any tree decomposition $\mathcal{T}$ of N. Let k be the width of the tree decomposition, i.e., the largest bag in the tree decomposition has size $k+1$. We will construct a tree decomposition $\mathcal{T}'$ for $D(N,T)$ as follows. For each vertex $u' \in V(N')$ we add $f(u')$ to every bag that contains u'. To show that $\mathcal{T}'$ is a valid tree decomposition for $D(N,T)$ we will show that it satisfies all the treewidth conditions. Condition (tw1) holds because f is a surjection.

To show that (tw2) holds for $\mathcal{T}'$, we fix an arbitrary edge $\{u,v\}$ in $E(T)$. Then it suffices to show that there exists some bag in $\mathcal{T}'$ which contains both u and v. By the third property of f as described in Observation 1, there exists a unique edge $\{\alpha,\beta\}$ in $E(N')$ with $f(\alpha) = u$ and $f(\beta) = v$. Noting that $\{\alpha,\beta\}$ is also an edge in $E(N)$, there exists a bag B in $\mathcal{T}$ with $\{\alpha,\beta\} \subseteq B$. Since $f(\alpha) = u$ and $f(\beta) = v$, both u and v will be added into B to form a bag in $\mathcal{T}'$ that contains both u and v, as required.

For the last property (tw3) we need to show that the bags of $\mathcal{T}'$ to which $v \in V(T)$ has been added form a connected component. For this, we use the second property of f as described in Observation 1: $\forall v \in V(T)$, the set $\{u \in V(N') : f(u) = v\}$ forms a connected subtree in N'. Hence, the set of bags that contain at least one element from $\{u \in V(N') : f(u) = v\}$ form a connected subtree in the tree decomposition. These are the bags to which v is added, ensuring that (tw3) indeed holds for v.

We now calculate the width of $\mathcal{T}'$: Observe that the size of each bag can at most double. This can happen when every vertex in the bag is in $V(N')$ and

$f(u') \neq f(v')$ for every two vertices u', v' in the bag. This causes the largest bag after this operation to have size at most $2(k+1)$. That is, the width of the new decomposition is at most $2k+1$. □

We move on and deliver a bound of the treewidth of the display graph $D(N,T)$ in terms of the level $\ell(N)$ of N. We remind the reader that a network N is a level-k network if the reticulation number of each biconnected component is at most k.

Lemma 3 *Let $N = (V,E)$ be an unrooted binary phylogenetic network and T an unrooted binary phylogenetic tree, both on X, such that $|X| \geq 3$ and N displays T. Then $tw(D(N,T)) \leq \ell(N) + 2$ where $\ell(N)$ is the level of N.*

Proof: Due to the fact that N displays T, there is a subgraph T' of N that is a subdivision of T. If T' is a spanning tree of N, then keep T' as is. Otherwise, construct a spanning tree T' of N by greedily adding edges to T' until all vertices of N are spanned. At this point, T' contains exactly $|V|-1$ edges and consists of a subdivision of T from which possibly some unlabelled pendant subtrees (i.e. pendant subtrees without taxa) are hanging.

We argue that $D(T',T)$ has treewidth 2, as follows. First, note that $D(T,T)$ can be obtained from $D(T',T)$ by repeatedly deleting unlabelled vertices of degree 1 and suppressing unlabelled degree 2 vertices. Since these operations cannot increase or decrease the treewidth [33], $D(T',T)$ has the same treewidth as that of $D(T,T)$. On the other hand, $D(T,T)$ has treewidth 2 because T is trivally compatible with T (and $|X| \geq 3$) [12]. Hence $D(T',T)$ has treewidth 2.

For the purposes of the present proof we need a tree decomposition of $D(T,T')$ of width 2 with a very particular structure which we now construct explicitly. For each vertex $a' \in V(T')$ we create a singleton bag $\{a'\}$. For each edge $\{a',b'\} \in E(T')$ we insert the bag $\{a',b'\}$ between the two singleton bags $\{a'\}$ and $\{b'\}$. Now, recall that each vertex $a \in V(T)$ has a unique image $a' \in V(T')$. For each vertex $a \in V(T)$, add a to the singleton bag $\{a'\}$. For each edge $\{a,b\} \in E(T)$, consider the vertices a' and b' in T'. We distinguish two cases:

Case 1. If $\{a',b'\} \in E(T')$, remove the bag $\{a',b'\}$ that lies between bags $\{a,a'\}$ and $\{b,b'\}$ and replace it with the pair of bags $\{a,a',b\},\{a',b',b\}$.

Case 2. If $\{a',b'\} \notin E(T')$, then edge $\{a,b\} \in V(T)$ corresponds to a path $a', v_1, \ldots, v_t, b'$ in T' where $t \geq 1$ and none of $v_1, \ldots, v_t$ are images of vertices from T. In the tree decomposition, this corresponds to the chain of bags $\{a,a'\},\{a',v_1\},\{v_1\},\{v_1,v_2\},\{v_2\},\ldots,\{v_t,b'\},\{b,b'\}$. In this case, we add a to the bag $\{a',v_1\}$, add both a and b to bag $\{v_1\}$, and add just b to all the remaining bags in the chain.

We denote the tree decomposition by $\mathcal{T}$. It is immediate to verify, by construction, that the above tree decomposition is indeed a valid tree decomposition, i.e., it satisfies all the three properties (tw1)-(tw3).

Crucially, the topology of $\mathcal{T}$ is a subdivision of T': each vertex $a' \in V(T')$ corresponds to a unique bag of $\mathcal{T}$, and each edge in $E(T')$ corresponds to a unique chain of bags in $\mathcal{T}$. We leverage this property as follows.

Let C be 2-connected component of N. Let $k = \ell(N)$. Then we have that $|E(C)| - (|V(C)| - 1) \leq k$. Combined with the fact that T' is a spanning tree of N, it follows that we can obtain N from T' by adding at most k missing edges to C (and repeating this for other non-trivial biconnected components). Let $M(C)$ be the at most k edges missing from C in T' and let $A(C)$ be a (not necessarily minimum) minimal vertex cover of the edges in $M(C)$; clearly $|A(C)| \leq k$ since in the worst case we can select one distinct vertex per edge. Due to the topological structure of $\mathcal{T}$ (i.e., that $\mathcal{T}$ is a subdivision of T') the vertices and edges of C in T' map unambiguously into bags and chains of bags in $\mathcal{T}$. We add all the vertices in $A(C)$ to all these bags. This will take care of all the edges $\{u, v\}$ of C that are not in T'. We repeat this for each non-trivial biconnected component of N. Due to the fact that N has maximum degree 3, the non-trivial biconnected components of N are vertex-disjoint, and hence the corresponding bags in $\mathcal{T}$ are all disjoint. This means that, after all the non-trivial biconnected components have been processed, each bag will contain at most $k + 3$ vertices.

It remains to show that this is indeed a valid tree decomposition for $D(N, T)$. The vertex set of $D(N, T)$ is the same as that of $D(T, T')$ so (tw1) is clearly satisfied. For each edge $\{x, y\} \in M(C)$, both x and y are inside C, so some bag (in the part of $\mathcal{T}$ corresponding to C) contained x and some bag contained y. Given that $A(C) \cap \{x, y\} \neq \emptyset$, adding all the vertices in $A(C)$ to all the bags (corresponding to C) ensures that some bag contains both x and y. Hence, (tw2) is satisfied. Regarding (tw3), observe that each vertex $x \in A(C)$ lies inside C, so in $\mathcal{T}$ some bag (in the part of the decomposition corresponding to C) already contained x. Moreover, all the bags corresponding to C induce a connected subtree of bags. Hence, adding x to all these bags cannot destroy the running intersection property for x. Hence, (tw3) holds. □

The following observation helps to contextualize Lemmas 2 and 3.

