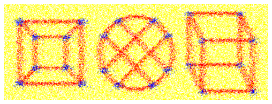
Journal of Graph Algorithms and Applications
http://jgaa.info/ vol. 21, no. 6, pp. 1003–1025 (2017)
DOI: 10.7155/jgaa.00447

Computing phylogenetic trees using topologically related minimum spanning trees

Prabhav Kalaghatgi Thomas Lengauer

Department of Computational Biology and Applied Algorithmics, Max-Planck Institut für Informatik, Saarbrücken, Germany

Abstract

Choi et al. [2] introduced a minimum spanning tree (MST)-based method called CLGrouping, for constructing tree-structured probabilistic graphical models, a statistical framework that is commonly used for inferring phylogenetic trees. While CLGrouping works correctly if there is a unique MST, we observe an indeterminacy in the method in the case that there are multiple MSTs. We demonstrate the indeterminacy of CLGrouping using a synthetic quartet tree and a tree over primate genera. The indeterminacy of CLGrouping can be removed if the input MST shares a topological relationship with the corresponding phylogenetic tree. We introduce so-called vertex order based MSTs (VMSTs) that are guaranteed to have the desired topological relationship. We relate the number of leaves in the VMST to the degree of parallelism that is offered by CLGrouping. We provide polynomial-time algorithms for constructing VMSTs and for selecting a VMST with the optimal number of leaves.

Submitted:	Reviewed:	Revised:	Accepted:	Final:
November 2016	April 2017	May 2017	August 2017	September 2017
		Published: October 2017		
	Article type: Regular paper		Communicated by: F. Vandin	

PK's work has been funded in part by the German Center for Infection Research (DZIF, German Ministry of Education and Research Grants No. TTU 05.805, TTU 05.809)

E-mail addresses: prabhavk@mpi-inf.mpg.de (Prabhav Kalaghatgi) lengauer@mpi-inf.mpg.de (Thomas Lengauer)

1 Introduction

Phylogenetic trees are tree-structured edge-weighted graphical models of evolutionary relationships containing two types of vertices: labeled vertices that represent observed organisms, and hidden vertices that represent ancestral, unobserved organisms. Phylogenies are usually constructed from homologous genomic sequences, protein sequences, and/or encoded morphological characteristics. Edges of phylogenetic trees are weighted with the average number of sequence changes (substitutions) per site.

Phylogenetic tree inference can be viewed as a combinatorial optimization problem. The most commonly used optimization criteria are maximum likelihood (ML), maximum parsimony (MP), minimum least-squares error (MLS), minimum evolution (ME), and balanced minimum evolution (BME). ML and MP are character based approaches, and, ME and MLS are distance based approaches. We briefly introduce each methodology below.

In the likelihood framework, phylogenies are modeled as tree-structured probabilistic graphical models, The objective in ML is to search for a combination of tree topology and edge lengths that maximizes the marginal likelihood of the sequence data. In an MP analysis the objective is to search for a tree topology that minimizes the total number of character changes over the edges of the tree. The MP problem was formalized by Foulds and Graham [9] as a Steiner minimal tree problem and was shown to be **NP-complete**. The ML problem was shown to be **NP-hard** by reduction from MP [3, 22].

The objectives MLS and ME are defined in terms of edge lengths that are fitted using least-squares regression to measures of evolutionary distance such as the Jukes-Cantor distance [14]. The MLS problem is to find a tree that minimizes the sum of squared errors, and was shown to be **NP-complete** by Day [5]. The ME problem is to find a shortest tree, that is a tree with the smallest sum of edge lengths. ME was shown to be **NP-complete** by Bastkowski et al. [1]. The objective BME is closely related to ME, and defines edge lengths using a special case of weighted least-squares [6]. BME was shown to be **NP-complete** by Fiorini et al. [8].

Due to the computational intractability of the optimization problems stated above, scalable methods in phylogenetics are designed using heuristics and perform local optimization, e.g., FastTree2 [19]. Neighbor joining (NJ) [23] is a widely used distance based method that performs a greedy search to find a BME tree [10].

Choi et al. [2] introduced a distance based method called Chow-Liu grouping (CLGrouping). Briefly, CLGrouping operates in two phases. The first phase involves constructing a minimum spanning tree using the pairwise distances. In the second phase, each non-leaf vertex v of the MST is visited and the subgraph that is induced by v and the neighbors N_v of v is replaced by a phylogenetic tree over the vertex group $\{v \cup N_v\}$.

