

Treewidth-Based Algorithms for the Small Parsimony Problem on Networks

Celine Scornavacca, Mathias Weller

▶ **To cite this version:**

Celine Scornavacca, Mathias Weller. Treewidth-Based Algorithms for the Small Parsimony Problem on Networks. 21st International Workshop on Algorithms in Bioinformatics (WABI), Aug 2021, Chicago. Due to COVID-19, WABI 2021 will be held online., United States. pp.6:1, 10.4230/LIPIcs.WABI.2021.6 . hal-03287112

HAL Id: hal-03287112
https://hal.science/hal-03287112

Submitted on 15 Jul 2021

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

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

Treewidth-based Algorithms for the Small Parsimony Problem on Networks

Celine Scornavacca
Institut des Sciences de l'Evolution, Université de Montpellier, CNRS, IRD, EPHE, France

Mathias Weller
LIGM, CNRS, Université Gustave Eiffel, Paris, France

Abstract

Phylogenetic reconstruction is one of the paramount challenges of contemporary bioinformatics. A subtask of existing tree reconstruction algorithms is modeled by the Small Parsimony problem: given a tree T and an assignment of character-states to its leaves, assign states to the internal nodes of T such as to minimize the *parsimony score*, that is, the number of edges of T connecting nodes with different states. While this problem is polynomial-time solvable on trees, the matter is more complicated if T contains reticulate events such as hybridizations or recombinations, i.e. when T is a network. Indeed, three different versions of the parsimony score on networks have been proposed and each of them is NP-hard to decide. Existing parameterized algorithms focus on combining the number of possible character-states with the number of reticulate events (per biconnected component). Here, we consider the treewidth of the undirected graph underlying the input network as parameter, presenting dynamic programming algorithms for (slight generalizations of) all three versions of the parsimony problem on networks. Our algorithms use a formulation of the treewidth that may facilitate formalizing treewidth-based dynamic programming algorithms on phylogenetic networks for other problems .

2012 ACM Subject Classification Theory of computation → Fixed parameter tractability; Applied computing → Molecular sequence analysis

Keywords and phrases Phylogenetics, parsimony, phylogenetic networks, parameterized complexity, dynamic programming, treewidth

Digital Object Identifier 10.4230/LIPIcs.WABI.2021.6

Funding This work was supported by French Agence Nationale de la Recherche through the CoCoAlSeq project (ANR-19-CE45-0012).

Acknowledgements We thank Christophe Paul for sharing his expertise on treewidth formulations.

1 Introduction

Molecular phylogenetic reconstruction consists in inferring a well-founded evolutionary scenario of a set of species from molecular data [12]. An evolutionary scenario, also called a *phylogeny*, is usually represented by a directed tree with a unique source called *root*. In a phylogeny, the tips of the tree are associated to extant species for which we have data, and each internal node represents an extinct species giving rise to new species – a *speciation*. Therefore, each internal node represents the hypothetical ancestor of all species below it, and the root models the lowest common ancestor of all the species at the tips.

Parsimony on Trees.

In this paper, molecular data consists of a set of molecular sequences (e.g. DNA or protein sequences) of the same length (one sequence per species). This kind of data can be seen as a matrix M of n sequences, each having m characters (exhibiting one of c possible states) where the state $M_{i,j}$ corresponds to the j^{th} character of the i^{th} species. There are several


© Celine Scornavacca and Mathias Weller;
licensed under Creative Commons License CC-BY 4.0



methods to reconstruct well-founded phylogenies from matrices of characters [12]. They are all based on the idea of retrieving similarities among species by comparing the states taken by these species at the different characters of M. Here, we will focus on *parsimony methods*. The main hypothesis of these methods is that character changes are not frequent. Thus, the phylogenies that best explain the data are those requiring the fewest evolutionary changes, i.e. the ones having the optimal *parsimony score*, formally defined in Section 4. The problem of finding the optimal parsimony score for a given phylogeny T with respect to a matrix M is called the SMALL PARSIMONY problem and can be solved in $O(n \cdot m \cdot c)$ time [14] since each column in the matrix can be analyzed independently in linear time. When T is unknown, the problem of finding the phylogeny minimizing the parsimony score is called the BIG PARSIMONY problem. This latter is known to be NP-hard and numerous heuristic techniques for it are known [12].

Parsimony on Networks.

When the evolution of the species of interest include, in additions to speciations, reticulate events such as *hybridizations* or *recombinations*, a single species may inherit from multiple direct ancestors. In this case, the phylogenies are no longer represented by rooted trees but by rooted DAGs [16] called *networks*. When scoring a given network, three very different definitions of the parsimony score have been proposed: the *hardwired* [20], the *softwired* [15, 26], and the *parental* parsimony score [32]. Roughly, the hardwired score takes into account all edges of the given network (characters are inherited from all parents), the softwired score takes only the edges of any "switching" (each character is inherited from one parent), and the parental score allows embedding lineages into the network (each allele of a character is inherited from one parent). See Section 4 for details and Figure 3 for an example. While these definitions coincide for trees, they give rise to three different small parsimony problems for networks.

When tracing mutually dependent characters (e.g. different genomic locations in a same non-recombinant region) on networks, we also have to make sure that dependent characters are inherited from the same parent (some columns of the matrix have to use the same "switching"/"embedding"). To avoid dealing with this problem, the small parsimony problems on networks have been studied predominantly under the assumption of independent genomic locations. This boils down to having $m = 1$ since each column of the matrix can be analyzed independently (as is the case for the small parsimony problem on trees). Another popular restriction is to consider *binary* networks, in which the root has outdegree 2, tips have indegree 1, and internal nodes have either indegree 1 and outdegree 2 (speciations) or indegree 2 and outdegree 1 (reticulations).

The hardwired small parsimony problem has been proven NP-hard and APX-hard whenever the number of states that a character can take, denoted c, is strictly greater than 2, and polynomial time solvable for binary characters [13]. A polynomial-time 1.35-approximation for all c and a $12/11$-approximation for $c = 3$ have been proposed [13]. Additionally, the problem has been shown fixed-parameter tractable (FPT) in the parsimony score [13], and with respect to $c + r$, where r is the number of reticulate events in the network [21].

The softwired small parsimony problem is also NP-hard and APX-hard [19, 13] for binary characters, and not FPT in the parsimony score (it is NP-hard to know if the softwired parsimony score is 1). Also, it has been shown that, for any constant $\epsilon > 0$, an approximation factor of $n^{1-\epsilon}$ is not possible in polynomial time, unless P = NP. On the positive side, the problem is FPT in $c + r$ [26, 13] and $c + \ell$, where ℓ is the *level* of the network [18, 13] (the maximum number of reticulations over all biconnected components of the network).

Unsurprisingly, the parental small parsimony problem has also been proven NP-hard,

even for very restricted classes of networks [29], but is FPT both with respect to $c + r$ and with respect to $c + \ell$.

In this paper, we consider the case of independent characters, showing that the three variants of the small parsimony problem on networks are fixed-parameter tractable with respect to $c + t$, where t is the treewidth of the input network. Our proofs are constructive in the sense that a dynamic programming algorithm is provided for each version of the problem. Since the treewidth can be arbitrary small, even for growing values of ℓ, our algorithms can potentially be orders of magnitude faster than the state-of-the-art solutions. Moreover, our formulations are not limited to binary networks and they can take into account polymorphism as well as external information controlling the states that ancestral species may take.

Treewidth for Phylogenetic Networks

The treewidth of a graph can roughly be described as a measure of "tree-likeness" and it ranks among the smallest of such parameters [2] (in particular, the treewidth can be seen to be smaller than the level ℓ on any network). Together with the fact that it facilitates the design of dynamic programming algorithms, this explains the enormous popularity the treewidth received in the parameterized complexity community [5]. Starting with the groundbreaking work of Bryant and Lagergren [7] (using the celebrated result of Courcelle [9]), treewidth also gained traction with researchers studying algorithms for phylogenetics-related problems (surveyed in [8]). While this yielded some algorithms parameterized by the treewidth *of the display graph* of multiple trees (the result of "gluing" all trees at their leaves), we are not aware of any algorithms parameterized by the treewidth of the input network. In an attempt to facilitate the use of this parameter in future work, we dedicate Section 3 to presenting a "phylogenetics-friendly" formulation by representing tree-decompositions of the input network as a rooted tree Γ on the same vertex set as the network. In particular, this formulation generalizes our previously considered parameter "scanwidth" [3], which can be expected to yield easier dynamic programming formulations at the cost of being slightly larger than the treewidth.

Missing proofs are deferred to the appendix at the end of the paper.

2 Preliminaries

Mappings.

For any x and y, we define $\delta(x, y)$ to be 0 if $x = y$ and 1, otherwise, and we abbreviate $1 - \delta(x, y) =: \overline{\delta}(x, y)$. We further abbreviate $\delta(\phi(x), \phi(y))$ as $\delta_\phi(x, y)$ for any function ϕ. We may denote a pair (x, y) as $x \to y$ if it is referring to an assignment of y to x by some function and as xy if it refers to an arc in a network. We sometimes use the name of a function $\phi : X \to Y$ to refer to its set of pairs $\{x \to y \mid \phi(x) = y\}$ and we let $\phi\,|_Z := \{(x \to y) \in \phi \mid x \in Z\}$ denote the *restriction* of ϕ to Z. We say $\phi(x) = \bot$ to indicate that ϕ is not defined for x. We denote the result of forcing $\phi(x) = y$ (whether or not x is mapped by ϕ) as

$$\phi\,[x \to y] := \begin{cases} \phi \cup \{x \to y\} & \text{if } \phi(x) = \bot \\ (\phi \setminus \{x \to \phi(x)\})\,[x \to y] & \text{otherwise} \end{cases}$$

Finally, for sets Z, X and $Y \subseteq X$ and functions ϕ and ψ, we write $\psi \trianglelefteq \phi$ (and say that ψ is a *subfunction* of ϕ) if (a) $\phi : X \to Z$ and $\psi : Y \to Z$ and $\psi(x) \leq \phi(x)$ for all $x \in Y$, or (b) $\phi : X \to 2^Z$ and $\psi : Y \to Z$ and $\psi(x) \in \phi(x)$ for all $x \in Y$, or (c) $\phi : X \to 2^Z$ and $\psi : Y \to 2^Z$ and $\psi(x) \subseteq \phi(x)$ for all $x \in Y$.

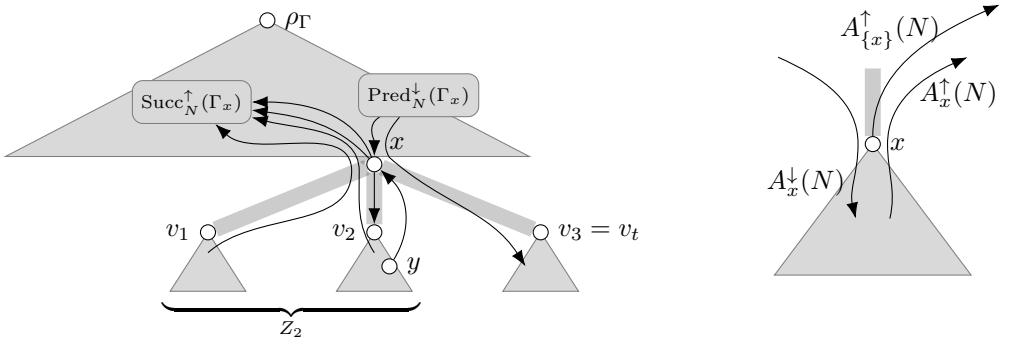


Figure 1 A tree Γ is depicted in gray and some arcs of N are depicted in black. Recall that t is the number of children of x and $Z_i := \bigcup_{1 \leq j \leq i} \Gamma_{v_j}$. Note that $x \in \mathrm{Succ}^{\uparrow}_N(Z_2) \setminus \mathrm{Succ}^{\uparrow}_N(\Gamma_x)$ since x is an ancestor of a node of Γ_{v_2} in N. Note that x is a reticulation of N with parents y (drawn) and z (not drawn) with $y <_\Gamma v_2 <_\Gamma x <_\Gamma z$. Thus, $z \in \mathrm{Pred}^{\downarrow}_N(x)$ but $y \in \mathrm{Pred}^{\uparrow v_2}_N(x) \subseteq \mathrm{Pred}^{\uparrow}_N(x)$. Finally, note that $\mathrm{YW}^\Gamma_x = \mathrm{Pred}^{\downarrow}_N(\Gamma_x) \cup \mathrm{Succ}^{\uparrow}_N(\Gamma_x)$ and $\bigcup_{i \leq t} \mathrm{YW}^\Gamma_{v_i} \subseteq \mathrm{YW}^\Gamma_x \uplus \{x\}$.

Graphs and Phylogenetic Networks.