Observation 2 *Let N be an unrooted binary phylogenetic network. Then $tw(N) - 1 \leq \ell(N) \leq r(N)$.*

Proof: $\ell(N) \leq r(N)$ follows by definition. To see that $tw(N) - 1 \leq \ell(N)$, it is well-known that the treewidth of a graph is equal to the maximum treewidth ranging over all biconnected components in the graph [7]. A spanning tree for each biconnected component can be obtained by deleting at most $\ell(N)$ edges, by definition. A tree has treewidth 1, and adding one edge to a graph can increase its treewidth by at most 1 [7]. Hence, each biconnected component has treewidth at most $1+\ell(N)$. (Alternatively, by observing that level-k networks are almost k-trees, [7, Theorem 74] can be leveraged). □

The following corollary is therefore immediate.

Corollary 1 *Let $N = (V, E)$ be an unrooted binary phylogenetic network and T an unrooted binary phylogenetic tree, both on X, where $|X| \geq 3$. If N displays T then $tw(D(N, T)) \leq r(N) + 2$.*

Combining the above results yields the following:

Theorem 4 *Let N be an unrooted binary phylogenetic network and T be an unrooted binary phylogenetic tree, both on X. Then if N displays T,*

$$tw(D(N, T)) \leq \min \left\{ 2tw(N) + 1, r(N) + 2, \ell(N) + 2 \right\}.$$

Here the term $r(N) + 2$ in the last theorem is included for completeness as $\ell(N) + 2 \leq r(N) + 2$ always holds in view of Observation 2. Note that, from the perspective of $r(N)$ and $\ell(N)$, the bounds $\ell(N) + 2$ and $r(N) + 2$ are sharp, since if $N = T$ then $r(N) = \ell(N) = 0$ and $D(N, T)$ has treewidth 2 [12]. Curiously, the treewidth bound gives 3 for this same instance: an additive error of 1. In Section 4.3 we will further analyse the sharpness of this bound.

We remark that $tw(N)$ can be arbitrarily small compared to $\ell(N)$ (and $r(N)$). For example, the display graph of two copies of the same tree T on n taxa has treewidth 2. Re-introducing taxa to turn the degree-2 vertices into degree-3 vertices, we obtain a biconnected treewidth 2 phylogenetic network $N = (V, E)$ with $3n - 4$ vertices and $5n - 6$ edges, so $\ell(N) = r(N) = |E| - (|V| - 1) \to \infty$ as $n \to \infty$. However, for N with low $\ell(N)$ the bound $\ell(N) + 2$ will potentially be stronger than $2tw(N) + 1$.

The above bounds raise a number interesting points about the phylogenetic interpretation of treewidth. First, consider the case where a binary network N *does not* display a given binary phylogenetic network T. As we can see in Figure 1, there is a network N and a tree T such that N does not display T and yet the treewidth of their display graph is equal to the treewidth of N which (as can be easily verified) is equal to three. Hence "does not display" does not necessarily cause an increase in the treewidth. On the other hand, the results from [33] show that for two incompatible unrooted binary phylogenetic trees (vacuously: neither of which displays the other, and both of which have treewidth 1) the treewidth of the display graph can be as large as linear in the size of the trees. The increase in treewidth in this situation is asymptotically maximal. So the relationship between "does not display" and treewidth is rather complex. Contrast this with the bounded growth in treewidth articulated in Theorem 4. Such bounded growth opens the door to algorithmic applications.

4.2 An algorithmic application

We give an example of how the upper bounds from the previous section can be leveraged algorithmically. The Unrooted Tree Containment problem (UTC) is simply the **NP**-hard problem of determining whether an unrooted binary phylogenetic network $N = (V, E)$ on X displays an unrooted binary phylogenetic tree T, also on X. In [44], a linear kernel (parameterized by reticulation number)

is described for the UTC problem, and, separately, a bounded-search branching algorithm. Summarizing, these yield FPT algorithms parameterized by $r(N) = |E| - (|V| - 1)$, i.e., algorithms that can solve UTC in time at most $f(r(N)) \cdot \text{poly}(|N| + |T|)$ for some function f that depends only on $r(N)$. We emphasize that these results are more involved than the trivial $2^{r(N)} \cdot \text{poly}(|N| + |T|)$ FPT algorithm for the *rooted* version of the problem.

Here we give an FPT proof using Courcelle's Theorem. We prove that the problem is FPT when parameterized by $tw(N)$. This result has not appeared in the literature before and is potentially interesting given that $tw(N)$ can be much smaller than $\ell(N)$. FPT in terms of $r(N)$ and $\ell(N)$ follow as a corollary of this, due to Observation 2.

Theorem 5 *Given an unrooted binary phylogenetic network $N = (V, E)$ and an unrooted binary phylogenetic tree both on X, we can determine in time $O(f(t) \cdot n)$ whether N displays T, where t is $tw(N)$ and $n = |V|$.*

Proof: We run Bodlaender's linear-time FPT algorithm [6] to compute a tree decomposition of $D(N, T)$ and return NO if the treewidth is larger than $2t + 1$[1]. This is correct by Lemma 2. Otherwise, we have a bound on the treewidth of $D(N, T)$ in terms of t. Subsequently, we construct the constant-length MSOL sentence described in Appendix A.1 and apply the Arnborg et al. [2] variant of Courcelle's Theorem [14]. This variant of Courcelle's Theorem permits (amongst other extensions) subsets of vertices and edges to be labelled; we use this to make X and other relevant subsets available to the logical formula. From this, the result follows. (Note that $D(N, T)$ has $O(n)$ vertices and $O(n)$ edges). The result can be made constructive if desired, i.e., in the event of a YES answer the actual set of edge cuts in N (to obtain an image of T) can be obtained. □

Corollary 2 *Given an unrooted binary network $N = (V, E)$ and an unrooted binary tree both on X, we can determine in time $O(f(k) \cdot n)$ whether N displays T, where $k = \ell(N)$ and $n = |V|$.*

Proof: Immediate from Theorem 5 and Observation 2. □

4.3 Treewidth lower bounds

In this subsection, we show that the upper bound $tw(D(N, T)) \leq 2tw(N) + 1$ is almost optimal, in the sense that there exist a family of display graphs $D(N, T)$ such that N displays T and $tw(D(N, T)) \geq 2tw(N)$. (Note that, irrespective of whether N displays T, $tw(D(N, T)) \geq tw(N)$ always holds because N is a minor of $D(N, T)$; see Figure 1 for examples when $tw(D(N, T)) = tw(N)$.)

Fix some integer r and an integer n such that $n > 2r + 2$. We will give a construction for a network N and tree T on a set of rn leaves, such that $tw(N) = r$, $tw(D(N, T)) \geq 2r$, and N displays T. For the sake of convenience, we will assume that r is even, though the construction can easily be modified

[1] The same algorithm can be used to first compute t, if it is not known.

to handle cases where r is odd. For the sake of simplicity, and as it has no affect on the treewidth of $D(N,T)$, in this subsection we take $D(N,T)$ to be the suppressed display graph.

The intuition behind the construction is as follows. The network N will have roughly the same structure as an $r \times (n+1)$ grid (with r rows and $n+1$ columns) with leaves attached to the horizontal edges. An $r \times (n+1)$ grid has treewidth $\min(r, n+1) = r$, and so N also has treewidth r. The tree T is a long caterpillar that weaves back and forth across the rows of the grid (see Figure 4). Thus T is displayed by N. However, the display graph $D(N,T)$ has (very roughly) the structure of a $2r \times (n+1)$ grid, and as such can be shown to have treewidth at least $2r$. We remind that a caterpillar graph is basically a tree where all degree-1 vertices are on distance 1 from a central path.

We now proceed with the formal construction.

Vertices of N and taxa: Let the taxon set $X = \{x_{i,j} : i \in [r], j \in [n]\}$. For each $i \in [r], j \in [n]$, N will contain a leaf labelled with $x_{i,j}$. The internal vertices of N are $y_{i,j}$ for each $i \in [r], j \in [n]$, and $u_{i,j}, v_{i,j}$ for each $i \in [r], j \in [n] \cup \{0\}$. (Note that some of these vertices will be deleted or suppressed at the end of the construction, in order to turn N into a phylogenetic network with no unlabelled leaves.)