Choi et al. [2] show that CLGrouping is more accurate than NJ at reconstructing phylogenetic trees with large diameter. Huang et al. [13] showed that CLGrouping affords a high degree of parallelism, because phylogenetic tree re-

construction can be performed independently for each vertex group.

During our attempt to implement CLGrouping we discovered that there are instances where the tree that is reconstructed using CLGrouping differs from the phylogenetic tree T, even if the input distances are the tree metric of T.

1.1 Our contributions

We show that the indeterminacy of CLGrouping is due to a lack of topological correspondence between the MST and the phylogenetic tree. We demonstrate the indeterminacy of CLGrouping using a synthetic quartet tree and a published primate phylogenetic tree. We introduce so-called vertex order based MSTs (VMSTs) that are guaranteed to remove the indeterminacy of CLGrouping. We relate the number of leaves in the VMST to the degree of parallelism that is offered by CLGrouping. We provide polynomial-time algorithms for constructing VMSTs and for selecting a VMST with the minimum number of leaves.

2 Terminology

A phylogenetic tree $T = (V_T = \{L_T, H_T\}, E_T)$ is an undirected edge-weighted acyclic graph with two types of vertices: labeled vertices L_T that represent observed organisms, and unlabeled hidden vertices H_T that represent unobserved organisms. Information, e.g., in the form of genomic sequences, is only present at labeled vertices. We refer to the edge weights of a phylogenetic tree as edge lengths. The length of an edge quantifies the estimated evolutionary distance between the sequences corresponding to the respective incident vertices. All edge lengths are strictly positive. A phylogenetic tree is a leaf-labeled tree if all the labeled vertices are leaves, otherwise, the phylogenetic tree is a generally labeled tree [15].

Each phylogenetic tree $T = (\{L_T, H_T\}, E_T)$ is equipped with a length function $w_T : E \to (0, \infty)$, and a unique tree metric $d_T : L_T \times L_T \to [0, \infty)$ over the labeled vertices that is defined as follows.

For each u and v in L_T such that $u \neq v$,

$$d_T(u,v) = \begin{cases} \sum_{\{i,j\} \in p_T(u,v)} w_T(i,j) & \text{if } u \neq v \\ 0 & \text{otherwise} \end{cases}$$

where $w(i,j)$ is the length of the edge $\{i,j\}$, and $p_T(u,v)$ is the alternating sequence of vertices and edges, that are visited when traversing the unique path in T from u to v.

A set of distances is additive in $T = (V_T, E_T)$ if the corresponding length function $w_T : E \to (0, \infty)$ gives rise to these distances. The distance graph $G = (V_G, E_G)$ of a phylogenetic tree T is the complete graph over L_T with each edge $\{u,v\}$ in E_G weighted with the additive distance $d_T(u,v)$. A minimum

spanning tree (MST) of an edge-weighted graph is a tree that spans all vertices of the graph, and has the minimum sum of edge weights.

Each phylogenetic tree $T = (\{L_T, H_T\}, E)$ is equipped with a split function $|_T$ that is defined as follows. The split $|_T : E_T \rightarrow \{2^{L_T}, 2^{L_T}\}$ of an edge $\{u, v\}$ is defined as the collection $\{A, B\}$ of the disjoint sets of labeled vertices such that $\{u, v\}$ is contained in each path from a vertex in A to a vertex in B. A split $\{A, B\}$ is said to be contained in a tree $T = (V_T, E_T)$ if there is an edge $\{u, v\}$ in E_T such that $|_T(u, v) = \{A, B\}$. A and B are referred to as the sides of the split. The most balanced edge of a tree is the edge that induces a split such that the difference in the set sizes of the sides is minimal.

A phylogenetic tree can be rooted by introducing a new hidden vertex ρ called the root, removing an edge $\{u, v\}$ and adding the edges $\{\rho, u\}$ and $\{\rho, v\}$, with new edge lengths satisfying $w(u, \rho) + w(\rho, v) = w(u, v)$. Rooting a tree constructs a directed acyclic graph in which each edge is directed away from the root.

A phylogenetic tree is an ultrametric tree if the tree can be rooted in such a way that all leaves are equidistant from the root.