In this work, we consider (weakly) connected directed acyclic graphs (DAGs) N that have a unique source ρ_N called *root*. If the sinks (aka *leaves*) of N are labeled, we call N a *phylogenetic network*. We denote the set of nodes of N with in-degree at least two by $R(N)$ and we call such nodes *reticulations*. If $R(N) = \varnothing$, then N is called a *tree*. The result of, for each $v \in R(N)$ removing all but one of its incoming arcs is called a *switching* of N and $\mathcal{S}(G)$ denotes the set of all switchings of N (observe that all switchings are spanning trees). Let $v \in V(N)$. We denote the successors (or "children") of v in G by $\mathrm{Succ}_G(v)$ and its predecessors (or "parents") by $\mathrm{Pred}_G(v)$. If N contains a directed u-w-path, then we say that w is a *descendant* of u and u is an *ancestor* of w (denoted as $w \leq_N u$ and $w <_N u$ if $u \neq w$). A set $Z \subseteq V(N)$ such that $u \not\leq_N w$ and $w \not\leq_N u$ for all $u, w \in Z$ is called an *anti-chain* in N. The *induced subgraph* $N[Z]$ of a set $Z \subseteq V(N)$ is the result of removing all nodes $x \in V(N) \setminus Z$ from N (together with their incident arcs) and, for any $v \in V(N)$, the network $N_v := N[\{w \mid w \leq_N v\}]$ is called the subnetwork *rooted at* v.

Large parts of this work are in context of a rooted tree Γ on $V(N)$ (see Figure 1). Specifically for the tree Γ, we permit ourselves to abbreviate $V(\Gamma_x)$ to Γ_x to increase readability. In such context, we additionally define the following sets for any nodes $y, z \in V(N)$: $\mathrm{Pred}^{\uparrow y}_G(z) := \mathrm{Pred}_G(z) \cap \Gamma_y$ and $\mathrm{Pred}^{\downarrow y}_G(z) := \mathrm{Pred}_G(z) \setminus \Gamma_y$ denote the respective *predecessors* of z in N that are or are not in Γ_y. Likewise, $\mathrm{Succ}^{\downarrow y}_G(z) := \mathrm{Succ}_G(z) \cap \Gamma_y$ and $\mathrm{Succ}^{\uparrow y}_G(z) := \mathrm{Succ}_G(z) \setminus \Gamma_y$ denote the respective *successors* of z in N that are or are not in Γ_y – notice that the arrow in the notation indicates the direction of the arc between z and the members of the set when drawing Γ top-to-bottom. If $z = y$, we drop y and simply write $\mathrm{Pred}^{\downarrow}_G(z)$, $\mathrm{Pred}^{\uparrow}_G(z)$, $\mathrm{Succ}^{\downarrow}_G(z)$, and $\mathrm{Succ}^{\uparrow}_G(z)$. We also abbreviate $\mathrm{Pred}^{\downarrow}_G(z) \cap R(G) =: \mathrm{Pred}^{\mathrm{R}\downarrow}_G(z)$ and $\mathrm{Succ}^{\uparrow}_G(z) \cap R(G) =: \mathrm{Succ}^{\mathrm{R}\uparrow}_G(z)$ as well as $\mathrm{Pred}^{\downarrow}_G(z) \setminus R(G) =: \mathrm{Pred}^{\mathrm{T}\downarrow}_G(z)$ and $\mathrm{Succ}^{\uparrow}_G(z) \setminus R(G) =: \mathrm{Succ}^{\mathrm{T}\uparrow}_G(z)$. All these functions generalize to sets $Z \subseteq V(N)$ (for example, $\mathrm{Pred}_G(Z) := \bigcup_{z \in Z} \mathrm{Pred}_G(z) \setminus Z$). Further, for any $X \subseteq V(N)$, we define the sets of arcs of N (a) from a node $u \in X$ to any ancestor of u in Γ as $A^{\uparrow}_X(N) := \{uw \in A(N) \mid u \in X \wedge u <_\Gamma w\}$ and (b) to a node $u \in X$ from any ancestor of u in Γ as $A^{\downarrow}_X(N) := \{uw \in A(N) \mid w \in X \wedge w <_\Gamma u\}$. For brevity, we abbreviate $A_X(N) := A^{\uparrow}_X(N) \cup A^{\downarrow}_X(N)$, $A^{\uparrow}_v(N) := A^{\uparrow}_{\Gamma_v}(N)$, $A^{\downarrow}_v(N) := A^{\downarrow}_{\Gamma_v}(N)$, and $A_v(N) := A_{\Gamma_v}(N)$.

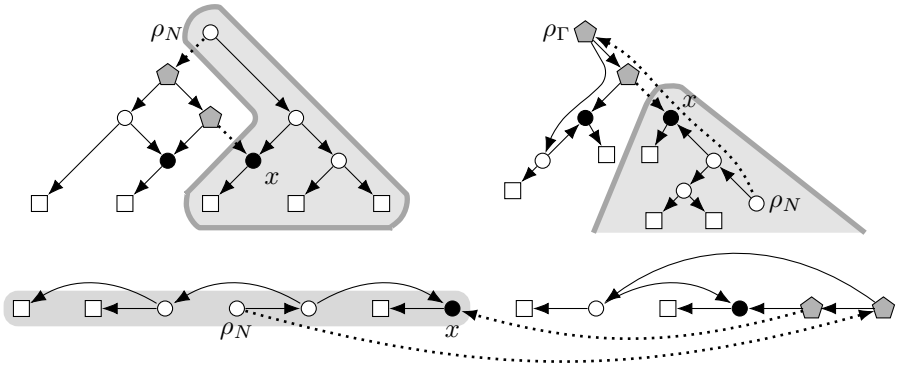


Figure 2 Example of a network N (left) with a linear order σ of its nodes (below) as well as their canonical tree Γ^σ (right) whose arcs are not drawn (the arcs of N are drawn in their stead). Reticulations are black, leaves are boxes. For the first (wrt. σ) reticulation x, the set $V(\Gamma^\sigma_x)$ is marked (gray area), the arcs $uv \in A_x(N)$ are dotted and the nodes in $\mathrm{YW}^\Gamma_v = \mathrm{ZW}^\sigma_v$ are gray pentagons.

3 An Alternative Formulation of Treewidth

In this section, we give an alternative definition of the *treewidth*, which allows to tackle the small parsimony problem for networks in a simpler and more intuitive way. Note that this alternative definition is known in the FPT community (Dendris et al. [11] call it the "support" of a vertex with respect to an ordering (when referring to Arnborg [1]) and Mescoff et al. [25], call it "tree vertex separation"). However, in these works its connection to treewidth is mostly touched in passing, so we felt the need to prove it explicitly here.

For a linear ordering σ of the nodes of an undirected graph G and a node x of G, let $\sigma[1..x]$ be the restriction of σ to the nodes preceeding x (that is, to $\{y \mid y \leq_\sigma x\}$). We write $x \rightsquigarrow_{G,\sigma} y$ if x and y are connected in $G[\sigma[1..x]]$. Note that $\rightsquigarrow_{G,\sigma}$ is a partial order on $V(G)$.

▶ **Definition 1.** *Let σ be a linear order of the nodes of a graph G and let $v \in V(G)$. Then,*

$$\mathrm{ZW}^\sigma_v := \{u >_\sigma v \mid \exists_{w \in \sigma[1..v]} uw \in E(G) \wedge v \rightsquigarrow_{G,\sigma} w\} \quad \text{and} \quad \mathrm{zw}^\sigma_v := |\,\mathrm{ZW}^\sigma_v\,|.$$

Further, we abbreviate $\mathrm{zw}(\sigma) := \max_v \mathrm{zw}^\sigma_v$ and $\mathrm{zw}(G) := \min_\sigma \mathrm{zw}(\sigma)$. Further, we call the transitive reduction of the directed graph $(V(G), A^*)$ with $A^* := \{uv \in V(G)^2 \mid u \rightsquigarrow_{G,\sigma} v\}$ the canonical tree Γ^σ *of σ for G (as it turns out, Γ^σ is a rooted tree, see below).*

In the following, we say that a rooted tree Γ on $V(G)$ *agrees* with a directed or undirected graph G if, for all $uv \in E(G)$ either $u <_\Gamma v$ or $v <_\Gamma u$. We also extend the definition of $\rightsquigarrow_{G,\sigma}$ to such trees by writing $u \rightsquigarrow_{G,\Gamma} v$ if u and v are connected in $G[\Gamma_u]$.

▶ **Definition 2.** *Let G be a graph and let Γ agree with G. For each $v \in V(G)$, we define*

$$\mathrm{YW}^\Gamma_v := \{u >_\Gamma v \mid \exists_{w \leq_\Gamma v} uw \in E(G)\} \quad \text{and} \quad \mathrm{yw}^\Gamma_v := |\,\mathrm{YW}^\Gamma_v\,|$$

(see Figure 2). Then, we abbreviate $\mathrm{yw}(\Gamma) := \max_v \mathrm{yw}^\Gamma_v$ and $\mathrm{yw}(G) := \min_\Gamma \mathrm{yw}(\Gamma)$.

▶ **Lemma 3.** *Let Γ and Γ' be rooted trees agreeing with an undirected graph G and let $\leq_{\Gamma'}$ be a subset of $\leq_\Gamma$, that is, $x \leq_{\Gamma'} y \Rightarrow x \leq_\Gamma y$ for all $x, y \in V(G)$. Then, $\mathrm{yw}(\Gamma') \leq \mathrm{yw}(\Gamma)$.*

Proof. Let $x \in V(G)$ and let $y \in \mathrm{YW}^{\Gamma'}_x$, that is, $y >_{\Gamma'} x$ and there is some $z \leq_{\Gamma'} x$ with $yz \in E(G)$. Since $\leq_\Gamma$ is a superset of $\leq_{\Gamma'}$, we have $y >_\Gamma x \geq z$, implying $y \in \mathrm{YW}^\Gamma_x$. ◀

▶ **Lemma 4.** *Let σ be a linear order of the nodes of an undirected, connected graph G and let Γ^σ be its canonical tree. Then,*
(a) for each u and v with $v \leq_{\Gamma^\sigma} u$, we have $v \leq_\sigma u$,
(b) for each $u, v \in V(G)$, we have $v \leq_{\Gamma^\sigma} u$ if and only if $u \rightsquigarrow_{G,\sigma} v$,
(c) Γ^σ is connected,
(d) Γ^σ is rooted at the last vertex r of σ,
(e) Γ^σ is a tree,
(f) for all $uv \in E(G)$ with $v <_\sigma u$, we have $v <_{\Gamma^\sigma} u$,
(g) Γ^σ agrees with G, and
(h) $\mathrm{YW}_x^{\Gamma^\sigma} = \mathrm{ZW}_x^\sigma$ for all $x \in V(G)$.

▶ **Observation 5.** *Let Γ be a tree, let Γ' be a contraction of Γ, and let $x, y \in \Gamma'$ be distinct. Then, $x <_{\Gamma'} y$ if and only if $x <_\Gamma y$.*

For the following lemmas, it makes sense to "normalize" some aspects of the structure of agreeing trees. To this end, for a rooted tree T and for $X \subset V(T)$ that does not contain the root r of T, we let $T \uparrow X$ denote the result of (1) replacing each arc uv with $uv \cap X = \{u\}$ with the arc wv where w is the lowest ancestor of u that is not in X, and (2) removing all nodes in X from T. Note that $T \uparrow X$ may have strictly larger out-degree than T, but does not create new ancestor-descendant relations.