Edges: The edges of N are as follows. For each $i \in [r], j \in [n]$, let $\{y_{i,j}, x_{i,j}\}$ be an edge in N. In addition let $\{u_{i,j-1}, v_{i,j-1}\}$, $\{v_{i,j-1}, y_{i,j}\}$, $\{y_{i,j}, u_{i,j}\}$, $\{u_{i,j}, v_{i,j}\}$ be "horizontal" edges in N. For each $i \in [r-1], j \in [n] \cup \{0\}$, let $\{v_{i,j}, u_{i+1,j}\}$ be a "vertical" edge in N. Figure 2 shows an example when $r = 4, n = 11$. Note that at this point N is not a network, as it has degree-2 vertices and unlabelled degree-1 vertices

Finally, we delete all unlabeled degree-1 vertices (namely $u_{1,0}$ and $v_{r,n}$), and then suppress all degree-2 vertices (namely $u_{i,0}$ and $v_{i,n}$ for all $i \in [r]$, as well as $u_{1,j}$ and $v_{r,j}$ for all $j \in [n] \cup \{0\}$, and the vertices $v_{1,0}$ and $u_{r,n}$). Note that this causes $v_{i,0}$ to be adjacent to $v_{i+1,0}$ for $2 \leq i \leq r-2$, and also $u_{i,n}$ to be adjacent to $u_{i+1,n}$ for $2 \leq i \leq r-2$. Figure 4 shows the final structure of N in the case when $r = 4, n = 11$, together with the tree T described below.

The tree T: We next construct the tree T as follows. For each $i \in [r], j \in [n]$, T will contain a leaf labelled with $x_{i,j}$. The internal vertices of T are $z_{i,j}$ for each $i \in [r], j \in [n]$. For each $i \in [r], j \in [n]$, there is an edge $\{z_{i,j}, x_{i,j}\}$. For each $i \in [r]$ and $j \in [n-1]$ there is an edge $\{z_{i,j}, z_{i,j+1}\}$. Furthermore, for odd $i \in [r-1]$ there is an edge $\{z_{i,n}, z_{i+1,n}\}$, and for even $i \in [r-1]$ there is an edge $\{z_{i,1}, z_{i+1,1}\}$. Finally, suppress the degree-2 vertices $z_{1,1}$ and $z_{r,1}$ (or $z_{1,1}$ and $z_{r,n}$ when r is odd). See Figure 3 for an example when $r = 4, n = 11$.

Lemma 4 *T is displayed by N.*

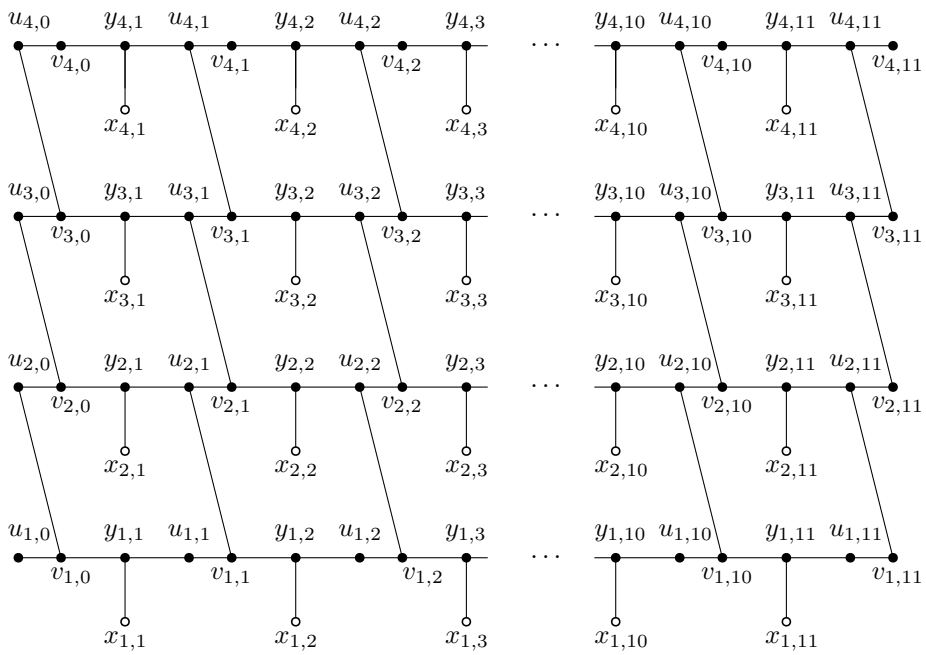

Figure 2: The construction of N before deleting unlabelled degree-1 vertices and suppressing degree-2 vertices, in the case $r = 4$ and $n = 11$.

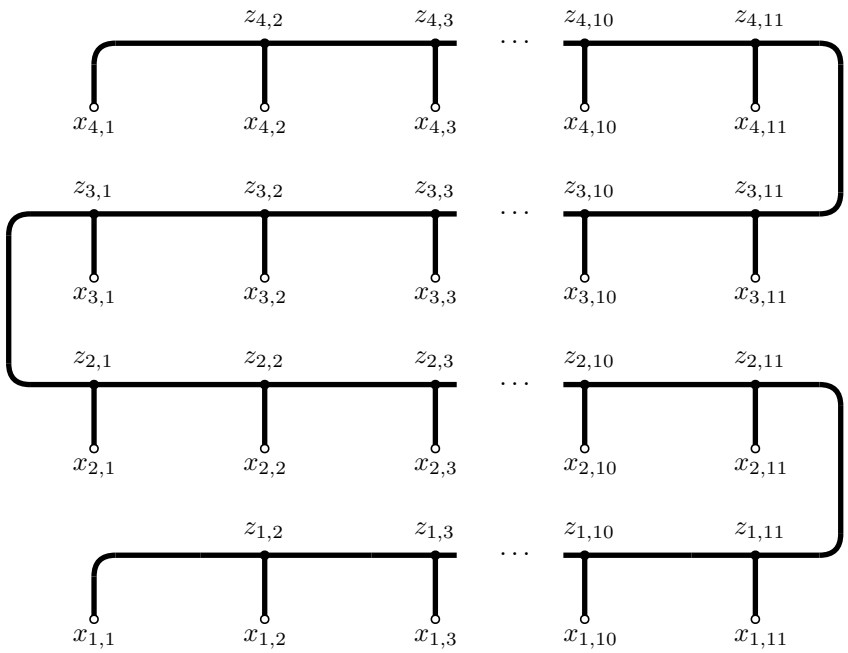

Figure 3: The tree T when $r = 4$ and $n = 11$.

Proof: Let N' be the network derived from N by deleting edges of the form $\{v_{i,j}, u_{i+1,j}\}$, as well as edges of the form $\{u_{i,n}, u_{i+1,n}\}$ for i even and $\{v_{i,0}, v_{i+1,0}\}$ for i odd, and the edges $\{x_{1,1}, v_{2,0}\}$, $\{v_{r-1,0}, y_{r,1}\}$. Observe that N' is a subtree of N, and that furthermore N' is a subdivision of T, which can be seen by mapping internal vertices $z_{i,j}$ of T to $y_{i,j}$. See Figure 4. □

This completes the construction of N and T. The (suppressed) display graph $D(N, T)$ is shown in Figure 5. For convenience, we keep the same names for internal vertices of N and T but it will always be clear from the context which structure we are referring to. Observe that any vertex $z_{i,j}$ is adjacent to $y_{i,j}$ in $D(N, T)$, as the degree-2 vertex $x_{i,j}$ has been suppressed.