3 Indeterminacy of Chow-Liu grouping

Choi et al. [2] introduced the procedure Chow-Liu grouping (CLGrouping) for constructing latent tree graphical models, a framework that is used for inferring phylogenetic trees. CLGrouping can be used for constructing phylogenetic trees from estimates of evolutionary distances. The authors show that CLGrouping is better at reconstructing phylogenetic trees with large diameter when compared to NJ. If the input distances are additive in the phylogenetic tree T then the authors claim that CLGrouping correctly reconstructs T.

CLGrouping consists of two stages. The first stage involves the construction of an MST M of G. The second stage iterates over the internal vertices of M and, for each internal vertex i that is visited, a vertex set V_i comprising i and the neighbors of i is constructed. Subsequently a phylogenetic tree T_i is constructed using distances between vertices in V_i. In the final step of the iteration, the graph in M that is induced by V_i is replaced by T_i (see Fig. 1E for an illustration). If i is not the first vertex to be visited then V_i may contain newly introduced hidden vertices. Let h_j be a hidden vertex that was introduced when processing the labeled vertex j. The distance from h_j to a labeled vertex l in V_i is computed as $d_{h_j l} = d_{jl} - d_{jh_j}$. The distance between two hidden vertices h_j and h_k is computed as $d_{h_j h_k} = d_{jk} - d_{jh_j} - d_{kh_k}$.

The order in which the internal vertices are visited is not specified by the authors and does not seem to be important. CLGrouping terminates once all the internal vertices of M have been visited once.

This procedure is called Chow-Liu grouping because the MSTs that are constructed using additive distances are topologically equivalent to Chow-Liu trees [4], for certain probability distributions. Please read Choi et al. [2] for further detail.

3.1 A quartet tree

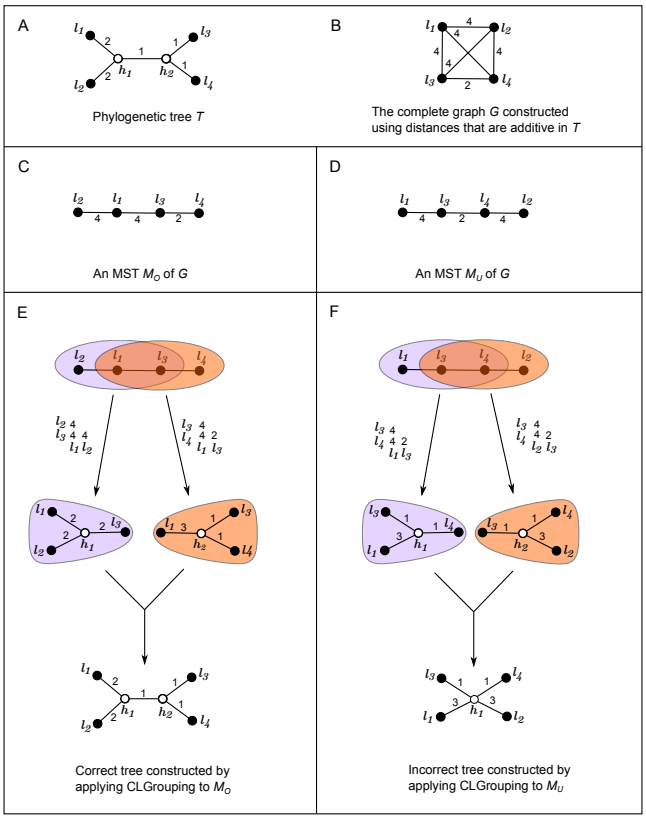

Figure 1: The example used to demonstrate that CLGrouping may not reconstruct the correct tree if there are multiple MSTs. The phylogenetic tree T that is used in this example is shown in panel A. The distance graph G of T is shown in panel B. Two MSTs of G, M_O and M_U are shown in panels C and D, respectively. M_O is a vertex-order based MST (VMST) and M_U is not a VMST. Panels E and F show the intermediate steps, and the final result of implementing CLGrouping using M_O and M_U respectively. CLGrouping reconstructs the original phylogenetic tree if it uses M_O but not if it is uses M_U.

We demonstrate the indeterminacy of CLGrouping for the quartet tree T (Fig. 1). For the corresponding distance graph G of T, two MSTs of G, M_U and M_O were constructed by hand. M_O is a vertex order based MST that was constructed using the order $l_1 < l_2 < l_3 < l_4$. M_U is not a vertex order based MST. The intermediate steps, and the final result of applying CLGrouping to M_U and M_O are shown in Fig. 1E and Fig. 1F, respectively. CLGrouping reconstructs the original phylogenetic tree if it is applied to the VMST M_O but not if it is applied to M_U.