▶ **Observation 6.** *Let T be a tree, let $X \subseteq V(T)$ not contain its root, and let $u \leq_{T\uparrow X} v$. Then, $u \leq_T v$.*

▶ **Lemma 7.** *Let Γ be a rooted tree agreeing with an undirected graph G. There is some Γ^* agreeing with G such that $\mathrm{yw}(\Gamma^*) \leq \mathrm{yw}(\Gamma)$ and, for all $u, v \in V(G)$ with $v \leq_{\Gamma^*} u$, we have $u \rightsquigarrow_{G,\Gamma^*} v$.*

▶ **Lemma 8.** *Let Γ be a tree agreeing with a graph G and let p be a non-empty path in G. Then, p contains a unique maximum u with respect to Γ, that is, $v \leq_\Gamma u$ for all vertices v of p.*

Proof. Let x on p be maximal with respect to Γ (that is, for all z on p, we have $x \not\leq_\Gamma z$) and assume towards a contradiction that there is another vertex $y \neq x$ on p that is maximal w.r.t. Γ. Without loss of generality, let x precede y in p and let p_{xy} denote the unique x-y-subpath of p. Since $y \not\leq_\Gamma x$, there is an edge $st \in E(G)$ on p_{xy} with $s \leq_\Gamma x$ and $t \not\leq_\Gamma x$. Hence, $t \not\leq_\Gamma s$. Further, $s \not\leq_\Gamma t$ since, otherwise, the unique t-s-path in Γ contains x, contradicting its maximality. But then Γ does not agree with G. ◀

▶ **Lemma 9.** *Let G be a graph. Then, $\mathrm{zw}(G) = \mathrm{yw}(G)$.*

▶ **Definition 10.** *Let G be a graph and let T be a rooted tree whose vertices are associated to subsets of $V(G)$ by a function $B : V(T) \to 2^{V(G)}$ such that*
(a) for each $uv \in E(G)$, there is some $x \in V(T)$ with $uv \subseteq B(x)$ and
(b) for each $v \in V(G)$, the nodes $x \in V(T)$ with $v \in B(x)$ are weakly connected in T.
We call (T, B) a tree decomposition *of G and its* width *is $\mathrm{tw}(T, B) := \max_{x \in V(T)} \mathrm{tw}_x^{T,B}$ with $\mathrm{tw}_x^{T,B} := |B(x)| - 1$. We call $\mathrm{tw}(G) := \min_{T,B} \mathrm{tw}(T, B)$ the* treewidth *of G. We call (T, B)* nice *if T is binary and all $x \in V(T)$ fall into one of the following categories*
"leaf": *x is a leaf of T and $B(x) = \varnothing$,*
"root": *x is the root of T and $B(x) = \varnothing$,*
"introduce v": *x has a single child y in T and $B(y) = B(x) - v$,*
"forget v": *x has a single child y in T and $B(x) = B(y) - v$,*
"join": *x has two children y and z and $B(x) = B(y) = B(z)$.*

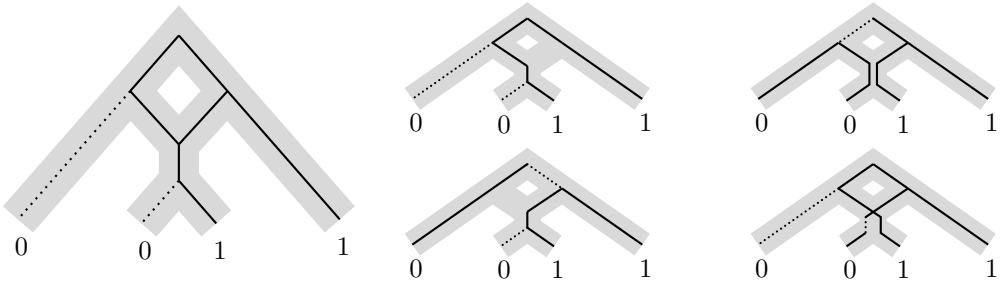


Figure 3 Example for parsimony scores of a network (in gray). Black edges participate in the score (solid = score 0, dotted = score 1). For the hardwired score (left), all edges of the network are considered. For the softwired score (2 possible trees: middle), only edges of any switching are considered. For the parental score (4 possible trees: middle & right), a tree is inscribed in the network.

All graphs G have a nice tree decomposition with $|V(T)| \in O(\mathrm{tw}(G) \cdot |G|)$ and width $\mathrm{tw}(G)$ [23]. Further, since all bags of (T, B) containing a vertex v of G are connected, we can observe the following.

▶ **Observation 11.** *Let (T, B) be a nice tree decomposition for an undirected graph G and let $v \in V(G)$. Then, T contains a single "forget v"-node x and $y <_T x$ for all y with $v \in B(y)$.*

▶ **Proposition 12.** *Let G be a graph. Then, $\mathrm{yw}(G) = \mathrm{tw}(G)$. Further, given a tree decomposition (T, B) for G, we can compute a tree Γ agreeing with G such that $\mathrm{yw}(\Gamma) = \mathrm{tw}(T, B)$ in linear time.*

4 Parsimony

Given states of a character, observed in extant species, as well as a species phylogeny, the small parsimony problem asks to infer states of the same character for all ancestral species such as to minimize the "parsimony score" of this assignment. This problem comes in three flavors called "hardwired", "softwired", and "parental" parsimony. Throughout this section, let C be a fixed finite set (a "character"). For convenient use of the $\trianglelefteq$-relation, let C be an anti-chain (that is, for each $x, y \in C$, we have $x \leq y$ only if $x = y$). Formally, for a phylogeny N and a function $\phi : V(N) \to 2^C$, we define the hardwired and softwired parsimony score as

$$\mathrm{par}_N^H(\phi) := \min_{\psi : V(N) \to C,\ \psi \trianglelefteq \phi} \sum_{uv \in A(N)} \delta_\psi(u, v) \qquad \mathrm{par}_N^S(\phi) := \min_{\substack{\psi : V(N) \to C,\ \psi \trianglelefteq \phi \\ T \in \mathcal{S}(N)}} \sum_{uv \in A(T)} \delta_\psi(u, v).$$

The "parental parsimony" is defined using "parental trees" but, in this work, we use the equivalent formulation using lineage functions [29].

▶ **Definition 13.** *A* lineage function *for a phylogeny N is any function $f : V(N) \to 2^C$. The* cost *of f is* $\mathrm{cost}(f) := \sum_{v \in V(N)} \mathrm{cost}_f(v)$ *where*

$$\mathrm{cost}_f(v) := \left| f(v) \setminus \bigcup_{u \in \mathrm{Pred}(v)} f(u) \right| + \begin{cases} -1 & \text{if } v = \rho_N \text{ and } |f(v)| = 1 \\ 0 & \text{if } v \neq \rho_N \text{ and } |f(v)| \leq \sum_{u \in \mathrm{Pred}(v)} |f(u)| \\ \infty & \text{otherwise} \end{cases}$$

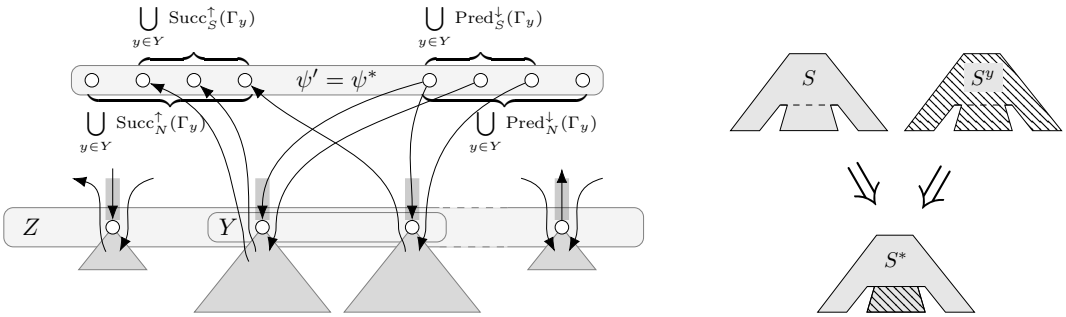


Figure 4 Lemma 14 proves that any solution (S, ψ) that is optimal on sub-trees rooted at Z in Γ must also be optimal (among all solutions with ψ's behavior on $\bigcup_{y\in Y} \mathrm{YW}^{\Gamma}_y$ (gray box on top)) on all sub-trees of Γ that are rooted below Z (at Y). That is, no solution (S^y, ψ_y) can be better than (S, ψ) on the sub-network induced by Γ_y for any $y \in Y$. To prove this, a new solution (S^*, ψ^*) is constructed by replacing the sub-solution of (S, ψ) below Y by the sub-solutions (S^y, ψ_y) below Y.

Given N and a function $\phi : V(N) \to 2^C$, we denote the set of all lineage functions f on N with $f \trianglelefteq \phi$ as $\mathcal{LF}_{N,\phi}$. Finally, the parental parsimony score *is*

$$\mathrm{par}^P_N(\phi) := \min_{f\in\mathcal{LF}_{N,\phi}} \mathrm{cost}(f) \tag{1}$$

For each of the presented variants, we give a dynamic programming formulation using a given tree Γ that agrees with the undirected graph G underlying the input network and corresponds to Lemma 7, that is, each non-leaf x of Γ has a child v with $x \in \mathrm{YW}^{\Gamma}_v$. The running time of the resulting algorithm will depend on the width $\mathrm{yw}(\Gamma)$ of Γ (recalling that $\mathrm{yw}(\Gamma)$ coincides with the treewidth of G for optimal Γ).

As stated in the introduction, in this paper we focus on the case of analyzing a specific position in the genome. Since the function ϕ can associate several states to a same leaf, our definition permits to describe polymorphism in a population. While, in our current formulation, the algorithms "choose" an optimal state to associate to each leaf, the parental parsimony can be easily modified to explain *all* states of each leaf at the end of the run. This allows keeping the information on polymorphism in all steps of the algorithm (see Section 4.3). Note also that ϕ can associate information to internal nodes, thus permitting the user to impose restrictions on the states associated to ancestral species.

In the presentation of the dynamic programming, a table entry $Q^y_x[z]$ means that x and y are considered fix for this table and z is a variable index. Further, tables $Q^{y_1}_{x_1}$ and $Q^{y_2}_{x_2}$ are independent of one another, allowing an implementation to forget $Q^{y_1}_{x_1}$ if it is no longer needed, even if $Q^{y_2}_{x_2}$ still is. In the following, for an anti-chain Y in Γ and a class $\mathcal{G}$ of subnetworks of N, a *Y-substitution system* of $\mathcal{G}$ is a series of subnetworks $(N^y)_{y\in Y}$ of N such that, for all $N' \in \mathcal{G}$, the digraph $(V(N), (A(N') \setminus \bigcup_{y\in Y} A_y(N')) \cup \bigcup_{y\in Y} A_y(N^y))$ is also in $\mathcal{G}$. Roughly, we can "swap out" the arcs in $A_y(N')$ for $A_y(N^y)$ for each $y \in Y$ without loosing membership in $\mathcal{G}$. Note that the N^y are not necessarily distinct, so a trivial Y-substitution system for $\{N'\}$ would be $(N')_{y\in Y}$. The formulations are based on the following lemma about independent sub-solutions, showing that an optimal solution (S, ψ) for a sub-network (of G) "below" an anti-chain Z in Γ is also optimal on any sub-network "below" an anti-chain Y in Γ that is itself "below" Z (among all solutions with ψ's behavior on $\bigcup_{y\in Y} \mathrm{YW}^{\Gamma}_y$).

▶ **Lemma 14** (see Figure 4). *Let $Y, Z \subseteq V(N)$ be anti-chains in Γ such that $Y \subseteq \bigcup_{z\in Z} \Gamma_z$. Let $\mathcal{G}$ be a class of subnetworks of N and let $S \in \mathcal{G}$ and $\psi : V(N) \to C$ such that (a) $\sum_{z\in Z} \sum_{uw\in A_z(S)} \delta_\psi(u, w)$ is minimum among all such S and ψ. Let $(S^y)_{y\in Y}$ be a*

Y-substitution system for $\mathcal{G}$ and let $\psi_y : V(N) \to C$ for each $y \in Y$ such that (b) ψ_y and ψ coincide on YW_y^Γ. Then,

$$\sum_{y \in Y} \sum_{uw \in A_y(S^y)} \delta_{\psi_y}(u,w) \geq \sum_{y \in Y} \sum_{uw \in A_y(S)} \delta_\psi(u,w).$$

4.1 Hardwired Parsimony

To compute the hardwired parsimony score at a node v of N, we require knowledge of the character assigned to v and its neighbors. For all $u \in \mathrm{YW}_v^\Gamma$, we thus "guess" the character $\psi(u)$ assigned to u by an optimal assignment. In our dynamic programming, we scan Γ bottom-up, computing a table entry $T^{\mathcal{HW}}[x,\psi]$ for each $x \in V(\Gamma) = V(N)$ and each $\psi : \mathrm{YW}_x^\Gamma \to C$, containing the parsimony cost incurred by all arcs in $A_x(N)$, assuming that all nodes in YW_x^Γ receive their characters according to ψ. Note that $A_x(N) = \bigcup_i A_{v_i}(N) \cup A_{\{x\}}(N)$, where the v_i are the children of x in Γ. Thus, $T^{\mathcal{HW}}[x,\psi]$ can be calculated as follows.

▶ **Definition 15.** *Let Γ be a tree that agrees with N, let $x \in V(N)$ and let $\psi_x : \mathrm{YW}_x^\Gamma \to C$ with $\psi_x \trianglelefteq \phi$. Let $v_1, v_2, \ldots, v_t$ denote the children of x in Γ ($t = 0$ if x is a leaf). Then, we define a table entry*

$$T^{\mathcal{HW}}[x,\psi_x] := \min_{c_x \in \phi(x)} \left(\sum_{1 \leq i \leq t} T^{\mathcal{HW}}\left[v_i, \psi_x\,[x \to c_x]\,|_{\mathrm{YW}_{v_i}^\Gamma}\right] + \sum_{z \in \mathrm{Pred}_N^\downarrow(x) \cup \mathrm{Succ}_N^\uparrow(x)} \delta(c_x, \psi_x(z)) \right) \quad (2)$$

▶ **Lemma 16.** *Let $x \in V(N)$ and let $\psi_x : \mathrm{YW}_x^\Gamma \to C$ with $\psi_x \trianglelefteq \phi$. Let $\psi : V(N) \to C$ with $\psi_x \trianglelefteq \psi \trianglelefteq \phi$ such that ψ minimizes $\sum_{uw \in A_x(N)} \delta_\psi(u,w)$. Then,*

$$T^{\mathcal{HW}}[x,\psi_x] = \sum_{uw \in A_x(N)} \delta_\psi(u,w)$$

Proof Sketch. For "$\geq$", we construct a mapping ψ' from mappings ψ_i that are optimal on $A_{v_i}(N)$ among all mappings with $\psi_i(x) := c_x$. This is possible since all such ψ_i coincide with ψ' and ψ_x on YW_x^Γ. By induction hypothesis, the cost of ψ' on $A_x(N)$ is $\sum_{1 \leq i \leq t} T^{\mathcal{HW}}\left[v_i, \psi'\,|_{\mathrm{YW}_{v_i}^\Gamma}\right] + \sum_{uw \in A_{\{x\}}(N)} \delta_{\psi'}(u,w)$. Then, "$\geq$" follows from optimality of ψ on $A_x(N)$.