Lemma 5 *The treewidth of N, $tw(N)$, is equal to r.*

Proof: To prove that $tw(N) \leq r$, we give a tree decomposition of N. We first ignore the nodes $x_{i,j}$ because those can be added to any tree decomposition of the remaining graph by adding the bags $\{x_{i,j}, y_{i,j}\}$ and connecting them to any bag containing $y_{i,j}$ for all i, j.

We will now give a tree decomposition (in fact a path decomposition[2]) of the remaining graph.

[2]A path-decomposition is a tree decomposition in which the underlying tree of the decomposition is a path graph.

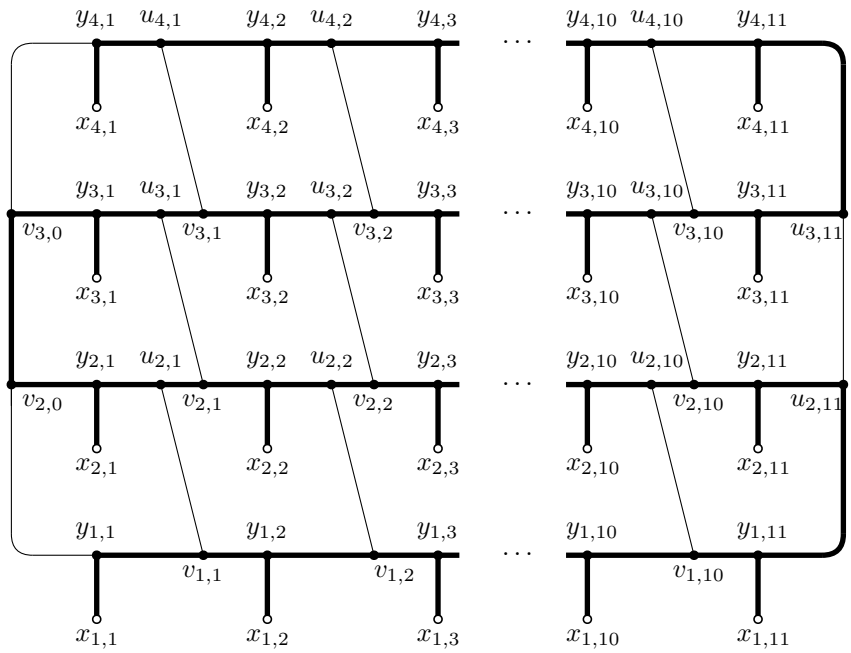

Figure 4: The network N for $r = 4, n = 11$, with the tree T drawn in bold.

Start with the bag

$$\{y_{1,1}, v_{2,0}, \ldots, v_{r-1,0}, y_{r,1}\},$$

which contains exactly r nodes. We now sequentially add one node and delete another to get a sequence of bags corresponding to a path decomposition of the remaining graph. Denote by (a, d) the process of taking the most recently created bag, B_q, adding a new bag $B_{q+1} = B_q \cup \{a\}$, and then adding a second new bag $B_{q+2} = B_{q+1} \setminus \{d\}$, with each new bag being adjacent to the one before it. Note that adding node a results in a bag with $r + 1$ nodes while deleting node d results in another bag with r nodes. Then the following steps bring us to the bag $\{v_{i,1}\}_{i\in[r-1]} \cup \{u_{r,1}\}$:

$$(v_{1,1}, y_{1,1}), (y_{2,1}, v_{2,0}), (u_{2,1}, y_{2,1}), (v_{2,1}, u_{2,1}), (y_{3,1}, v_{3,0}), \ldots, (v_{r-1,1}, u_{r-1,1}), (u_{r,1}, y_{r,1}).$$

Now we use a similar sequence of steps to go from the bag $\{v_{i,j}\}_{i\in[r-1]} \cup \{u_{r,j}\}$ to the next $\{v_{i,j+1}\}_{i\in[r-1]} \cup \{u_{r,j+1}\}$:

$$\begin{aligned}
&(y_{1,j+1}, v_{1,j}), (v_{1,j+1}, y_{1,j+1}), (y_{2,j+1}, v_{2,j}), && (u_{2,j+1}, y_{2,j+1}), (v_{2,j+1}, u_{2,j+1}),\\
&\qquad (y_{3,j+1}, v_{3,j}), && (u_{3,j+1}, y_{3,j+1}), (v_{3,j+1}, u_{3,j+1}),\\
&&& \ldots\\
&\qquad (y_{r-1,j+1}, v_{r-1,j}), && (u_{r-1,j+1}, y_{r-1,j+1}), (v_{r-1,j+1}, u_{r-1,j+1}),\\
&(y_{r,j+1}, u_{r,j}), (u_{r,j+1}, y_{r,j+1}).
\end{aligned}$$

Finally, do the following sequence of additions and deletions to the bags starting from $\{v_{i,n-1}\}_{i\in[r-1]} \cup \{u_{r,n-1}\}$:

$$(y_{1,n}, v_{1,n-1}), (y_{2,n}, v_{2,n-1}), (u_{2,n}, y_{2,n}), (y_{3,n}, v_{3,n-1}), \cdots, (u_{r-1,n}, y_{r-1,n}), (y_{r,n}, u_{r,n-1}).$$

Hence we get a path decomposition of N minus the nodes $x_{i,j}$ and their incoming edges. This can be seen by inspecting when nodes are added and deleted. Nodes in the initial bag only get deleted, nodes in the final bag only get added, and all other nodes are first added then deleted, therefore we have the running intersection property. It is also clear that each node is in at least one bag, so we still have to check that each edge is represented in a bag. We consider each type of edge separately, and find a bag where the edge is represented.

- The edges $\{y_{1,1}, v_{2,0}\}$, $\{v_{2,0}, v_{3,0}\}, \ldots, \{v_{r-2,0}, v_{r-1,0}\}$ and $\{v_{r-1,0}, y_{r,1}\}$ are in the initial bag;
- The edges $\{v_{i,0}, y_{i,1}\}$ for $i \in \{2, \cdots, r-1\}$ are in the intermediate bag for the addition/deletion $(y_{i,1}, v_{i,0})$ in the first part of the sequence;
- $\{u_{i,j}, v_{i,j}\}$ for each $i \in \{2, \cdots, r-1\}$ and $j \in [n-1]$ is in the intermediate bag for the addition/deletion $(v_{i,j}, u_{i,j})$;
- $\{v_{i,j}, y_{i,j+1}\}$ for each $i \in [r-1]$ and $j \in [n-1]$ is in the intermediate bag for the addition/deletion $(y_{i,j+1}, v_{i,j})$;
- $\{y_{i,j}, u_{i,j}\}$ for each $i \in \{2, \cdots, r\}$ and $j \in [n-1]$ is in the intermediate bag for the addition/deletion $(u_{i,j}, y_{i,j})$;
- $\{y_{1,j}, v_{1,j}\}$ for each $j \in [n-1]$ is in the intermediate bag for the addition/deletion $(v_{1,j}, y_{1,j})$;
- $\{u_{r,j}, y_{r,j+1}\}$ for each $j \in [n-1]$ is in the intermediate bag for the addition/deletion $(y_{r,j+1}, u_{r,j})$;
- $\{v_{i,j}, u_{i+1,j}\}$ for each $i \in [r-1]$ and $j \in [n-1]$ is in the intermediate bag for the addition/deletion $(u_{i+1,j}, y_{i+1,j})$, this is clear when we realize that $v_{i,j}$ is added in the addition/deletion step $(v_{i,j}, u_{i,j})$ or $(v_{1,j}, y_{1,j})$ two steps before $(u_{i+1,j}, y_{i+1,j})$;
- The edges $\{y_{i,n}, u_{i,n}\}$ for $i \in \{2, \cdots, r-1\}$ are in the intermediate bag for the addition/deletion $(u_{i,n}, y_{i,n})$ in the last part of the sequence;
- The edges $\{y_{1,n}, v_{2,0}\}$, $\{u_{2,n}, u_{3,n}\}, \ldots, \{u_{r-2,n}, u_{r-1,n}\}$ and $\{u_{r-1,n}, y_{r,n}\}$ are in the final bag.