3.2 A primate phylogenetic tree

In this subsection we demonstrate the indeterminacy of CLGrouping, using the phylogeny over the primate genera [18].

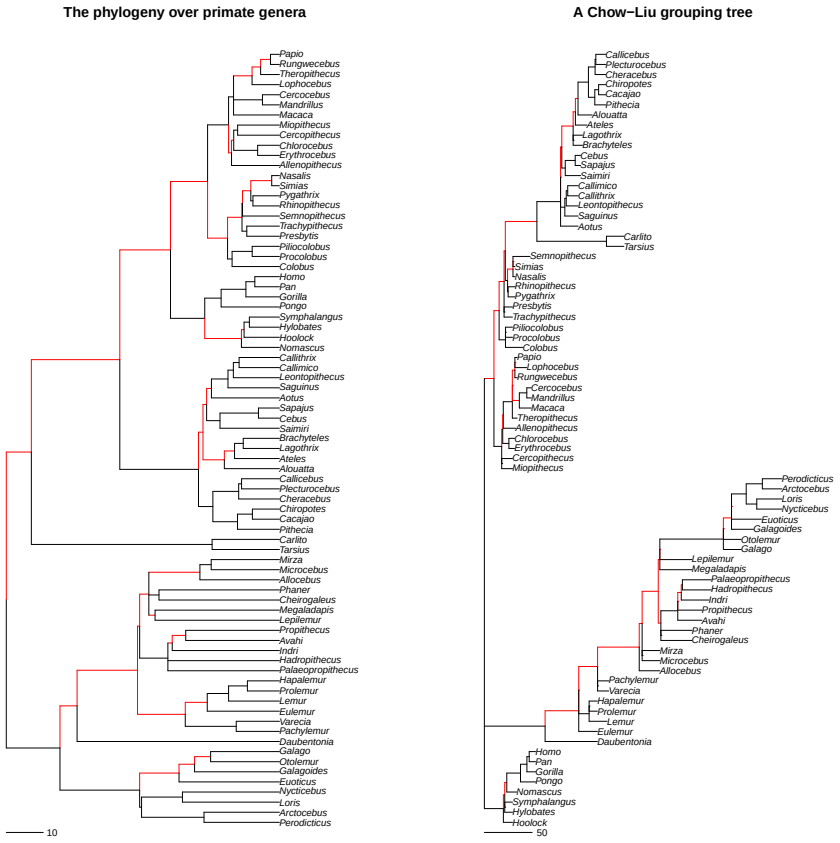

Figure 2: Left: The empirically established phylogeny T over primate genera [11, 18]. Right: A phylogeny that was constructed by applying Chow-Liu grouping to an MST of the distance graph of T. The edges that are highlighted in red correspond to splits that are contained in one tree but not the other tree. The branches in each phylogeny are scaled in units of million yrs.

3.2.1 Methodological details

The primate phylogeny was downloaded from the TimeTree database which is a comprehensive collection of published phylogenies [11, 12, 17]. The branches of the primate phylogeny represent calendar time and are scaled in units of million yrs. The primate phylogeny contains three branches of length zero that cannot

be inferred from the corresponding tree metric. A modified primate phylogenetic tree T was constructed by contracting all branches of length zero. The edges of the distance graph of T were arranged in order of increasing weight, and edges with identical weight were randomly shuffled. One hundred MSTs were constructed by applying Kruskal's algorithm [16] to the edges that were ordered as described above. We implemented CLGrouping such that the phylogeny over each vertex group was constructed using NJ. We applied CLGrouping to each MST, and computed the topological distance between each output phylogeny and the primate phylogeny using the Robinson-Foulds distance [21]. The Robinson-Foulds distance is defined as the fraction of unique splits that are present in one tree and not the other. We selected a Chow-Liu grouping tree that maximizes the Robinson-Foulds distance from the primate phylogeny. The selected Chow-Liu grouping tree is 0.4 RF distance away from the primate phylogeny and is shown in Fig. 2. In order to enable a visual comparison we rooted the Chow-Liu grouping tree at the midpoint of the most balanced edge. The primate phylogeny is an ultrametric tree and has been rooted such that the root is equidistant from the leaves. As can be seen, both trees in Fig. 2 are substantially different.

3.3 Topological relationship between MSTs and phylogenetic trees

The correctness of CLGrouping depends on a topological relationship between MSTs and phylogenetic trees that was introduced by Choi et al. [2].