For "$\leq$", it suffices to show that the cost of ψ on $A_x(N)$ is equal to the result of setting $c_x := \psi(x)$ in the right hand side of (2) (which is a valid choice for the minimum since $\psi(x) \in \phi(x)$). First, the cost of ψ on $A_{v_i}(N)$ is $T^{\mathcal{HW}}\left[v_i, \psi\,|_{\mathrm{YW}_{v_i}^\Gamma}\right]$ by independence of sub-solutions and the induction hypothesis. Second, the cost of ψ on $A_{\{x\}}^\downarrow(N)$ is $\sum_{z \in \mathrm{Pred}_N^\downarrow(x)} \delta(c_x, \psi_x(z))$ and the cost of ψ on $A_{\{x\}}^\uparrow(N)$ is $\sum_{z \in \mathrm{Succ}_N^\uparrow(x)} \delta(c_x, \psi_x(z))$ since ψ and ψ_x coincide on YW_x^Γ. ◀

In order to solve the hardwired parsimony problem given N, ϕ and Γ, all we have to do is compute $T^{\mathcal{HW}}[x,\psi_x]$ for each x bottom-up in Γ and each of the (at most) $|C|^{|\,\mathrm{YW}_x^\Gamma\,|}$ many choices of $\psi_x : \mathrm{YW}_x^\Gamma \to C$ with $\psi_x \trianglelefteq \phi$. Then, by Lemma 16, the hardwired parsimony score of N with respect to ϕ can be read from $T^{\mathcal{HW}}[\rho_\Gamma, \varnothing]$. To compute $T^{\mathcal{HW}}$, the sum over the children of x for all $x \in V(N)$ in (2) can be computed in amortized $O(|A(N)|)$ time and, with a bit of bookkeeping, it is possible to maintain the value of the second sum in (2) in $O(|A(N)|)$ amortized time per choice of ψ. Then the following holds:

▶ **Theorem 17.** *Given a network N, some $\phi : V(N) \to 2^C$ and a tree Γ agreeing with N, the hardwired parsimony score of (N,ϕ) can be computed in $O(|C|^{\mathrm{yw}(\Gamma)+1} \cdot |A(N)|)$ time.*

Proposition 12 lets us turn tree decompositions of N into trees Γ agreeing with N, allowing us to replace $\mathrm{yw}(\Gamma)$ by $\mathrm{tw}(N)$, incurring an additional running time of $|N| \cdot 2^{O(\mathrm{tw}(N)^3)}$ [4].

▶ **Corollary 18.** *Let (N, ϕ) be an instance of* Hardwired Parsimony. *Let $t \geq \mathrm{tw}(N)$ and let T be the time in which a width-t tree decomposition of N can be computed. Then, the hardwired parsimony score of (N, ϕ) can be computed in $O(T + |C|^{t+1} \cdot |A(N)|)$ time.*

4.2 Softwired Parsimony

In contrast to the hardwired parsimony score, where the computation of the cost of the incident edges of a node x only required knowledge of the characters assigned to neighbors of x, computing the *softwired* score additionally requires knowledge of which parent of x remains a parent in the sought switching. A table entry $T^{\mathcal{SW}}[x, \ldots]$ contains the smallest combined cost of all arcs in $A_x(S)$ for a switching S of N minimizing this cost. To be able to compute an entry for $x \in V(N)$, we not only need to "guess" ψ_x but, additionally, some representation of the switching S. In particular, in S, no child of x may have another parent than x. However, since children of x in N may be above x in Γ, we have to "guess" which children of x in N are still children of x in S. Such a guess manifests itself as an additional index R^x of the dynamic programming table (note that we clearly only have to store this information for children of x that are reticulations). Indeed, this information has to be stored for all nodes considered below x who still have children in YW_x^Γ. Thus, we index our DP-table also by a subset $R^x \subseteq \mathrm{YW}_x^\Gamma \cap R(N)$ containing a reticulation $r \in R(N)$ if and only if Γ_x contains a parent v of r and vr is an arc of an optimal switching S for $N[\Gamma_x \cup \mathrm{YW}_x^\Gamma]$.

▶ **Definition 19.** *Let Γ be a tree that agrees with N, let $x \in V(N)$, let $\psi_x : \mathrm{YW}_x^\Gamma \to C$ with $\psi_x \trianglelefteq \phi$, and let $R^x \subseteq \mathrm{Succ}_N^{R\uparrow}(\Gamma_x)$. Let $v_1, v_2, \ldots, v_t$ denote the children of x in Γ ($t = 0$ if x is a leaf in Γ). Then, set*

$$T^{\mathcal{SW}}[x, \psi_x, R^x] := \min_{c_x \in \phi(x)} \min_{R^* \subseteq R^x \cap \mathrm{Succ}_N^{R\uparrow}(x)} \sum_{r \in R^* \cup \mathrm{Succ}_N^{T\uparrow}(x)} \delta(c_x, \psi_x(r)) + \min \begin{cases} Q^{\psi_x}_{x,c_x}[t, R^x \setminus R^*] + \min\limits_{y \in \mathrm{Pred}_N^{\downarrow}(x)} \delta(c_x, \psi_x(y)) & \text{if } \mathrm{Pred}_N^{\downarrow}(x) \neq \varnothing \\ Q^{\psi_x}_{x,c_x}[t, (R^x \setminus R^*) \cup (\{x\} \cap R(N))] & \text{if } \mathrm{Pred}_N^{\uparrow}(x) \neq \varnothing \end{cases} \tag{3}$$

where

$$Q^{\psi_x}_{x,c_x}[i, R'] := \begin{cases} \min\limits_{R^* \subseteq R' \cap \mathrm{Succ}_N^{R\uparrow}(\Gamma_{v_i})} Q^{\psi_x}_{x,c_x}[i-1, R' \setminus R^*] + T^{\mathcal{SW}}[v_i, \psi_i, R^*] & \text{if } i \neq 0 \\ 0 & \text{if } i = 0 \text{ and } R' = \varnothing \\ \infty & \text{otherwise} \end{cases} \tag{4}$$

where $\psi_i := \psi_x[x \to c_x]|_{\mathrm{YW}_{v_i}^\Gamma}$ for all $i \leq t$. (Note how $Q^{\psi_x}_{x,c_x}[i, R']$ is used to assign the nodes in R^x to the v_i (with $v_0 = x$) such that every node in R^x has a parent in some Γ_{v_i}).

In the following, for any anti-chain$X$ in Γ and all $Z \subseteq \bigcup_{x \in X} \mathrm{YW}_x^\Gamma$, let $\mathcal{S}^{X \to Z}(N)$ denote the set of all switchings S of N with $\mathrm{Succ}_S^{\mathrm{R}\uparrow}(X) = Z$.

▶ **Lemma 20.** *Let Γ be a tree that agrees with N, let $x \in V(N)$, let $\psi_x : \mathrm{YW}_x^\Gamma \to C$ with $\psi_x \trianglelefteq \phi$, and let $R^x \subseteq \mathrm{Succ}_N^{R\uparrow}(\Gamma_x)$. If $\mathcal{S}^{\Gamma_x \to R^x}(N) = \varnothing$, then $T^{\mathcal{SW}}[x, \psi_x, R^x] = \infty$. Otherwise, let $S \in \mathcal{S}^{\Gamma_x \to R^x}(N)$ and $\psi : V(N) \to C$ such that (a) $\psi_x \trianglelefteq \psi \trianglelefteq \phi$ and (b) $\sum_{uw \in A_x(S)} \delta_\psi(u, w)$ is minimum among all such S and ψ. Then,*

$$T^{\mathcal{SW}}[x, \psi_x, R^x] = \sum_{uw \in A_x(S)} \delta_\psi(u, w). \tag{5}$$

Proof Sketch. Let us abbreviate $Z_i := \bigcup_{j \leq i} V(\Gamma_{v_j})$. We first show that the table Q does what we expect it to do.

▷ Claim 21. $Q^{\psi_x}_{x,c_x}[i, R'] = \sum_{j \leq i} \sum_{uw \in A_{v_j}(S_i)} \delta_{\psi_i}(u, w)$ for optimal $S_i \in \mathcal{S}^{Z_i \to R'}$ and ψ_i coincides with $\psi_x[x \to c_x]$ on $\bigcup_{j \leq i} \mathrm{YW}^{\Gamma}_{v_j}$.

Proof Sketch. For "$\geq$", let $R^* \subseteq R' \cap \mathrm{Succ}^{\mathrm{R}\uparrow}_N(\Gamma_{v_i})$ such that equality holds in (4). We consider a switching $S' \in \mathcal{S}^{Z_i \to R'}$ constructed from switchings $S_{i-1} \in \mathcal{S}^{Z_{i-1} \to R' \setminus R^*}$ and $S^* \in \mathcal{S}^{\Gamma_{v_i} \to R^*}$ as well as a mapping ψ' coinciding with $\psi_x[x \to c_x]$ on $\bigcup_{j<i} \mathrm{YW}^{\Gamma}_{v_j}$ constructed from mappings ψ_{i-1} and ψ^* such that (a) ψ_{i-1} coincides with $\psi_x[x \to c_x]$ on $\bigcup_{j<i} \mathrm{YW}^{\Gamma}_{v_j}$, (b) ψ^* coincides with $\psi_x[x \to c_x]$ on $\mathrm{YW}^{\Gamma}_{v_i}$, (c) the cost of ψ_{i-1} is optimal on $A_{Z_{i-1}}(S_{i-1})$ and (d) the cost of ψ^* is optimal on $A_{v_i}(S^*)$. By induction hypotheses, these costs are $Q^{\psi_x}_{x,c_x}[i-1, R' \setminus R^*]$ and $T^{\mathcal{SW}}[v_i, \psi_x[x \to c_x], R^*]$, respectively. Then, "$\geq$" follows by optimality of S_i and ϕ_i.

For "$\leq$", we let $R^* := \mathrm{Succ}^{\mathrm{R}\uparrow}_{S_i}(\Gamma_{v_i})$ and use independence of sub-solutions and the induction hypotheses to show that the cost of ϕ_i on $A_{Z_{i-1}}(S_i)$ is $Q^{\psi_x}_{x,c_x}[i-1, R' \setminus R^*]$ and the cost of ϕ_i on $A_{v_i}(S_i)$ is $T^{\mathcal{SW}}[v_i, \phi_i, R^*]$. Then, "$\leq$" follows from the fact that R^* is only one of the possible choices for the minimum in (4). ■

For "$\geq$", let $c_x \in \phi(x)$ and $R^* \subseteq R^x \cap \mathrm{Succ}^{\mathrm{R}\uparrow}_N(x)$ be such that equality holds in (3). We consider a switching $S' \in \mathcal{S}^{\Gamma_x \to R^x}$ constructed from switchings S_t and S^* with $S_t \in \mathcal{S}^{Z_t \to R^x \setminus R^*}$ (if $\mathrm{Pred}^{\downarrow}_N(x) \neq \varnothing$) or $S_t \in \mathcal{S}^{Z_t \to (R^x \setminus R^*) \cup \{x\}}$ (if $x \in R(N)$ and $\mathrm{Pred}^{\uparrow}_N(x) \neq \varnothing$), and $S^* \in \mathcal{S}^{\{x\} \to R^*}$, as well as a mapping ψ' coinciding with ψ_x on YW^{Γ}_x constructed from mappings ψ_t and ψ^* such that **1.** ψ_t coincides with $\psi_x[x \to c_x]$ on $\bigcup_{i \leq t} \mathrm{YW}^{\Gamma}_{v_i}$, **2.** ψ^* coincides with ψ_x on YW^{Γ}_x, **3.** $\psi^*(x) = c_x$, **4.** the cost of ψ_t is optimal on $A_{Z_t}(S_t)$ and **5.** the cost of ψ^* is optimal on $A_{\{x\}}(S^*)$. Then, the cost of ψ^* on $A^{\uparrow}_{\{x\}}(S^*)$ is $\sum_{r \in R^* \cup \mathrm{Succ}^{\mathrm{T}\uparrow}_N(x)} \delta(c_x, \psi_x(r))$, the cost of ψ^* on $A^{\downarrow}_{\{x\}}(S^*)$ is $\min_{y \in \mathrm{Pred}^{\downarrow}_N(x)} \delta(c_x, \psi_x(y))$ if the parent of x in S_t is above x in Γ (that is, $x \notin \mathrm{Succ}^{\mathrm{R}\uparrow}_{S_t}(Z_t)$) and, by the claim above, the cost of ψ_t on $A_{Z_t}(S_t)$ is $Q^{\psi_x}_{x,c_x}\left[t, \mathrm{Succ}^{\mathrm{R}\uparrow}_{S_t}(Z_t)\right]$. Then, as $S' \in \mathcal{S}^{\Gamma_x \to R^x}$, "$\geq$" follows by optimality of S and ϕ.