Hence our proposed tree decomposition is indeed a tree decomposition, and the treewidth of N is at most r.

For the lower bound, observe that the $r \times (n+1)$ grid is a minor of N. This grid has treewidth r, so $tw(N) \geq r$. Combining the upper and lower bound, we conclude that the treewidth of N is exactly r. $\square$

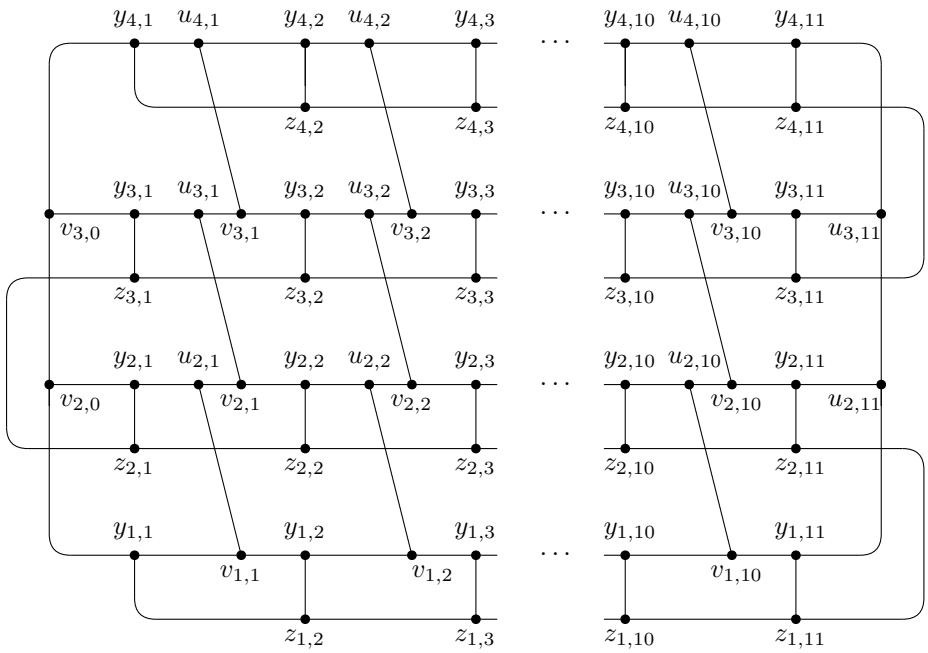

Figure 5: The display graph $D(N,T)$.

In order to show that $tw(D(N,T)) \geq 2r$, we use the concept of *brambles*. We will construct a bramble B in $D(N,T)$ of order $2r+1$. This implies that $tw(D(N,T)) \geq 2r$. Below we describe the vertex sets of the brambles in B. We note that for convenience of description, some of these sets contain vertices such as $v_{1,0}$ that were deleted or suppressed in the construction of N. Such vertices should be ignored for the purposes of defining an induced subgraph.

The bramble B contains the subgraphs induced by $D(N,T)$ on the following sets:

- For each $i \in [r-1]$ and $1 \leq j < n$, the set

$$S_{i,j} = \{u_{i,l}, v_{i,l}, y_{i,l} : l \in [n-1] \cup \{0\}\} \cup \{y_{h,j}, u_{h,j}, v_{h,j} : h \in [r]\}$$

- For each $i \in [r]$ and $1 \leq j < n$, the set

$$T_{i,j} = \{z_{i,l} : l \in [n]\} \cup \{y_{h,j}, u_{h,j}, v_{h,j} : h \in [r]\}$$

- The set $End = \{y_{h,n}, u_{h,n} : h \in [r]\}$
- The set $Top = \{y_{r,l}, u_{r,l} : l \in [n-1]\}$

Intuitively, one may think of the graph $D(N,T)$ as being split up into "rows" and "columns", with a "column" being made up of the vertices $y_{i,j}, u_{i,j}, v_{i,j}$ for

some fixed j and all values of i. A "row" either consists of all $y_{i,j}, u_{i,j}, v_{i,j}$ for a fixed i, or all $z_{i,j}$ for a fixed i. The set End consists of all vertices in the last column, and the set Top consists of all vertices in the top row (except for those already in End). The sets $S_{i,j}$ and $T_{i,j}$ combine all vertices from a given row and column (except those vertices already in End). Note that End is vertex-disjoint from all the other sets; this will be crucial for the lower bound on the order of B.

Lemma 6 *B is a bramble in $D(N,T)$.*

Proof: Observe that all the sets induce a connected subgraph of $D(N,T)$. (In particular, the "columns" are connected because of the edges $\{v_{i,j}, u_{i+1,j}\}$; also note that for $T_{i,j}$ the sets $\{z_{i,l} : l \in [n-1]\}$ and $\{y_{h,j}, u_{h,j}, v_{h,j} : h \in [r]\}$ are connected by the edge $\{z_{i,j}, y_{i,j}\}$.) It remains to show that for each pair of sets in B the sets either share a vertex or are joined by an edge with one vertex in each set.

To see that the sets Top and End touch, observe that Top contains $u_{r,n-1}$ and End contains $y_{r,n}$, and these vertices are connected by an edge. To see that End touches the other sets, observe that all other sets contain either the vertex $z_{i,n}$ or $v_{i,n-1}$ for some $i \in [r]$. As both of these vertices are adjacent to $y_{i,n}$, it follows that End is touches each of these sets.

To see that Top touches each of the other sets except for End, observe that each of these sets contains $y_{r,j}$ for some $1 \leq j < n$. As $y_{r,j}$ is also in Top, these sets touch.

It remains to consider pairs of sets where each set is $S_{i,j}$ or $T_{i,j}$ for some $i \in [r]$ and $j \in [n-1]$. First consider a set $S_{i,j}$ and a set $T_{i',j'}$. As both these sets contain $y_{i,j'}$, the sets touch. Next consider sets $S_{i,j}$ and $S_{i',j'}$. As both these sets contain $y_{i,j'}$, the sets touch. Finally consider the set $T_{i,j}$ and $T_{i',j'}$. Then $T_{i,j}$ contains $z_{i,j'}$ and $T_{i',j'}$ contains $y_{i,j'}$. As these vertices are adjacent, the sets touch. □

Lemma 7 *The order of B is $2r+1$.*

Proof: Observe that the set $\{y_{i,2}, z_{i,2} : i \in [r]\} \cup \{y_{1,n}\}$ is a hitting set of size $2r+1$.

To see that any hitting set must have size at least $2r+1$, suppose for a contradiction that H is a hitting set for B with $|H| \leq 2r$. As $n > 2r+2$, there exists some $1 < j < n$ such that H does not contain $u_{i,j}, v_{i,j}, y_{i,j}$ or $z_{i,j}$ for any $i \in [r]$. For each $i \in [r]$, H contains elements from $T_{i,j}$, from which it follows that H must contain some element from $\{z_{i,l} : l \in [n]\}$ for each $i \in [r]$. Similarly as H contains elements from $S_{i,j}$, H must contain some element from $\{u_{i,l}, v_{i,l}, y_{i,l} : l \in [n-1] \cup \{0\}\}$ for each $i \in [r-1]$. In addition, H must contain some element from $Top = \{y_{r,l}, u_{r,l} : l \in [n-1]\}$.

As these sets are disjoint and there are $2r$ of them, H must contain exactly one element from each of these sets. But as each of these sets is disjoint from $End = \{y_{h,n}, u_{h,n} : h \in [r]\}$, it follows that H contains no element of End, a contradiction. □

This shows that the treewidth of the display graph $D(N,T)$ is at least $2r$. From the above three lemmas we have the following:

Theorem 6 *For any positive integer r, there is a network N of treewidth r and a tree T such that N displays T and $tw(D(N,T)) \geq 2r$.*

5 Discussion and conclusions

An obvious open question is whether we can match the theoretical upper and constructive lower bound on the treewidth of $D(N,T)$ in terms of the treewidth of N. This means either finding a tight example of the inequality $tw(D(N,T)) \leq 2tw(N)+1$, *or* improving the upper bound to match the $2tw(N)$ lower bound of the construction from the previous section. It is also natural to explore *empirically* how large the treewidth of $D(N,T)$ is compared to the treewidth of N, when N displays T. We conjecture that for realistic phylogenetic trees and networks $tw(D(N,T))$ will be much smaller than $2tw(N)$.