In order to establish a topological relationship between minimum spanning trees and phylogenetic trees Choi et al. [2] introduced the notion of a surrogate vertex.

The surrogate vertex of a hidden vertex is the closest labeled vertex, wrt distances defined on the phylogenetic tree. Choi et al. [2] claim that minimum spanning trees can be constructed by contracting all edges along the path between each hidden vertex and the corresponding surrogate vertex. Since the procedure that constructs the MST is not aware of the true phylogenetic tree, the surrogate vertex of each hidden vertex must be selected implicitly.

In the example shown in Fig. 1, the MST M_O can be constructed by contracting the edges $\{h_1, l_1\}$, and $\{h_2, l_3\}$. Clearly there is no selection of surrogate vertices such that M_U can be constructed by contracting the path between each hidden vertex and the corresponding surrogate vertex.

Let the surrogate vertex set $S(h)$ of a vertex h be the set of all labeled vertices that are closest to h. Consider two hidden vertices h_1 and h_2, such that there are multiple labeled vertices, l_1 and l_2, that are common to the corresponding surrogate vertex sets $S(h_1)$ and $S(h_2)$. Choi et al. [2] assume that it is always possible to apply the following tie-breaking rule for implicitly selecting the corresponding surrogate vertices. A labeled vertex that is common to $S(h_1)$ and $S(h_2)$ (either l_1 or l_2) is selected as the surrogate vertex of both h_1 and h_2.

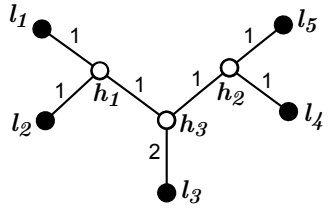

Figure 3: The phylogenetic tree that is used to demonstrate that the tie-breaking rule as defined by Choi et al. [2] cannot be applied in general.

This rule for selecting surrogate vertices cannot be applied in general. We demonstrate this with an example. For the tree shown in Fig. 3 we have $S(h_1) = \{l_1, l_2\}$, $S(h_2) = \{l_4, l_5\}$, and $S(h_3) = \{l_1, l_2, l_3, l_4, l_5\}$. There is no selection of surrogate vertices that satisfies the tie-breaking rule.

4 Vertex order based MSTs

In order to construct an MST that is guaranteed to have the desired topological correspondence with the phylogenetic tree, we propose the following definition of a surrogate vertex.

Definition 1 *Given a phylogenetic tree $T = (V_T = \{L_T, H_T\}, E_T)$ and the corresponding tree metric d_T, let there be a total order $<_V$ over the set of all labeled vertices of T. The vertex order based surrogate vertex of a vertex v in V_T is the labeled vertex in L_T that is closest wrt the tree metric d_T, and smallest wrt to the vertex order $<_V$. That is,*

$$s(v) = \underset{l \in L_T}{\operatorname{argmin}} \{d_T(l, v), l_{<_V}\}$$

*where $l_{<_V}$ is the rank of l in the order $<_V$, and the lexicographic order is applied to the ordered pair following "*argmin*" in the formula.*

The inverse surrogate set $S^{-1}(l)$ of a labeled vertex l is the set of all vertices whose surrogate vertex is l. Note that each labeled vertex is contained in its inverse surrogate set.

In order to ensure that the surrogate vertices are selected on the basis of tree metric and vertex order, it is necessary that information pertaining to vertex order is used when selecting the edges of the MST. We use Kruskal's algorithm for constructing the desired MST. Since Kruskal's algorithm takes as input a set of edges sorted wrt edge weight, we modify the input by sorting edges with respect to edge weight and vertex order as follows. It is easy to modify other algorithms for constructing MSTs in such a way that vertex order is taken into account.

Definition 2 *Given an edge-weighted graph $G = (V, E)$, and a total order $<_V$ over the vertices in V. Let $w(u, v)$ be the weight of the edge $\{u, v\}$. Edges in E are sorted wrt edge weight and vertex order using the lexicographic order that is defined below. Let the sorting be defined using the total order $<_E$. For each pair of edges $\{a, b\}$ and $\{c, d\}$ in E,*

$$\{a, b\} <_E \{c, d\} \text{ if and only if}$$

$$(w(a, b), \min(a_{<_V}, b_{<_V}), \max(a_{<_V}, b_{<_V})) < (w(c, d), \min(c_{<_V}, d_{<_V}), \max(c_{<_V}, d_{<_V}))$$

The modified algorithm for constructing a vertex order based MST (VMST) is described in Algorithm 1.