For "$\leq$", let $c_x := \phi(x)$ and let $R^* := \mathrm{Succ}^{\mathrm{R}\uparrow}_S(\Gamma_x)$. We use independence of sub-solutions and the induction hypothesis to show that the cost of ϕ on $A_{Z_t}(S)$ is $Q^{\psi_x}_{x,c_x}[t, R' \setminus R^*]$ (if $x \notin R(N)$ or the parent of x in S is above x in Γ) or $Q^{\psi_x}_{x,c_x}[t, (R' \setminus R^*) \cup \{x\}]$ (if $x \in R(N)$ and the parent of x in S is in Γ_x). Further, the cost of ψ on $A^{\uparrow}_{\{x\}}(S)$ is $\sum_{r \in R^* \cup \mathrm{Succ}^{\mathrm{T}\uparrow}_N(x)} \delta(c_x, \psi_x(r))$, the cost of ψ on $A^{\downarrow}_{\{x\}}(S)$ is $\min_{y \in \mathrm{Pred}^{\downarrow}_N(x)} \delta(c_x, \psi_x(y))$ if the parent of x in S is above x in Γ. Then, "$\leq$" follows from the fact that our choices of c_x and R^* are only one of the possible choices for the minimum in (3). ◀

In order to solve the softwired parsimony problem given N, ϕ and Γ, all we have to do is compute $T^{\mathcal{SW}}[x, \psi_x, R^x]$ for each x bottom-up in Γ, each of the (at most) $|C|^{|\mathrm{YW}^{\Gamma}_x|}$ many choices of $\psi_x : \mathrm{YW}^{\Gamma}_x \to C$ with $\psi_x \trianglelefteq \phi$, and each $R^x \subseteq \mathrm{Succ}^{\mathrm{R}\uparrow}_N(x) \subseteq \mathrm{YW}^{\Gamma}_x \cap R(N)$. To this end, $Q^{\psi_x}_{x,c_x}[i, R^x \setminus R^*]$ and $Q^{\psi_x}_{x,c_x}[i, (R^x \setminus R^*) \cup \{x\}]$ have to be computed for each child v_i of x in Γ and each $R^* \subseteq R^x \cap \mathrm{Succ}^{\mathrm{R}\uparrow}_N(x)$. Then, by Lemma 20, the softwired parsimony score of N with respect to ϕ can be read from $T^{\mathcal{SW}}[\rho_\Gamma, \varnothing, \varnothing]$. In the following, let ψ_x be fix. Then, for fix c_x, we can compute $Q^{\psi_x}_{x,c_x}[i, R']$ for all choices of x, i and R' in $O(2^{|R' \cap \mathrm{Succ}^{\mathrm{R}\uparrow}_N(v_i)|} + \sum_{x \in \Gamma} |\mathrm{Succ}_\Gamma(x)|) \subseteq O(2^{|\mathrm{YW}^{\Gamma}_x|+1} + |\Gamma|)$ time total. Further, the values of $\min_{y \in \mathrm{Pred}^{\downarrow}_N(x)} \delta(c_x, \phi_x(y))$ can be pre-computed for all $x \in \Gamma$ in $O(|A(N)|)$ time total. Then, to compute $T^{\mathcal{SW}}[x, \psi_x, R^x]$ for all x and R^x, we have to check $|V(N)|$ choices for x, as well as $|\phi(x)| \leq |C|$ choices for c_x and $3^{|\mathrm{Succ}^{\mathrm{R}\uparrow}_N(x)|}$ choices for R^x and $R^* \subseteq R^x$ combined. Altogether,

the table $T^{\mathcal{SW}}$ can be computed in $O(|C|^{|\operatorname{YW}_x^\Gamma|} \cdot (3^{|\operatorname{YW}_x^\Gamma|} \cdot |C| \cdot |V(N)| + |A(N)|))$ time. The computation of $Q^{\psi_x}_{x,c_x}$ in $O(2^{|\operatorname{YW}_x^\Gamma|} + |A(N)|)$ time is absorbed by this. For practical purposes, note that estimating $|\operatorname{Succ}_N^{\mathrm{R}\uparrow}(x)| \leq |\operatorname{YW}_x^\Gamma|$ is quite crude and equality will almost never be attained. Then, the following result holds:

▶ **Theorem 22.** *Given a network N, $\phi : V(N) \to 2^C$ and a tree Γ agreeing with N, the softwired parsimony score of (N, ϕ) can be computed in $O(|C|^{\operatorname{yw}(\Gamma)} \cdot (3^{\operatorname{yw}(\Gamma)} \cdot |C| \cdot |V(N)| + |A(N)|))$ time.*

Again, we can replace $\operatorname{yw}(\Gamma)$ by $\operatorname{tw}(N)$ using Proposition 12.

▶ **Corollary 23.** *Let (N, ϕ) be an instance of Softwired Parsimony. Let $t \geq \operatorname{tw}(N)$ and let T be the time in which a width-t tree decomposition of N can be computed. Then, the softwired parsimony score of (N, ϕ) can be computed in $O(T + |C|^t \cdot (3^t \cdot |C| \cdot |V(N)| + |A(N)|))$ time.*

4.3 Parental Parsimony

For ease of presentation, we introduce some additional notation. First, for any a and b, we abbreviate $\max\{a - b, 0\} =: a \dot{-} b$. Let ψ and ψ' be functions with the same codomain. If ψ maps all items to $\varnothing$ or to 0, then we say that ψ is a *zero-function* and we write $\psi = \overrightarrow{0}$. We use $\psi - \psi'$ to denote the function defined on the domain of ψ for which $(\psi - \psi')(x) = \psi(x)$ if $\psi'(x) = \perp$ and $(\psi - \psi')(x) = \psi(x) - \psi'(x)$, otherwise. This definition extends to functions mapping to sets in a natural way.

Each lineage function gives rise to one or more phylogenetic trees, called *lineages*, embedded in N. For each $x \in V(N)$, $f(x)$ represents the set of branches of such a lineage passing through x. Each such lineage-branch may "choose" a parent among the parents of x in N. This models the biological circumstance that a character trait may be inherited from any parent. We compute (the cost of) an optimal lineage function on N using a tree Γ that agrees with N. To compute $\operatorname{cost}_f(x)$, we require knowledge of $\sum_{y \in \operatorname{Pred}(x)} |f(y)|$ as well as $\bigcup_{y \in \operatorname{Pred}(x)} f(y)$. For all $y \in \operatorname{YW}_x^\Gamma$, we thus store the set $\lambda(y) := f(y)$ of lineages in y, the subset $\psi(y)$ of lineages of y that also occur in parents (in N) of y that are below x in Γ, that is, $\operatorname{Pred}_N^{\uparrow x}(y)$ (such lineages are inherited by y at no cost), and the total number $\eta(y)$ of lineages of y that can be inherited from parents (in N) of y that are below x in Γ, that is, $\operatorname{Pred}_N^{\uparrow x}(y)$ (cost 0 or 1). Then, $\sum_{y \in \operatorname{Pred}_N(x)} |f(y)| = \eta(x) + \sum_{y \in \operatorname{Pred}_N^\downarrow(x)} |\lambda(y)|$ and $\bigcup_{y \in \operatorname{Pred}_N(x)} f(y) = \psi(x) \cup \bigcup_{y \in \operatorname{Pred}_N^\downarrow(x)} \lambda(y)$.

In order to compute an entry $T^{\mathcal{PT}}[x, \lambda_x, \psi_x, \eta_x]$, we "guess" the set $U \subseteq \phi(x)$ of lineages passing through x in an optimal solution, as well as the set $D \subseteq U$ of lineages inherited from nodes in $\operatorname{Pred}_N^\uparrow(x)$. Then, the cost incurred by x is the number of lineages of x that are not lineages of any $r \in \operatorname{Pred}_N(x)$, that is, the number of lineages in $U \setminus (D \cup \bigcup_{r \in \operatorname{Pred}_N^\downarrow(x)} \lambda(r))$. For the recursive table lookup, we have to make sure that $\lambda(x) = U$, $\psi(x) = D$, and that all lineage branches of x that do not come from $\operatorname{Pred}_N^\downarrow(x)$ can be inherited from $\operatorname{Pred}_N^\uparrow(x)$, that is, $\eta(x) = |\lambda(x)| \dot{-} \sum_{r \in \operatorname{Pred}_N^\downarrow(x)} |\lambda(r)|$. Further, each child y of x in N may inherit a lineage from x and, if y is above x in Γ, this has to be registered by removing the lineages of U from $\psi(y)$ and subtracting $|U|$ from $\eta(y)$. Finally, the lineage branches represented by ψ and η are distributed among the children of x in Γ using the table Q. In the following, in order to avoid treating the case that $x = \rho_N$ separately, we define $\rho(x) := 1 - \delta(x, \rho_N)$, that is, $\rho(x) = 1$ if and only if $x = \rho_N$.

▶ **Definition 24.** *Let Γ be a tree that agrees with N, $x \in V(N)$, $\lambda_x : \operatorname{YW}_x^\Gamma \to 2^C$ with $\lambda_x \trianglelefteq \phi$ and $\psi_x \trianglelefteq \lambda_x$. Let $\{v_1, v_2, \ldots, v_t\} = \operatorname{Succ}_\Gamma(x)$ ($t = 0$ if x is a leaf in Γ). Then, set*

$T^{\mathcal{PT}}[x, \lambda_x, \psi_x, \eta_x]$ *to*

$$\min_{\substack{D \subseteq U \subseteq \phi(x) \\ U \neq \emptyset}} Q_x^{\lambda_x[x \to U]} \left[t, \psi_x \left[\begin{smallmatrix} x \to D \\ \forall_{w \in \mathrm{Succ}_N^{\uparrow}(x)} w \to \psi_x(w) \setminus U \end{smallmatrix} \right], \eta_x \left[\begin{smallmatrix} x \to |U| \dot{-} \sum_{u \in \mathrm{Pred}_N^{\downarrow}(x)} |\lambda_x(u)| \\ \forall_{w \in \mathrm{Succ}_N^{\uparrow}(x)} w \to \eta_x(w) \dot{-} |U| \end{smallmatrix} \right] \right] + \left| U \setminus \left(D \cup \bigcup_{u \in \mathrm{Pred}_N^{\downarrow}(x)} \lambda_x(u) \right) \right| \tag{6}$$

where $Q_x^{\lambda}[i, \psi, \eta]$ *equals*

$$\begin{cases} \min\limits_{\psi' \trianglelefteq \psi|_{\mathrm{YW}_{v_i}^{\Gamma}}} \min\limits_{\eta' \trianglelefteq \eta|_{\mathrm{YW}_{v_i}^{\Gamma}}} Q_x^{\lambda}[i-1, \psi - \psi', \eta - \eta'] + T^{\mathcal{PT}}\left[v_i, \lambda|_{\mathrm{YW}_{v_i}^{\Gamma}}, \psi', \eta'\right] & \text{if } i > 0 \\ -\rho(x) & \text{if } i = 0 \text{ and } \psi = \overrightarrow{0} \text{ and } \eta = \overrightarrow{0}[x \to \rho(x)] \\ \infty & \text{otherwise} \end{cases} \tag{7}$$

Note how the table Q_x^{λ} distributes the lineage branches of x whose parents are in Γ_x among the children of x in Γ. Observe that both $T^{\mathcal{PT}}$ and Q_x^{λ} are monotone in ψ and η (wrt. $\trianglelefteq$) by construction.