As touched upon in Section 4 it could additionally be interesting to identify non-trivial examples when N does not display T but $tw(D(N,T)) = tw(N)$ and to give, if possible, a phylogenetic interpretation to this. Phylogenetics has defined many topologically-restricted subclasses of phylogenetic networks, such as *tree-based* networks [21], precisely to prohibit networks (such as that shown in Figure 1) that are artificially large and complex with respect to the number/location of taxa in the network. Possibly the display relation will behave differently on such restricted subclasses with respect to $tw(D(N,T))$. In any case, recent advances in treewidth solvers will be useful here (see e.g. [4]) since display graphs can quickly become quite large. We now understand that, after suppression of degree-2 nodes, display graphs of two phylogenetic trees are exactly those (biconnected, cubic) graphs of tree arboricity 2; is there any hope of computing treewidth quickly on these graphs? See the related discussion in [33].

Algorithmically, the obvious challenge that (still!) remains is to convert MSOL formulations into practical dynamic programming algorithms running over tree decompositions. This remains tempting, for the following reason. In [34] it is reported that display graphs of two trees T_1, T_2 often have low treewidth compared to even conservative phylogenetic dissimilarity measures on T_1, T_2, such as Tree Bisection and Reconnect (TBR) distance, and this makes computation of these measures (paramerized by treewidth of the display graph) attractive. But what about networks - as opposed to display graphs? In phylogenetics it is quite common to construct phylogenetic networks by asking for a network N that simultaneously displays two (or more) trees T_1, T_2 and which minimizes $r(N)$; this is the well-studied *hybridization number* problem [11, 43]. In such an N, $r(N)$ will be equal to the TBR-distance of T_1 and T_2 [44] which, as mentioned earlier, can be large compared to $tw(D(T_1,T_2))$. The question arises how $tw(N)$ relates to $tw(D(T_1,T_2))$ and, in particular, whether $tw(N)$ is also "low". If so, there is some hope that phylogenetic networks arising in

practice will also have low treewidth, compared to other phylogenetic measures. More empirical study is needed in this area.

The obvious theoretical shortcoming of this approach is that phylogenetic MSOL formulations are complex and explicit dynamic programs require some effort to write and understand (see e.g. [3]) with relatively high exponential dependency on the treewidth bound. The UTC formulation in this article nevertheless seems a promising candidate for a "clean" explicit dynamic program since it has, by phylogenetic standards, a comparatively straightforward combinatorial structure.

Looking forward we observe that, as phylogenetic networks become more commonplace in computational biology, it is natural to compare networks, rather than trees (see e.g. [22, 30]). In this regard, *network-network* display graphs are certainly worthy of investigation. For example, it is straightforward to prove that if two phylogenetic networks N_a, N_b both display a tree T, $tw(D(N_a, N_b)) \leq r(N_a)+r(N_b)+2$. Now, if N_a and N_b are two distinct optima (i.e. competing hypotheses) produced by an algorithm solving the hybridization number problem for two trees T_1, T_2, then $r(N_a)$ and $r(N_b)$ are both equal to the TBR-distance d of T_1 and T_2 [44]. Hence, $tw(D(N_a, N_b)) \leq 2d + 2$. In particular: the treewidth of the display graph formed from the networks, will be bounded as a function of the TBR-distance of the two original trees. Similarly, the proof of Lemma 2 goes through essentially unchanged for two networks on the same set of taxa: if N_2 displays N_1 then $tw(D(N_2, N_1)) \leq 2tw(N_2) + 1$.

Perhaps such treewidth bounds can help in the development of compact FPT MSOL proofs for determining the dissimilarity of networks. There is quite some potential here. Topological decompositions in phylogenetics (into quartets, triplets, *agreement forests* and so on) can be modelled fairly naturally within MSOL [34]. Higher-order analogues are emerging for decomposing phylogenetic networks (see e.g. [28]) - and it is plausible that such structures could also be encoded within MSOL.

Finally, stepping away from phylogenetics, the study of display graphs continues to generate interesting new questions for algorithmic graph theory. In particular, the behaviour (and "phylogenetic meaning") of (forbidden) minors in display graphs remains a subject where much is still to be learned [19, 33]. Indeed, display graphs can be viewed as a special case of a more generic problem. Given a set of graphs and a well-defined protocol for merging them, how do parameters of the merged graph (and topological features such as minors) relate to parameters and features of the constituent graphs?

Acknowledgements

Mark Jones and Remie Janssen were supported by Leo van Iersel's Vidi grant (NWO): 639.072.602. Georgios Stamoulis was supported by an NWO TOP 2 grant. Part of the work was supported by CNRS "Projet international de cooperation scientifique (PICS)" grant number 230310 (CoCoAlSeq).

A Appendix

A.1 Unrooted tree containment (UTC) is FPT when parameterized by treewidth: a proof via Courcelle's Theorem

This leverages the upper bound on $tw(D(N,T))$ as a function of the treewidth $tw(N)$ of N proven earlier in the paper, see Lemma 2.

The high-level idea of the following MSOL formulation is that, if N displays T, then (as discussed in Section 4) N contains some subtree T' that is a subdivision of T and which can be "grown" into a spanning tree T'' of N. Spanning trees of N are precisely those subgraphs obtained by deleting a subset of edges E' from N to make it connected and acyclic. Note that the set of quartets (unrooted phylogenetic trees on subsets of exactly 4 taxa) displayed by T'' is identical to those displayed by T', which is identical to those displayed by T. (In other words, subdivision operations, and pendant subtrees without taxa that possibly hang from T'', do not induce any extra quartets.)

The core idea underpinning MSOL is to query properties of a graph using universal and existential quantification ranging not just over vertices and edges, but also subsets of these objects. For the benefit of readers not familiar with MSOL we now show how various basic auxiliary predicates can be easily constructed and combined to obtain more powerful predicates. (The article [34] gives a more comprehensive inroduction to the use of these techniques in phylogenetics). The MSOL sentence will be queried over the display graph $D(N,T)$ where we let V be the vertex set of $D(N,T)$ and E its edge set. Here R^D is the edge-vertex incidence relation on $D(N,T)$. We let V_T, V_N, E_T, E_N denote those vertices and edges of $D(N,T)$ which belong to T, N respectively (note that $V_T \cap V_N = X$). Alongside X, V, E all this information is available to the MSOL formulation via its *structure*.