Algorithm 1 Constructing a vertex-order based MST (VMST)

Input: $(G = (V, E), <_V)$
$E_{<_V} \leftarrow$ edges in E ordered wrt edge weight and vertex order
$M_{<_V} \leftarrow$ MST constructed by applying Kruskal's algorithm to $E_{<_V}$
Output: $M_{<_V}$

Using the notion of VMSTs we will prove Lemma 1, and consequently show that the indeterminacy of CLGrouping can be removed if CLGrouping is applied to a VMST.

Lemma 1 *Adapted from parts (i) and (ii) of Lemma 8 in Choi et al. [2]. Given a phylogenetic tree $T = (V_T, E_T)$ and a total order $<$ over the labeled vertices in T, let $G = (V_G, E_G)$ be the distance graph of T. Let $M = (V_M, E_M)$ be the VMST constructed by applying Algorithm 1 to $(G, <)$. The surrogate vertex of each hidden vertex is defined with respect to the tree metric d_T and a vertex order as given in Definition 1. M is related to T as follows.*

1. If $l \in V_M$ and $h \in S^{-1}(l)$ s.t. $h \neq l$, then every vertex in the path in T that connects l and h belongs to the inverse surrogate set $S^{-1}(l)$.

2. For any two vertices that are adjacent in T, their surrogate vertices, if distinct, are adjacent in M, i.e., for all $i, j \in V_T$ with $s(i) \neq s(j)$,

$$\{i, j\} \in E_T \Rightarrow \{s(i), s(j)\} \in E_M.$$

Proof: (i). Assume that there is a vertex u on the path between h and l, such that $s(u) = k \neq l$. Since $s(u) = k$ implies that $(d_T(u, k), r_{<_V}(k)) <_V (d_T(u, l), r_{<_V}(l))$, we have $d_T(u, k) \leq d_T(u, l)$, with equality holding only if $k <_V l$.

There are seven ways to position k wrt h, u, and l (see Figure 4). We only consider the general positions.

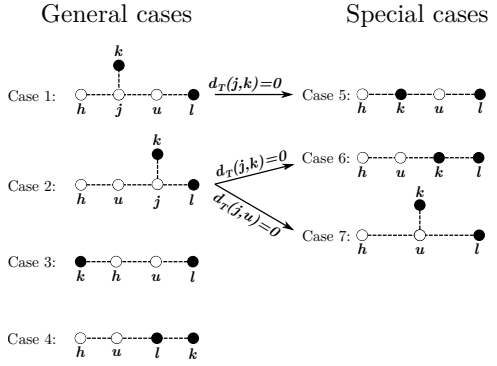

Figure 4: The cases that were considered in the proof of Lemma 1 part (i). Each case specifies one of the seven possible positions of a labeled vertex k wrt hidden vertices h and u, and a labeled vertex l. Hidden vertices are represented with white circles and labeled vertices are represented with black circles. Each dashed line represents a path between the two vertices at its end points. The condition on top of each solid arrow describes how the special cases can be constructed from the corresponding general cases.

For case 1 we have
$$\begin{aligned} & d_T(h,l) \leq d_T(h,k) \\ \Leftrightarrow & d_T(h,j) + d_T(j,u) + d_T(u,l) \leq d_T(h,j) + d_T(j,k) \\ \Leftrightarrow & d_T(j,u) + d_T(u,l) \leq d_T(j,k) \\ \Rightarrow & d_T(u,l) < d_T(u,j) + d_T(j,k) \\ \Leftrightarrow & d_T(u,l) < d_T(u,k) \text{ (contradiction since } s(u) = k) \end{aligned}$$

For case 2 we have
$$\begin{aligned} & d_T(h,l) \leq d_T(h,k) \\ \Leftrightarrow & d_T(h,u) + d_T(u,j) + d_T(j,l) \leq d_T(h,u) + d_T(u,j) + d_T(j,k) \\ \Leftrightarrow & d_T(u,j) + d_T(j,l) \leq d_T(u,j) + d_T(j,k) \\ \Leftrightarrow & d_T(u,l) \leq d_T(u,k) \text{ (contradiction since } s(u) = k) \end{aligned}$$