▶ **Lemma 25.** *Let* $x \in V(N)$*, let* $i \in \mathbb{N}$*, let* $\lambda : \mathrm{YW}_x^{\Gamma} \to 2^C$*, let* $\eta, \eta' : \mathrm{YW}_x^{\Gamma} \to \mathbb{N}$*, and let* $\psi, \psi' : \mathrm{YW}_x^{\Gamma} \to 2^C$ *such that* $\psi' \trianglelefteq \psi \trianglelefteq \lambda$ *and* $\overrightarrow{0}[x \to \rho(x)] \trianglelefteq \eta' \trianglelefteq \eta$*. Then,*

$$T^{\mathcal{PT}}[x, \lambda, \psi', \eta'] \leq T^{\mathcal{PT}}[x, \lambda, \psi, \eta] \qquad \text{and} \qquad Q_x^{\lambda}[i, \psi', \eta'] \leq Q_x^{\lambda}[i, \psi, \eta]$$

Proof Sketch. The lemma can be proved by induction on the height of x in Γ and the value of i. If x is a leaf, then $Q_x^{\lambda}[0, \psi, \eta]$ is finite only if $\psi = \overrightarrow{0}$ and $\eta = \overrightarrow{0}[x \to \rho(x)]$, implying the second inequality. For monotony of $T^{\mathcal{PT}}$, fix the sets $D \subseteq U \subseteq C$ for which the minimum in the formula of $T^{\mathcal{PT}}[x, \lambda, \psi, \eta]$ is attained. Then, by monotony of Q_x^{λ}, replacing ψ by ψ' and η by η' in this formula does not increase its value and this value is at most $T^{\mathcal{PT}}[x, \lambda, \psi', \eta']$ since it is obtained for one of several possible choices for D and U. If x is not a leaf in Γ then monotonicity of $Q_x^{\lambda}[i, \ldots]$ is implied by monotonicity of $Q_x^{\lambda}[i-1, \ldots]$ and monotonicity of $T^{\mathcal{PT}}[v, \ldots]$ for the children v of x. Finally, monotonicity of $T^{\mathcal{PT}}$ follows from monotonicity of Q_x^{λ} as in the induction base. ◀

▶ **Lemma 26.** *Let* Γ *be a tree agreeing with* N*, let* $x \in V(N)$*, let* $\psi_x, \lambda_x : \mathrm{YW}_x^{\Gamma} \to 2^c$ *and* $\eta_x : \mathrm{YW}_x^{\Gamma} \to \mathbb{N}$*. Let* f *minimize* $\mathrm{cost}(f)$ *among all lineage functions in* $\mathcal{LF}_{N,\phi}$ *such that, for all* $w \in \mathrm{YW}_x^{\Gamma}$*,* $\lambda_x(w) = f(w)$*,* $\psi_x(w) = f(w) \cap \bigcup_{u \in \mathrm{Pred}_N^{\uparrow x}(w)} f(u)$*, and* $\eta_x(w) \leq \sum_{u \in \mathrm{Pred}_N^{\uparrow x}(w)} |f(u)|$*. If there are no such* f*, then* $T^{\mathcal{PT}}[x, \lambda_x, \psi_x, \eta_x] = \infty$*. Otherwise,*

$$T^{\mathcal{PT}}[x, \lambda_x, \psi_x, \eta_x] = \sum_{z \leq_{\Gamma} x} \mathrm{cost}_f(z)$$

Proof Sketch. Let us abbreviate $Z_i := \bigcup_{j \leq i} V(\Gamma_{v_j})$. We first show that the table Q does what we expect it to do.

▷ Claim 27. Let $\lambda, \psi : \mathrm{YW}_x^{\Gamma} \cup \{x\} \to 2^C$ and $\eta : \mathrm{YW}_x^{\Gamma} \cup \{x\} \to \mathbb{N}$ such that $\psi \trianglelefteq \lambda \trianglelefteq \phi$. Let $f_i \in \mathcal{LF}_{N,\phi}$ have minimum cost on $\bigcup_{j \leq i} \Gamma_{v_j}$ among all lineage functions for N that, for all $w \in \bigcup_{j \leq i} \mathrm{YW}_{v_j}^{\Gamma}$, satisfy (a) $\lambda(w) = f_i(w)$, (b) $\psi(w) = f_i(w) \cap \bigcup_{j \leq i} \bigcup_{u \in \mathrm{Pred}_N^{\uparrow v_j}(w)} f_i(u)$, and (c) $\eta(w) \leq \sum_{j \leq i} \sum_{u \in \mathrm{Pred}_N^{\uparrow v_j}(w)} |f_i(u)|$ Then, $Q_x^{\lambda}[i, \psi, \eta] = \sum_{j \leq i} \sum_{u \in \Gamma_{v_j}} \mathrm{cost}_{f_i}(u)$.

Proof Sketch. For "$\geq$", let $\psi' \trianglelefteq \psi\,|_{\mathrm{YW}^\Gamma_{v_i}}$ and $\eta' \trianglelefteq \eta\,|_{\mathrm{YW}^\Gamma_{v_i}}$ such that equality holds in (7). Let $f_{i-1} \in \mathcal{LF}_{N,\phi}$ minimize $\sum_{j<i}\sum_{u\in\Gamma_{v_j}} \mathrm{cost}_{f_{i-1}}(u)$ among all lineage functions satisfying (a)–(c) for $i-1$. Let $f^* \in \mathcal{LF}_{N,\phi}$ minimize $\sum_{u\in\Gamma_{v_i}} \mathrm{cost}_{f^*}(u)$ among all lineage functions that, for all $w \in \mathrm{YW}^\Gamma_{v_i}$, satisfy $\lambda(w) = f^*(w)$, $\psi'(w) = f^*(w) \cap \bigcup_{u\in\mathrm{Pred}^{\uparrow v_i}_N(w)} f^*(u)$ and $\eta'(w) = \sum_{u\in\Gamma_{v_i}} |f^*(u)|$. By induction hypotheses, the cost of f_{i-1} on Z_i is $Q^\lambda_x[i-1, \psi-\psi', \eta-\eta']$ and the cost of f^* on Γ_{v_i} is $T^{\mathcal{PT}}\left[v_i, \lambda\,|_{\mathrm{YW}^\Gamma_{v_i}}, \psi', \eta'\right]$. From f_{i-1} and f^*, we construct a lineage function $f' \in \mathcal{LF}_{N,\phi}$ whose cost on Z_i is $\sum_{j<i}\sum_{u\in\Gamma_{v_j}} \mathrm{cost}_{f_{i-1}}(u) + \sum_{u\in\Gamma_{v_i}} \mathrm{cost}_{f^*}(u)$. Then, "$\geq$" follows by optimality of f_i on Z_i.

For "$\leq$", let ψ' and η' be such that, for all $w \in \mathrm{YW}^\Gamma_{v_i}$, we have $\psi'(w) = f_i(w) \cap \bigcup_{u\in\mathrm{Pred}^{\uparrow v_i}_N(w)} f_i(u) \subseteq \psi(w)$ and $\eta'(w) = \sum_{u\in\mathrm{Pred}^{\uparrow v_i}_N(w)} |f_i(u)|$. By independence of sub-solutions, f_i is optimal on Z_{i-1} and on Γ_{v_i} so, by induction hypotheses, the cost of f_i on Z_{i-1} is $Q^\lambda_x[i-1, \psi-\psi', \eta-\eta']$ and the cost of f_i on Γ_{v_i} is $T^{\mathcal{PT}}\left[v_i, \lambda\,|_{\mathrm{YW}^\Gamma_{v_i}}, \phi', \eta'\right]$. Since ψ' and η' are only one of the possible choices for the minimum in (7), "$\leq$" follows. ■

For "$\geq$", let $D \subseteq U \subseteq \phi(x)$ such that equality holds in (6). We construct a lineage function f' that assigns $f'(x) = U$ and such that the lineages of D are inherited from parents of x (in N) that are below x in Γ. To this end, we ask the dynamic programming table for the cost of a lineage function that is optimal on Z_t and such that 1. $\psi'(x) = D$ *(lineages in D are inherited from parents of x in Γ_x)* 2. $\psi'(w) = \psi'(w) \setminus U$ for all $w \in \mathrm{Succ}^\uparrow_N(x)$ *(children of x in YW^Γ_x no longer need to inherit the lineages in U from Γ_x)* 3. $\eta'(x) = |U| \dot{-} \sum_{u\in\mathrm{Pred}^\downarrow_N(x)} |\lambda_x(u)|$ *(x needs to inherit $|U|$ lineages in total: $|\lambda_x(u)|$ come from every parent u of x in YW^Γ_x while the rest has to be inherited from Γ_x)* and 4. $\eta'(w) = \eta_x(w) \dot{-} |U|$ for all $w \in \mathrm{Succ}^\uparrow_N(x)$ *(children of x in YW^Γ_x can inherit a maximum of $|U|$ lineages from x)*. Since the functions $\lambda' := \lambda_x\,[x \to U]$, $\psi' := \psi_x\left[x \to D, \forall_{u\in\mathrm{Succ}^\uparrow_N(x)} w \to \psi_x(w) \setminus U\right]$ and $\eta' := \eta_x\left[x \to |U| \dot{-} \sum_{u\in\mathrm{Pred}^\downarrow_N(x)} |\lambda_x(u)|, \forall_{u\in\mathrm{Succ}^\uparrow_N(x)} w \to \eta_x(w) \dot{-} |U|\right]$ satisfy the conditions of Claim 27, the optimal cost of such a lineage function f' on Z_t is $Q^\lambda_x[t, \psi', \eta']$. Further, the cost of f' on x is the number of lineages in U that is not inherited "for free" from parents of x, that is, $|U \setminus (D \cup \bigcup_{u\in\mathrm{Pred}^\downarrow_N(x)} \lambda_x(u))|$. Then, "$\geq$" follows by optimality of f on Γ_x.

For "$\leq$", let $U := f(x)$ and let $D := U \cap \bigcup_{u\in\mathrm{Pred}^\uparrow_N(x)} f(x)$ be the set of lineages of U that are inherited from parents of x in N that are below x in Γ. By independence of sub-solutions, f is optimal on Z_t so, by Claim 27, its cost on Z_t is $Q^\lambda_x[t, \psi', \eta']$ where $\psi' := \psi_x[\ldots]$ and $\eta' := \eta_x[\ldots]$ are defined as in (6) and its cost on x is $|f(x) \setminus (\bigcup_{u\in\mathrm{Pred}^\uparrow_N(x)} f(x) \cup \bigcup_{u\in\mathrm{Pred}^\downarrow_N(x)} f(x))| = |U \setminus (D \cup \bigcup_{\mathrm{Pred}^\downarrow_N(x)} f(x))|$. Then, "$\leq$" follows from the fact that U and D are only one of the possible choices for the minimum in (6). ◀

To solve the parental parsimony problem given N, ϕ and Γ, we compute $T^{\mathcal{PT}}[x, \lambda_x, \psi_x, \eta_x]$ for each x bottom-up in Γ, each $\psi_x, \lambda_x : \mathrm{YW}^\Gamma_x \to 2^C$ with $\psi_x \trianglelefteq \lambda_x \trianglelefteq \phi$ and each $\eta_x : \mathrm{YW}^\Gamma_x \to \{0, \ldots, |C|\}$ (by Definition 24, no value larger than $|C|$ ever enters η_x and all modifications to η_x decrease the mapped-to values). To this end, $Q^\lambda_x[i, \psi, \eta]$ is computed for each x, i, λ, ψ, and η by making at most $2^{|C|\cdot|\,\mathrm{YW}^\Gamma_x\,|} \cdot |C|^{|\,\mathrm{YW}^\Gamma_x\,|}$ queries to $Q^{\psi_x}_{x,c_x}$ and $T^{\mathcal{PT}}$. As there are $O(|A(N)|)$ valid combinations of x and i, the table Q can be computed in $O(|A(N)| \cdot 3^{|C|\cdot\mathrm{yw}(N)} \cdot |C|^{\mathrm{yw}(N)} \cdot 2^{|C|\cdot\mathrm{yw}(N)} \cdot |C|^{\mathrm{yw}(N)}) = O(|A(N)| \cdot 6^{|C|\cdot\mathrm{yw}(N)|} \cdot 4^{\mathrm{yw}(N)\cdot\log|C|})$ time. Further, computing each $T^{\mathcal{PT}}[x, \lambda_x, \psi_x, \eta_x]$ requires testing $3^{|\phi(x)|} \leq 3^{|C|}$ choices for $D \subseteq U \subseteq \phi(x)$ and computing $|U \setminus (D \cup \bigcup_{u\in\mathrm{Pred}^\downarrow_N(x)} \lambda_x(u))|$ in $O(|C|)$ time (we precompute $\bigcup_{u\in\mathrm{Pred}^\downarrow_N(x)} \lambda_x(u)$ for each fix x and λ_x). Thus, the table $T^{\mathcal{PT}}$ can be computed in $O(3^{|C|\cdot\mathrm{yw}(N)} \cdot (|C|^{\mathrm{yw}(N)+1} \cdot 3^{|C|} + |A(N)|))$ time, which is dominated by the construction of Q.

▶ **Theorem 28.** *Given a network N, $\phi : V(N) \to 2^C$ and a tree Γ agreeing with N, the parental parsimony score of (N, ϕ) can be computed in $O(6^{\mathrm{yw}(\Gamma)\cdot|C|} \cdot 4^{\mathrm{yw}(\Gamma)\cdot\log|C|} \cdot |A(N)|)$ time.*

Again, we can replace $\mathrm{yw}(\Gamma)$ by $\mathrm{tw}(N)$ using Proposition 12.

▶ **Corollary 29.** *Let (N, ϕ) be an instance of* Parental Parsimony. *Let $t \geq \mathrm{tw}(N)$ and let T be the time in which a width-t tree decomposition of N can be computed. Then, the parental parsimony score of (N, ϕ) can be computed in $O(T + 6^{t\cdot|C|} \cdot 4^{t\cdot\log|C|} \cdot |A(N)|)$ time.*

Note that the parental parsimony setting supports assigning multiple states of a character to a single species, thereby modeling species carrying multiple alleles of a single gene. By forcing $D \subseteq U = \phi(x)$ instead of $D \subseteq U \subseteq \phi(x)$ if x is a leaf, we can trivially modify our dynamic programming to explain multiple character states in extant species.