- test that Z is equal to the union of two sets P and Q:

  $$P \cup Q = Z := \forall z(z \in Z \Rightarrow z \in P \vee z \in Q) \\ \wedge \forall z(z \in P \Rightarrow z \in Z) \wedge \forall z(z \in Q \Rightarrow z \in Z).$$

- test that $P \cap Q = \emptyset$:

  $$\text{NoIntersect}(P,Q) := \forall u \in P(u \notin Q).$$

- test that $P \cap Q = \{v\}$:

  $$\text{Intersect}(P,Q,v) := (v \in P) \wedge (v \in Q) \wedge \forall u \in P(u \in Q \Rightarrow (u = v)).$$

- test if the non-empty sets P and Q are a bipartition of Z:

  $$\text{Bipartition}(Z,P,Q) := (P \neq \emptyset) \wedge (Q \neq \emptyset) \wedge (P \cup Q = Z) \wedge \text{NoIntersect}(P,Q).$$

- test if the elements in $\{x_1, x_2, x_3, x_4\}$ are pairwise different:

$$\text{allDiff}(x_1, x_2, x_3, x_4) := \bigwedge_{i \neq j \in \{1,2,3,4\}} x_i \neq x_j.$$

- check if the nodes p and q are adjacent:

$$\text{adj}(p, q) := \exists e \in E(R^D(e, p) \wedge R^D(e, q)).$$

The complex predicate $PAC(Z, x_1, x_2, K)$ ("path avoiding edge cuts?") asks: is there a path from x_1 to x_2 entirely contained inside vertices Z that avoids all the edges K? We model this by observing that this does *not* hold if you can partition Z into two pieces P and Q, with $x_1 \in P$ and $x_2 \in Q$, such that the only edges that cross the induced cut (if any) are in K.

$$\begin{aligned}
&PAC(Z, x_1, x_2, K) := \\
&\quad (x_1 = x_2) \vee \neg\exists P, Q \Big(\text{Bipartition}(Z, P, Q) \wedge x_1 \in P \wedge x_2 \in Q \\
&\quad \wedge \Big(\forall p, q \Big(p \in P \wedge q \in Q \Rightarrow \neg\text{adj}(p, q) \vee \big(\exists g \in K \left(R^D(g, p) \wedge R^D(g, q) \right) \big) \Big) \Big) \Big)
\end{aligned}$$

The following predicate QAC^i ("quartet avoiding edge cuts?"), where $i \in \{T, N\}$, returns true if and only if i contains an image of quartet $x_a x_b | x_c x_d$ that is disjoint from the edge cuts K. As usual we write $x_a x_b | x_c x_d$ to denote the quartet where the path between x_a and x_b is disjoint from the path between x_c and x_d. (The tree T shown in Figure 1, for example, is the quartet $ab|cd$).

$$\begin{aligned}
&QAC^i(x_a, x_b, x_c, x_d, K) := \\
&\quad \exists u, v \in V_i \Big((u \neq v) \wedge \exists A, B, C, D, P \subseteq V_i \Big(u \in P \wedge v \in P \\
&\quad \wedge x_a, u \in A \qquad \wedge x_b, u \in B \qquad \wedge x_c, v \in C \\
&\quad \wedge x_d, v \in D \qquad \wedge \text{Intersect}(A, B, u) \qquad \wedge \text{Intersect}(A, P, u) \\
&\quad \wedge \text{Intersect}(B, P, u) \qquad \wedge \text{Intersect}(C, D, v) \qquad \wedge \text{Intersect}(C, P, v) \\
&\quad \wedge \text{Intersect}(D, P, v) \qquad \wedge \text{NoIntersect}(A, C) \qquad \wedge \text{NoIntersect}(B, C) \\
&\quad \wedge \text{NoIntersect}(A, D) \qquad \wedge \text{NoIntersect}(B, D) \qquad \wedge PAC(A, u, x_a, K) \\
&\quad \wedge PAC(B, u, x_b, K) \qquad \wedge PAC(C, v, x_c, K) \qquad \wedge PAC(D, v, x_d, K) \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \wedge PAC(P, u, v, K) \Big) \Big)
\end{aligned}$$

We need a prediate which asks: is the subgraph induced by vertex subset Z, and then with edges K deleted, connected? We model this as follows: for

every pair of vertices u and v in Z a path should exist from u to v completely contained inside Z and which avoids the edges K. Hence,

$$\text{Connected}(Z, K) \quad := \forall u, v \in Z(PAC(Z, u, v, K)).$$

In a similar vein, we need a predicate which asks: is the subgraph induced by vertex subset Z, and then with edges K deleted, *acyclic?* The idea here is that, if it is not acyclic, there will exist two distinct vertices $u, v \in Z$ such that u can reach v via two distinct, vertex-disjoint paths P and Q:

$$\begin{aligned}\text{Acyclic}(Z, K) \quad := \quad & \neg\exists u, v \in Z\Big(\exists P, Q \subseteq Z\big(u \neq v \wedge P \cap Q = \{u, v\} \\ & \wedge P \neq Q \wedge PAC(P, u, v, K) \wedge PAC(Q, u, v, K)\big)\Big).\end{aligned}$$

(The predicate $P \cap Q = \{u, v\}$ is a simple modification of the earlier Intersect predicate.)

The final formulation is shown as below. The first line asks for a subset E' (representing the edges we delete from N to obtain T'') such that the N part of $D(N, T)$ remains connected and acyclic after deletion of E' (and thus induces a spanning tree), and from the second line onwards we stipulate that, after deletion of E', the set of quartets that survive is exactly the same as the set of quartets displayed by T. (This is leveraging the well-known result from phylogenetics that two trees are compatible if and only if they display the same set of quartets [38]). Note that the overall length of the MSOL fragment is fixed, i.e., it is not dependent on parameters of the input.

$$\begin{aligned}&\exists E' \subseteq E_N\Bigg(\text{Connected}(V_N, E') \wedge \text{Acyclic}(V_N, E') \\ &\quad \wedge \forall x_1, x_2, x_3, x_4 \in X\Bigg(\text{allDiff}(x_1, x_2, x_3, x_4) \\ &\quad \Rightarrow \quad \Big(\big(QAC^T(x_1, x_2, x_3, x_4, \emptyset) \quad \Leftrightarrow QAC^N(x_1, x_2, x_3, x_4, E')\big) \\ &\qquad \wedge \big(QAC^T(x_1, x_3, x_2, x_4, \emptyset) \quad \Leftrightarrow QAC^N(x_1, x_3, x_2, x_4, E')\big) \\ &\qquad \wedge \big(QAC^T(x_1, x_4, x_2, x_3, \emptyset) \quad \Leftrightarrow QAC^N(x_1, x_4, x_2, x_3, E')\big)\Big)\Bigg)\Bigg).\end{aligned}$$

A.2 MSOL proof for recognizing display graphs

The following MSOL fragment checks whether a cubic, simple graph $G = (V, E)$ is a suppressed display graph. We re-use predicates defined in the previous section.

$$\exists V_1, V_2(\text{Bipartition}(V, V_1, V_2) \wedge_{i=1,2} \text{Connected}(V_i, \emptyset) \wedge_{i=1,2} \text{Acyclic}(V_i, \emptyset)).$$

References

[1] B. Allen and M. Steel. Subtree transfer operations and their induced metrics on evolutionary trees. *Annals of Combinatorics*, 5:1–15, 2001. doi:10.1007/s00026-001-8006-8.

[2] S. Arnborg, J. Lagergren, and D. Seese. Easy problems for tree-decomposable graphs. *Journal of Algorithms*, 12:308 – 340, 1991. doi:10.1016/0196-6774(91)90006-K.

[3] J. Baste, C. Paul, I. Sau, and C. Scornavacca. Efficient fpt algorithms for (strict) compatibility of unrooted phylogenetic trees. *Bulletin of Mathematical biology*, 79(4):920–938, 2017. doi:10.1007/s11538-017-0260-y.

[4] S. Berndt. Computing tree width: From theory to practice and back. In F. Manea, R. G. Miller, and D. Nowotka, editors, *Sailing Routes in the World of Computation - 14th Conference on Computability in Europe, CiE 2018, Proceedings*, volume 10936 of *Lecture Notes in Computer Science*, pages 81–88. Springer, 2018. doi:10.1007/978-3-319-94418-0_8.

[5] H. Bodlaender. A tourist guide through treewidth. *Acta cybernetica*, 11(1-2):1–21, 1994.

[6] H. Bodlaender. A linear-time algorithm for finding tree-decompositions of small treewidth. *SIAM Journal on Computing*, 25(6):1305–1317, Dec. 1996. doi:10.1137/S0097539793251219.

[7] H. Bodlaender. A partial k-arboretum of graphs with bounded treewidth. *Theoretical Computer Science*, 209(1-2):1–45, 1998. doi:10.1016/S0304-3975(97)00228-4.

[8] H. Bodlaender and A. Koster. Treewidth computations I. Upper bounds. *Information and Computation*, 208(3):259–275, 2010. doi:10.1016/j.ic.2009.03.008.