For case 3 we have
$$\begin{aligned} & d_T(h,l) \leq d_T(h,k) \\ \Leftrightarrow & d_T(h,u) + d_T(u,l) \leq d_T(h,k) \\ \Rightarrow & d_T(u,l) < d_T(h,k) + d_T(h,u) \\ \Leftrightarrow & d_T(u,l) < d_T(u,k) \text{ (contradiction since } s(u) = k) \end{aligned}$$

For case 4 we have
$$\begin{aligned} & d_T(u,k) = d_T(u,l) + d_T(l,k) \\ \Rightarrow & d_T(u,k) > d_T(u,l) \text{ (contradiction since } s(u) = k) \end{aligned}$$

(*ii*). Consider the edge $\{i, j\}$ in E_T such that $s(i) \neq s(j)$. Let V_i and V_j be the sides of the split that is induced by the edge $\{i, j\}$, such that V_i and V_j contain i and j, respectively. Let L_i and L_j be sets of labeled vertices that are defined as $V_i \cap V_M$ and $V_j \cap V_M$ respectively. From part (*i*) of Lemma 1 we know that $s(i) \in L_i$ and $s(j) \in L_j$. Consider the labeled vertices $l_i \in L_i \backslash \{s(i)\}$ and $l_j \in L_j \backslash \{s(j)\}$.

We have

$$\begin{aligned} d_T(l_i, l_j) &= d_T(l_i, i) + d_T(i, j) + d_T(l, j) \\ &\geq d_T(s(i), i) + d_T(i, j) + d_T(s(j), j) \\ &= d_T(s(i), s(j)) \end{aligned}$$

It follows that

$$d_T(s(i), s(j)) \leq d_T(l_i, l_j), \tag{1}$$

with equality holding only if

$$s(i) <_V l_i \text{ and } s(j) <_V l_j. \tag{2}$$

The cut property of MSTs states that, given a graph $G = (V, E)$, for each pair V_1, V_2 of disjoint sets such that $V_1 \cup V_2 = V$, each MST of G contains one of the smallest edges (wrt edge weight) which have one endpoint in V_1 and the other endpoint in V_2. Thus M contains at most one of the following edges $\{l_i, l_j\}, \{s(i), l_j\}, \{l_i, s(j)\}$ and $\{s(i), s(j)\}$. Note that the vertex order based MST M is constructed using edges that are sorted wrt edge weight and the vertex order $<_V$. Let the ordered set of edges be defined using the total order $<_E$ over E.

From equations (1) and (2) we have

$$\begin{aligned} &(d_T(s(i), s(j)), \min(s(i)_{<_V}, s(j)_{<_V}), \max(s(i)_{<_V}, s(j)_{<_V})) \\ &\qquad < (d_T(l_i, l_j), \min(l_{i<_V}, l_{j<_V}), \max(l_{i<_V}, l_{j<_V})) \end{aligned} \tag{3}$$

Thus, according to Definition 2, it follows that $\{s(i), s(i)\} <_E \{l_i, l_j\}$. Through a similar construction it can be shown that $\{s(i), s(j)\} <_E \{s(i), l_j\}$ and $\{s(i), s(j)\} <_E \{l_i, s(j)\}$. It follows that $\{s(i), s(j)\} \in E_M$. □

CLGrouping can be shown to be correct using Lemma 1 and the rest of the proof that was provided by Choi et al. [2].

The authors of CLGrouping provide a matlab implementation of their algorithm. The implementation takes as input a distance matrix which has the following property: the row index, and the column index of each labeled vertex is equal. The MST that is constructed in the authors implementation is a vertex order based MST. The vertex order is equal to the order over the column/row indices of the labeled vertices. The implementation provided by Choi et al. [2] correctly reconstructs the model tree even if there are multiple MSTs in the underlying distance graph.

5 Selecting optimal VMSTs

In the context of parallel programming, Huang et al. [13] showed that it is possible to parallelize CLGrouping by independently constructing phylogenetic trees over the vertex group associated with each non-leaf vertex, and merging them in order to construct the full phylogenetic tree. The step involving tree mergers requires a shared memory architecture.

Thus, with respect to parallelism, an optimal VMST would have the maximum number of vertex groups, and equivalently, the minimum number of leaves.