Corollaries 18, 23 and 29 give the running times of our algorithms as depending on the treewidth of N. The state-of-the-art solutions for Hardwired Parsimony, Softwired Parsimony and Parental Parsimony have the following respective running times: $O(|C|^{r+2}|V(N)|)$ [21], $O(2^\ell|C|^2|V(N)||A(N)|)$ [13] and $O(|2^C|^{\ell+3}|V(N)|)$ [29]. Since the scanwidth of N is potentially much smaller than its level ℓ [27], and the treewidth of N is smaller than its scanwidth [3], we have $\mathrm{tw}(N) - 1 \leq \ell \leq r$. Thus, we expect that there will be several cases where our algorithms will be faster than the current best-known ones.

5 Discussion

In this paper, we focused on the small version of the parsimony problem for networks given a specific position in the genome. When markers can be assumed to be independent, as it is the case when a certain distance is preserved between genomic locations included in the matrix, each position can be analyzed separately, and the parsimony score of a network w.r.t. the matrix is simply the sum of the parsimony scores of the network for each genomic location. Thus, the algorithms presented here can be easily expanded to several independent genomic locations. Moreover, our formulations are defined for networks that are not necessarily binary, can account for polymorphism and can impose restrictions on ancestral states. As discussed above, our algorithms can be orders of magnitude faster than the state-of-the-art solutions. A comparison of the reticulation number, the level, the scanwidth and the treewidth for practically relevant classes of networks would thus be an interesting project for future work.

Our results are slightly overshadowed by the fact that optimal tree decompositions are very hard to compute, with even the best-known parameterized algorithm being considered impractical (see survey [5]). However, the treewidth can be 2-approximated in single-exponential time [24] and, with development driven by recent issues of the PACE challenge [10], more practical exact algorithms are now available as well [28]. We would welcome similar efforts also for the scanwidth, which is also hard to compute [3].

The ability to fast-score phylogenetic networks under the parsimony framework could be a big help in designing likelihood-based heuristics or bayesian methods to infer networks from independent markers [31, 27] by providing fast heuristics to compute the initial networks with which to start the likelihood or bayesian search, or to design fast local-search techniques.

In the future, we would like to tackle the Small Parsimony problem for several *dependent* genomic locations (e.g. a gene). Little is known for this problem, except that it stays NP-hard even for binary characters even on level-1 networks [22] and that it is fixed-parameter tractable in the number of reticulations of the network [26]. Another important direction would be to study the Big Parsimony problem, which is currently wide open, even lacking a consensus of the definition of optimality [26, 17, 30, 6].

References

1 Stefan Arnborg. Efficient algorithms for combinatorial problems on graphs with bounded decomposability—a survey. *BIT Numerical Mathematics*, 25(1):1–23, 1985.

2 Various Authors. The graph parameter hierarchy. Available at https://gitlab.com/gruenwald/parameter-hierarchy, April 2021.

3 Vincent Berry, Celine Scornavacca, and Mathias Weller. Scanning phylogenetic networks is NP-hard. In *Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM'20)*, pages 519–530. Springer, 2020.

4 Hans L Bodlaender. A linear-time algorithm for finding tree-decompositions of small treewidth. *SIAM Journal on Computing*, 25(6):1305–1317, 1996.

5 Hans L. Bodlaender. Discovering treewidth. In *Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM'05)*, pages 1–16, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

6 Christopher Bryant, Mareike Fischer, Simone Linz, and Charles Semple. On the quirks of maximum parsimony and likelihood on phylogenetic networks. *Journal of Theoretical Biology*, 417:100–108, 2017.

7 David Bryant and Jens Lagergren. Compatibility of unrooted phylogenetic trees is FPT. *Theoretical Computer Science*, 351(3):296–302, 2006.

8 Laurent Bulteau and Mathias Weller. Parameterized algorithms in bioinformatics: an overview. *Algorithms*, 12(12):256, 2019.

9 Bruno Courcelle. The monadic second-order logic of graphs. i. recognizable sets of finite graphs. *Information and computation*, 85(1):12–75, 1990.

10 Holger Dell, Christian Komusiewicz, Nimrod Talmon, and Mathias Weller. The PACE 2017 Parameterized Algorithms and Computational Experiments Challenge: The Second Iteration. In *12th International Symposium on Parameterized and Exact Computation (IPEC 2017)*, volume 89 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 30:1–30:12, Dagstuhl, Germany, 2018. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

11 Nick D. Dendris, Lefteris M. Kirousis, and Dimitrios M. Thilikos. Fugitive-search games on graphs and related parameters. *Theoretical Computer Science*, 172(1):233–254, 1997.

12 Joseph Felsenstein. *Inferring phylogenies*, volume 2. Sinauer associates Sunderland, MA, 2004.

13 Mareike Fischer, Leo Van Iersel, Steven Kelk, and Celine Scornavacca. On computing the maximum parsimony score of a phylogenetic network. *SIAM Journal on Discrete Mathematics*, 29(1):559–585, 2015.

14 Walter M Fitch. Toward defining the course of evolution: minimum change for a specific tree topology. *Systematic Biology*, 20(4):406–416, 1971.

15 Jotun Hein. Reconstructing evolution of sequences subject to recombination using parsimony. *Mathematical Biosciences*, 98(2):185–200, 1990.

16 Daniel H Huson, Regula Rupp, and Celine Scornavacca. *Phylogenetic networks: concepts, algorithms and applications*. Cambridge University Press, 2010.

17 G. Jin, L. Nakhleh, S. Snir, and T. Tuller. Inferring phylogenetic networks by the maximum parsimony criterion: A case study. *Molecular Biology and Evolution*, 24(1):324–337, 2006.

18 G. Jin, L. Nakhleh, S. Snir, and T. Tuller. Maximum likelihood of phylogenetic networks. *Bioinformatics*, 22(21):2604–2611, 2006.

19 Guohua Jin, L. Nakhleh, S. Snir, and T. Tuller. Parsimony score of phylogenetic networks: Hardness results and a linear-time heuristic. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 6(3):495–505, 2009.

20 Lavanya Kannan and Ward C. Wheeler. Maximum Parsimony on Phylogenetic networks. *Algorithms for Molecular Biology*, 7(1):9, 2012.

21 Lavanya Kannan and Ward C. Wheeler. Exactly computing the parsimony scores on phylogenetic networks using dynamic programming. *Journal of Computational Biology*, 21(4):303–319, 2014.

22 Steven Kelk, Fabio Pardi, Celine Scornavacca, and Leo van Iersel. Finding a most parsimonious or likely tree in a network with respect to an alignment. *Journal of Mathematical Biology*, 78(1-2):527–547, 2019.

23 Ton Kloks. *Treewidth: computations and approximations*, volume 842. Springer Science & Business Media, 1994.

24 Tuukka Korhonen. Single-exponential time 2-approximation algorithm for treewidth. *CoRR*, abs/2104.07463, 2021. arXiv:2104.07463.

25 Guillaume Mescoff, Christophe Paul, and Dimitrios Thilikos. A polynomial time algorithm to compute the connected tree-width of a series-parallel graph, 2021. arXiv:2004.00547v5.

26 Luay Nakhleh, Guohua Jin, Fengmei Zhao, and John Mellor-Crummey. Reconstructing phylogenetic networks using maximum parsimony. In *2005 IEEE Computational Systems Bioinformatics Conference (CSB'05)*, pages 93–102. IEEE, 2005.

27 Charles-Elie Rabier, Vincent Berry, Marnus Stoltz, João D. Santos, Wensheng Wang, Glaszmann Jean-Christophe, Fabio Pardi, and Celine Scornavacca. On the inference of complicated phylogenetic networks by Markov Chain Monte-Carlo. Submitted.

28 Hisao Tamaki. Positive-instance driven dynamic programming for treewidth. *Journal of Combinatorial Optimization*, 37(4):1283–1311, 2019.

29 Leo Van Iersel, Mark Jones, and Celine Scornavacca. Improved maximum parsimony models for phylogenetic networks. *Systematic Biology*, 67(3):518–542, 2018.

30 Ward C Wheeler. Phylogenetic network analysis as a parsimony optimization problem. *BMC Bioinformatics*, 16(1):1–9, 2015.

31 Jiafan Zhu, Dingqiao Wen, Yun Yu, Heidi M Meudt, and Luay Nakhleh. Bayesian inference of phylogenetic networks from bi-allelic genetic markers. *PLoS Computational Biology*, 14(1):e1005932, 2018.

32 Jiafan Zhu, Yun Yu, and Luay Nakhleh. In the light of deep coalescence: revisiting trees within networks. *BMC Bioinformatics*, 17(14):271–282, 2016.

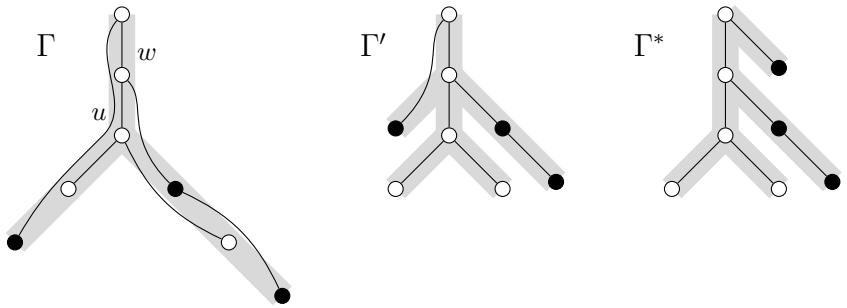


Figure 5 Example for the construction of Γ' (middle) from Γ (left) in Lemma 7. Repeated application yields Γ^* (right), for which $v \leq_{\Gamma^*} u \Rightarrow u \rightsquigarrow_{G,\Gamma^*} v$. The rooted trees Γ, Γ', and Γ^* are drawn with thick, gray lines. Thin, black lines are edges of G. For the indicated node u, the black nodes are in X, that is, they are below u in Γ but not connected to u in $G[\Gamma_u]$.

A Proofs of results in the main text

A.1 Proof of Lemma 4

Proof. (a), (b): We show for all vertices w on a u-v-path p in Γ^σ that $w \leq_\sigma u$ and $u \rightsquigarrow_{G,\sigma} w$. The base case $w = u$ holds trivially. For the induction step, let q preceed w in p. Since Γ^σ contains the arc qw, Definition 1 implies $q \rightsquigarrow_{G,\sigma} w$ and, since $q \leq_\sigma u$ by induction hypothesis, $w \leq_\sigma q \leq_\sigma u$ and $u \rightsquigarrow_{G,\sigma} w$. For the reverse direction of (b), note that, by Definition 1, uv is an arc of the DAG whose transitive reduction Γ^σ is.

(c),(d): Since $G[\sigma[1..r]] = G$ and G is connected, there is an r-x-path in $G[\sigma[1..r]]$ for all $x \in V(G)$ and, thus, Γ^σ is connected and rooted at r.

(e): To prove that Γ^σ is a tree, assume there is a vertex $x \in V(G)$ with two distinct parents y and z in Γ^σ. Without loss of generality, let $y <_\sigma z$. By (b), $y \rightsquigarrow_{G,\sigma} x$ and $z \rightsquigarrow_{G,\sigma} x$. Since $\sigma[1..y] \subsetneq \sigma[1..z]$, we conclude $z \rightsquigarrow_{G,\sigma} y$, implying $zy \in A(\Gamma^\sigma)$ and contradicting Γ^σ being a transitive reduction.

(f): Note that $u \rightsquigarrow_{G,\sigma} v$, implying $v \leq_{\Gamma^\sigma} u$ by (b).

(g): For each $uv \in E(G)$, either $u <_\sigma v$, implying $u \leq_{\Gamma^\sigma} v$, or $v <_\sigma u$, implying $v \leq_{\Gamma^\sigma} u$ (both by (f)).

(h) "$\subseteq$": Let $x \in V(G)$ and let $y \in \mathrm{YW}_x^{\Gamma^\sigma}$. By Definition 2, $y >_{\Gamma^\sigma} x$ (implying $y >_\sigma x$ by (a)) and there is some $z \leq_{\Gamma^\sigma} x$ (implying $z \leq_\sigma x$ by (a)) with $yz \in E(G)$. Then, by (b), $x \rightsquigarrow_{G,\sigma} z$. But then, $y \in \mathrm{ZW}_x^\sigma$ by Definition 1.

(h) "$\supseteq$": Let $x \in V(G)$ and let $y \in \mathrm{ZW}_x^{\Gamma^\sigma}$, that is, $x <_\sigma y$ and there is some $z \in \sigma[1..x]$ with $x \rightsquigarrow_{G,\sigma} z$ and $yz \in E(G)$. Then, $z \leq_\sigma x <_\sigma y$. By (b), $z \leq_{\Gamma^\sigma} x$ and, by (f), $z \leq_{\Gamma^\sigma} y$. Thus, as Γ^σ is a tree (by (e)), x and y are not unrelated in Γ^σ. Moreover, $y \not\leq_\sigma x$ implies $y \not\leq_{\Gamma^\sigma} x$ by (b) and, thus, $x <_{\Gamma^\sigma} y$. Together with $z \leq_{\Gamma^\sigma} x$ and $yz \in E(G)$, this implies $y \in \mathrm{YW}_x^{\Gamma^\sigma}$. ◀

A.2 Proof of Lemma 7

(See Figure 5).