[9] H. Bodlaender and A. Koster. Treewidth computations II. Lower bounds. *Information and Computation*, 209(7):1103–1119, 2011. doi:10.1016/j.ic.2011.04.003.

[10] M. Bordewich, C. Scornavacca, N. Tokac, and M. Weller. On the fixed parameter tractability of agreement-based phylogenetic distances. *Journal of Mathematical Biology*, 74(1-2):239–257, 2017. doi:10.1007/s00285-016-1023-3.

[11] M. Bordewich and C. Semple. Computing the hybridization number of two phylogenetic trees is fixed-parameter tractable. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 4(3):458–466, 2007. doi:10.1109/tcbb.2007.1019.

[12] D. Bryant and J. Lagergren. Compatibility of unrooted phylogenetic trees is FPT. *Theoretical computer science*, 351(3):296–302, 2006. doi:10.1016/j.tcs.2005.10.033.

[13] G. Chang, C. Chen, and Y. Chen. Vertex and tree arboricities of graphs. *Journal of Combinatorial Optimization*, 8(3):295–306, 2004. doi:10.1023/B:JOCO.0000038912.82046.17.

[14] B. Courcelle. The monadic second-order logic of graphs. I. Recognizable sets of finite graphs. *Information and Computation*, 85:12–75, 1990. doi:10.1016/0890-5401(90)90043-H.

[15] M. Cygan, F. Fomin, L. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, and S. Saurabh. *Parameterized Algorithms*. Springer Publishing Company, Incorporated, 1st edition, 2015. doi:10.1007/978-3-319-21275-3.

[16] Y. Deng and D. Fernández-Baca. Fast compatibility testing for rooted phylogenetic trees. *Algorithmica*, 80(8):2453–2477, 2018. doi:10.1007/s00453-017-0330-4.

[17] R. Diestel. *Graph Theory*. Springer-Verlag Berlin and Heidelberg GmbH & Company KG, 2010.

[18] J. Felsenstein. *Inferring Phylogenies*. Sinauer Associates Sunderland, MA, 2004.

[19] D. Fernández-Baca and S. Vakati. On compatibility and incompatibility of collections of unrooted phylogenetic trees. *Discrete Applied Mathematics*, 245:42–58, 2018. doi:10.1016/j.dam.2017.05.002.

[20] M. Fischer, L. Van Iersel, S. Kelk, and C. Scornavacca. On computing the maximum parsimony score of a phylogenetic network. *SIAM Journal on Discrete Mathematics*, 29(1):559–585, 2015. doi:10.1137/140959948.

[21] A. Francis, K. Huber, and V. Moulton. Tree-based unrooted phylogenetic networks. *Bulletin of Mathematical Biology*, 80(2):404–416, 2018. doi:10.1007/s11538-017-0381-3.

[22] A. Francis, K. Huber, V. Moulton, and T. Wu. Bounds for phylogenetic network space metrics. *Journal of Mathematical Biology*, 76(5):1229–1248, 2018. doi:10.1007/s00285-017-1171-0.

[23] P. Gambette, V. Berry, and C. Paul. Quartets and unrooted phylogenetic networks. *Journal of Bioinformatics and Computational Biology*, 10(4):1250004, 2012. doi:10.1142/S0219720012500047.

[24] A. Grigoriev, S. Kelk, and L. Lekic. On low treewidth graphs and supertrees. *Journal of Graph Algorithms and Applications*, 19(1):325–343, 2015. doi:10.7155/jgaa.00361.

[25] M. Grohe, D. Neuen, and P. Schweitzer. A faster isomorphism test for graphs of small degree. *CoRR*, abs/1802.04659, 2018. URL: `http://arxiv.org/abs/1802.04659`.

[26] A. Gunawan, B. Lu, and L. Zhang. A program for verification of phylogenetic network models. *Bioinformatics*, 32(17):i503–i510, 2016. `doi:10.1093/bioinformatics/btw467`.

[27] K. Huber, V. Moulton, and T. Wu. Transforming phylogenetic networks: Moving beyond tree space. *Journal of Theoretical Biology*, 404:30–39, 2016. `doi:10.1016/j.jtbi.2016.05.030`.

[28] K. Huber, L. van Iersel, V. Moulton, C. Scornavacca, and T. Wu. Reconstructing phylogenetic level-1 networks from nondense binet and trinet sets. *Algorithmica*, 77(1):173–200, 2017. `doi:10.1007/s00453-015-0069-8`.

[29] D. Huson, R. Rupp, and C. Scornavacca. *Phylogenetic Networks: Concepts, Algorithms and Applications*. Cambridge University Press, 2011.

[30] R. Janssen, M. Jones, P. Erdős, L. van Iersel, and C. Scornavacca. Exploring the tiers of rooted phylogenetic network space using tail moves. *Bulletin of Mathematical Biology*, 80(8):2177–2208, 2018. `doi:10.1007/s11538-018-0452-0`.

[31] J. Keijsper and R. Pendavingh. Reconstructing a phylogenetic level-1 network from quartets. *Bulletin of Mathematical Biology*, 76(10):2517–2541, 2014. `doi:10.1007/s11538-014-0022-z`.

[32] S. Kelk and C. Scornavacca. Constructing minimal phylogenetic networks from softwired clusters is fixed parameter tractable. *Algorithmica*, 68(4):886–915, 2014. `doi:10.1007/s00453-012-9708-5`.

[33] S. Kelk, G. Stamoulis, and T. Wu. Treewidth distance on phylogenetic trees. *Theoretical Computer Science*, 731:99–117, 2018. `doi:10.1016/j.tcs.2018.04.004`.

[34] S. Kelk, L. van Iersel, C. Scornavacca, and M. Weller. Phylogenetic incongruence through the lens of monadic second order logic. *Journal of Graph Algorithms and Applications*, 20(2):189–215, 2016. `doi:10.7155/jgaa.00390`.

[35] E. Luks. Isomorphism of graphs of bounded valence can be tested in polynomial time. *Journal of Computer and System Sciences*, 25(1):42–65, 1982. `doi:10.1016/0022-0000(82)90009-5`.

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

[37] A. Raspaud and W. Wang. On the vertex-arboricity of planar graphs. *European Journal of Combinatorics*, 29(4):1064–1075, 2008. doi:10.1016/j.ejc.2007.11.022.

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

[39] P. Seymour and R. Thomas. Graph searching and a min-max theorem for tree-width. *Journal of Combinatorial Theory, Series B*, 58(1):22–33, 1993. doi:10.1006/jctb.1993.1027.

[40] C. Solís-Lemus and C. Ané. Inferring phylogenetic networks with maximum pseudolikelihood under incomplete lineage sorting. *PLoS Genetics*, 12(3):1–21, 2016. doi:10.1371/journal.pgen.1005896.

[41] M. Steel. *Phylogeny: Discrete and random processes in evolution*. SIAM, 2016.

[42] S. Vakati and D. Fernández-Baca. Graph triangulations and the compatibility of unrooted phylogenetic trees. *Applied Mathematics Letters*, 24(5):719–723, 2011. doi:10.1016/j.aml.2010.12.015.

[43] L. van Iersel, S. Kelk, and C. Scornavacca. Kernelizations for the hybridization number problem on multiple nonbinary trees. *Journal of Computer and System Sciences*, 82(6):1075 – 1089, 2016. doi:10.1016/j.jcss.2016.03.006.

[44] L. van Iersel, S. Kelk, G. Stamoulis, L. Stougie, and O. Boes. On unrooted and root-uncertain variants of several well-known phylogenetic network problems. *Algorithmica*, 80(11):2993–3022, 2018. doi:10.1007/s00453-017-0366-5.

[45] L. van Iersel, C. Semple, and M. Steel. Locating a tree in a phylogenetic network. *Information Processing Letters*, 110(23):1037–1043, 2010. doi:10.1016/j.ipl.2010.07.027.