Proof. Let $u \in V(G)$ such that $X := \{v <_\Gamma u \mid u \not\rightsquigarrow_{G,\Gamma} v\} \neq \varnothing$. We will modify Γ into Γ' with $\mathrm{yw}(\Gamma') \leq \mathrm{yw}(\Gamma)$ such that Γ' agrees with G and the relation $\leq_{\Gamma'}$ is a strict subset of $\leq_\Gamma$. To this end, note that u has a parent w in Γ as, otherwise, $G[\Gamma_u] = G$, implying $X = \varnothing$. Then, Γ' results from Γ by

1. replacing Γ by $\Gamma \uparrow (\Gamma_u \setminus X)$ and
2. dangling $\Gamma_u \uparrow X$ from w.

First, we show that Γ' agrees with G. To this end, let $xy \in E(G)$ and let x and y be unrelated in Γ'. If neither x nor y are in Γ_u then, by construction of Γ', they are also unrelated in Γ, contradicting that Γ agrees with G. So, without loss of generality, suppose $x \leq_\Gamma u$. Since $xy \in E(G)$ and Γ is a tree agreeing with G, we thus know that u and y are not unrelated in Γ. If $u <_\Gamma y$, then $w \leq_\Gamma y$ and, thus, $x \leq_{\Gamma'} y$. Thus, suppose $y \leq_\Gamma u$. Clearly, if $x, y \in X$ or $x, y \notin X$, then x and y are also unrelated in Γ, contradicting its agreement with G. Thus, without loss of generality, suppose $x \in X$ and $y \notin X$, that is, $u \not\rightsquigarrow_{G,\Gamma} x$ and $u \rightsquigarrow_{G,\Gamma} y$, contradicting $xy \in E(G)$.

Second, we show that $\leq_{\Gamma'}$ is a strict subset of $\leq_\Gamma$. To this end, let $xy \in A(\Gamma')$ and assume towards a contradiction that $y \not<_\Gamma x$. Clearly, if $x \not\leq_{\Gamma'} w$, then $xy \in A(\Gamma)$ contradicting $y \not<_\Gamma x$. Further, if $x = w$, then either $y \in X$ or y is a child of w in Γ, all of which imply $y <_\Gamma x$. Thus, $x <_{\Gamma'} w$. Since $xy \cap X = \{x\}$ or $xy \cap X = \{y\}$ contradicts $xy \in A(\Gamma')$, we have $x, y \in X$ or $x, y \notin X$. But then, $y <_\Gamma x$ by Observation 6. Thus, $\leq_{\Gamma'}$ is a subset of $\leq_\Gamma$ and it is strict since we have $v \leq_\Gamma u$ and $v \not\leq_{\Gamma'} u$ for all $v \in X \neq \varnothing$.

Third, $\operatorname{yw}(\Gamma') \leq \operatorname{yw}(\Gamma)$ follows by Lemma 3. ◀

A.3 Proof of Lemma 9

Proof. "$\geq$": Let σ be an ordering of $V(G)$ such that $\operatorname{zw}(\sigma) = \operatorname{zw}(G)$. By Lemma 4(h), we have $\operatorname{zw}(\sigma) = \operatorname{yw}(\Gamma^\sigma)$ for the canonical extension tree Γ^σ of σ. Thus, $\operatorname{zw}(G) = \operatorname{zw}(\sigma) = \operatorname{yw}(\Gamma^\sigma) \geq \operatorname{yw}(G)$.

"$\leq$": Let Γ be some rooted tree agreeing with G such that $\operatorname{yw}(\Gamma) = \operatorname{yw}(G)$ and, by Lemma 7, suppose

$$u \leq_\Gamma v \Rightarrow v \rightsquigarrow_{G,\Gamma} u. \tag{8}$$

Let σ be any ordering of $V(G)$ obtained by repeatedly picking and removing any leaf of Γ.

▷ Claim 30. For each $u, v \in V(G)$, we have $u \leq_\Gamma v$ if and only if $v \rightsquigarrow_{G,\sigma} u$.

Proof. First, note that all nodes below v in Γ are chosen before v, so $\Gamma_v \subseteq \sigma[1..v]$.

"$\Rightarrow$": Let $u \leq_\Gamma v$, that is, $u \in \Gamma_v$, implying $u \leq_\sigma v$. By (8), v is connected to u in $G[\Gamma_v]$ and, as $\Gamma_v \subseteq \sigma[1..v]$, also in $G[\sigma[1..v]]$.

"$\Leftarrow$": Let p be a v-u-path in $G[\sigma[1..v]]$. By Lemma 8, p has a unique maximum w in Γ. Hence, $v \leq_\Gamma w$ and, by "$\Rightarrow$", we have $v \leq_\sigma w$. Since p lives entirely in $G[\sigma[1..v]]$, that is, $V(p) \subseteq \sigma[1..v]$, we also have $w \leq_\sigma v$. Thus, $v = w$ and, since $u \in V(p)$, we have $u \leq_\Gamma w = v$ by maximality of w. ■

To prove the lemma, we show $\operatorname{YW}_x^\Gamma \supseteq \operatorname{ZW}_x^\sigma$ for each $x \in V(G)$. Let $y \in \operatorname{ZW}_x^\sigma$, that is $y >_\sigma x$ and there is some $z \in \sigma[1..x]$ with $yz \in E(G)$ and $x \rightsquigarrow_{G,\sigma} z$. By Claim 30, $z \leq_\Gamma x$. Further, as $yz \in E(G)$ and Γ agrees with G, y and z are not unrelated in Γ and, since $z \leq_\Gamma x$, neither are x and y. Since $y <_\Gamma x$ implies $y <_\sigma x$ by Claim 30, contradicting $y >_\sigma x$, we conclude $x <_\Gamma y$. Together with $z \leq_\Gamma x$ and $yz \in E(G)$, this implies $y \in \operatorname{YW}_x^\Gamma$. ◀

A.4 Proof of Proposition 12

Proof. "$\leq$": Let (T, B) be a nice tree decomposition for G of width $\operatorname{tw}(G)$ and let $F \subset V(T)$ denote the set of all "forget"-nodes in T (noting that the root of T is in F). We construct Γ

from T by contracting all nodes in $V(T) \setminus F$ onto their respective parents[1] and identifying all nodes $x \in F$ with the vertex $v \in V(G) \setminus B(x)$ of G that is forgotten in x. By Observation 11, $V(\Gamma) = V(G)$.

First, we show that Γ agrees with G. To this end, let $uv \in E(G)$ and let $f_u, f_v \in V(T)$ denote the unique "forget u" and "forget v"-nodes in T, which are distinct since T is nice. By Definition 10(a), there is a node $q \in V(T)$ with $uv \subseteq B(q)$ and, by Observation 11, $q <_T f_u, f_v$. Thus, f_u and f_v are not unrelated in T and, by Observation 5, neither in Γ.

Second, we show for all $v \in \Gamma$ and the unique "forget v"-node f_v in T that $\mathrm{YW}_v^\Gamma \subseteq B(f_v)$. Let $u \in \mathrm{YW}_v^\Gamma$, that is, $u >_\Gamma v$ and there is some $w \leq_\Gamma v$ with $uw \in E(G)$ (note that $w \neq u$ but $w = v$ is possible). Let f_u and f_w be the unique "forget u" and "forget w"-nodes in T, which are distinct since T is nice. Then, $w \leq_\Gamma v <_\Gamma u$ and, by Observation 5, $f_w \leq_T f_v <_T f_u$. Since $uw \in E(G)$, Definition 10(a) implies that there is a node q of T with $uw \subseteq B(q)$, implying $q <_T f_u, f_w$. Then, by Definition 10(b), $u \in B(x)$ for all x with $q \leq_T x <_T f_u$ and, since $q <_T f_w <_T f_v <_T f_u$, we have $u \in B(f_v)$. Thus, $\mathrm{YW}_v^\Gamma \subseteq B(f_v)$, implying $\mathrm{yw}(G) \leq \mathrm{YW}_v^\Gamma \leq |B(f_v)|$ and, since f_v has a child x with $B(x) = B(f_v) \cup \{v\}$, we know $|B(f_v)| = |B(x)| - 1 \leq \mathrm{tw}(T, B) = \mathrm{tw}(G)$.

"$\geq$": Let Γ be a tree with $\mathrm{yw}(\Gamma) = \mathrm{yw}(G)$ that agrees with G. For all $u \in V(G)$, we define $B(u) := \mathrm{YW}_u^\Gamma \cup \{u\}$ and show that (Γ, B) is a tree-decomposition for G noting that its width is $\mathrm{yw}(\Gamma) = \mathrm{yw}(G)$.

First, to prove Definition 10(a), let $uv \in E(G)$. Since Γ agrees with G, either $u <_\Gamma v$ or $v <_\Gamma u$. Without loss of generality, suppose the latter. Then, $u \in \mathrm{YW}_v^\Gamma$ by Definition 2 (using $w = v$), implying that $uv \in B(v)$.

Second, let $u, v \in V(G)$ be distinct such that $u \in B(v) = \mathrm{YW}_v^\Gamma \cup \{v\}$, implying $u \in \mathrm{YW}_v^\Gamma$ since $u \neq v$. By Definition 2, there is some $w \leq_\Gamma v$ with $uw \in E(G)$ and $v <_\Gamma u$, implying that Γ contains a unique u-v-path p. To show Definition 10(b), it suffices to prove $u \in B(x)$ for all $x \in V(p)$ (since v has been chosen arbitrarily, a path with these properties exists for all v' with $u \in B(v')$, so they all contain the node u and are, thus, connected). For $x = u$ this follows by definition of $B(u)$. Otherwise, $x <_\Gamma u$ since $x \in V(p)$. But then, $w \leq_\Gamma v \leq_\Gamma x <_\Gamma u$ and $uw \in E(G)$, implying $u \in \mathrm{YW}_x^\Gamma \subseteq B(x)$. ◀

A.5 Proof of Lemma 14

Proof. Towards a contradiction, assume that the lemma is false. We construct $\psi^* : V(N) \to C$ with

$$\psi^*(u) = \begin{cases} \psi_y(u) & \text{if } u \in \Gamma_y \text{ for any } y \in Y \\ \psi(u) & \text{otherwise} \end{cases}$$

Note that ψ^* and ψ coincide with ψ_y on YW_y^Γ for all $y \in Y$. Thus, $\delta_{\psi^*}(u, w) = \delta_{\psi_y}(u, w)$ if $uw \in A_y(S^*)$ for any $y \in Y$ and $\delta_{\psi^*}(u, w) = \delta_\psi(u, w)$, otherwise. Further, we construct a digraph $S^* := (V(N), (A(S) \setminus \bigcup_{y \in Y} A_y(S)) \cup \bigcup_{y \in Y} A_y(S^y))$ which is in $\mathcal{G}$ since $(S^y)_{y \in Y}$ is a Y-substitution system for $\mathcal{G}$. Since all S^y are subnetworks of N, we know that Γ agrees with S^*. Furthermore, since $Y \subseteq \bigcup_{z \in Z} \Gamma_z$, we know that each $y \in Y$ has a $z \in Z$ with $y \leq_\Gamma z$.

[1] One can also describe Γ as the transitive reduction of $(F, >_T \cap (F \times F))$.

Thus,

$$\begin{aligned}
\sum_{z\in Z}\sum_{uw\in A_z(S^*)}\delta_{\psi^*}(u,w) &= \sum_{z\in Z}\sum_{v\in\Gamma_z}\sum_{uw\in A_{\{v\}}(S^*)}\delta_{\psi^*}(u,w)\\
&= \sum_{z\in Z}\sum_{\substack{v\in\Gamma_z\\ v\notin\bigcup_{y\in Y}\Gamma_y}}\sum_{uw\in A_{\{v\}}(S^*)}\delta_{\psi^*}(u,w) + \sum_{y\in Y}\sum_{uw\in A_y(S^*)}\delta_{\psi^*}(u,w)\\
&= \sum_{z\in Z}\sum_{\substack{v\in\Gamma_z\\ v\notin\bigcup_{y\in Y}\Gamma_y}}\sum_{uw\in A_{\{v\}}(S)}\delta_{\psi}(u,w) + \sum_{y\in Y}\sum_{uw\in A_y(S^y)}\delta_{\psi_y}(u,w)\\
&\overset{\text{assumption}}{<} \sum_{z\in Z}\sum_{\substack{v\in\Gamma_z\\ v\notin\bigcup_{y\in Y}\Gamma_y}}\sum_{uw\in A_{\{v\}}(S)}\delta_{\psi}(u,w) + \sum_{y\in Y}\sum_{uw\in A_y(S)}\delta_{\psi}(u,w)\\
&= \sum_{z\in Z}\sum_{uw\in A_z(S)}\delta_{\psi}(u,w)
\end{aligned}$$

contradicting optimality of S and ψ (that is, Lemma 14(a)) since $S^* \in \mathcal{G}$. ◀