5.1 Tree shape

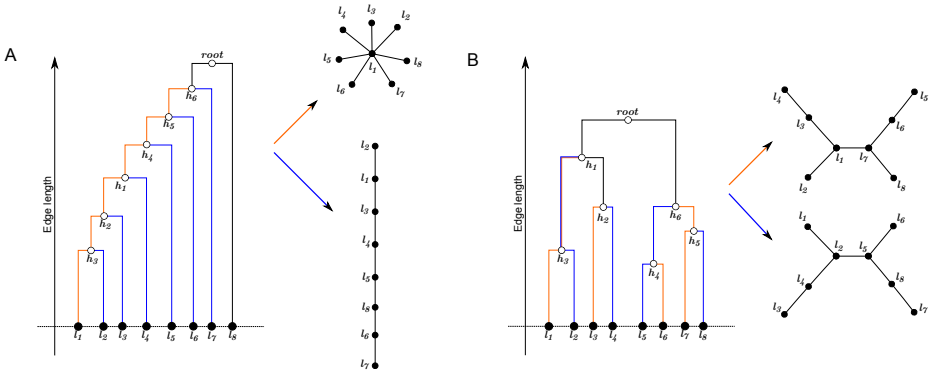

Figure 5: Both panels show ultrametric trees (left) and VMSTs with the maximum and the minimum number of leaves (right) that are constructed by contracting corresponding edges that are highlighted in orange and blue, respectively. The difference between the maximum and the minimum number of leaves in VMSTs is largest for the caterpillar tree shown in panel A, and smallest for the balanced tree shown in panel B.

In order to relate the shape of a phylogenetic tree to the number of leaves in a corresponding VMST, we consider ultrametric caterpillar trees and ultrametric balanced trees [25]. Ultrametric trees are leaf-labeled rooted phylogenetic trees satisfying the condition that the root is equidistant from all leaves. A caterpillar tree is a phylogenetic tree for which all non-leaf vertices are contained in a single path. A balanced tree is a rooted phylogenetic tree for which the path from each leaf to the root contains the same number of edges.

Consider an ultrametric caterpillar tree. There exists a corresponding VMST which has a star topology that can be constructed by contracting edges between each hidden vertex and one labeled vertex that is in the surrogate vertex set of each hidden vertex (see Fig. 5 A). A star-shaped VMST has only one vertex group, comprising all the vertices in the VMST, and does not afford any parallelism.

Instead, if the VMST was to be constructed by contracting edges between each hidden vertex h and a labeled vertex that is incident to h, then the number of the vertex groups would be $n-2$, where n is the number of vertices in the phylogenetic tree. The resulting VMST would have the minimum number of leaves (two).

Consider a phylogenetic tree $T = (\{L_T, H_T\}, E_T)$ which is an ultrametric balanced tree. For each leaf $l_1$in L_T there is another leaf l_2 in L_T such that l_1 and l_2 are incident to the same hidden vertex h in H_T. Since l_1 and l_2 are closest to h, the surrogate vertex of h is either l_1 or l_2. In each VMST of T, either l_1 or l_2 will be a leaf in the VMST. Since this is true for all leaves in L_T, each VMSTs of T will have $L_T/2$ leaves (see Fig. 5 B).

Whether or not the phylogenetic trees that are estimated from real data are ultrametric depends on the set of organisms that are being studied. Genetic sequences that are sampled from closely related organisms have been estimated to undergo substitutions at a similar rate, resulting in ultrametric phylogenetic trees [7]. With respect to the phenomenon of adaptation by natural selection, phylogenetic trees are caterpillar-like if there is strong selection; the longest path from the root represents the best-adapted lineage.

In the next section we will present an algorithm for constructing a VMST with the minimum number of leaves.

5.2 Overview of our approach

Our approach to selecting optimal VMSTs makes use of three notions, (*i*), the maximum degree $\delta_{\max}$ of each vertex across all MSTs, (*ii*), the so-called MST union graph which is a graph containing all the edges that are present in at least one MST, and, (*iii*), a common structure over the MSTs that can be defined as a laminar family.

The intuition behind our approach is described as follows. From Lemma 1 it follows that each non-leaf vertex of a VMST is a surrogate vertex. Thus we want to choose a vertex order such that we maximize the number of distinct surrogate vertices. In Section 7, we show that such a vertex order can be constructed by arranging vertices in order of non-decreasing $\delta_{\max}$. In Section 6 we show how the common laminar family and the MST union graph can be used to compute $\delta_{\max}$. The construction is exemplified graphically in Fig. 5.2.

On a related note, the general problem of selecting an MST with the minimum number of leaves (MLMST) is in **NP-complete** by reduction from the Hamiltonian path problem. MLMST specializes the problem of finding spanning trees with minimum number of leaves which is also in **NP-complete** by a similar reduction [24].

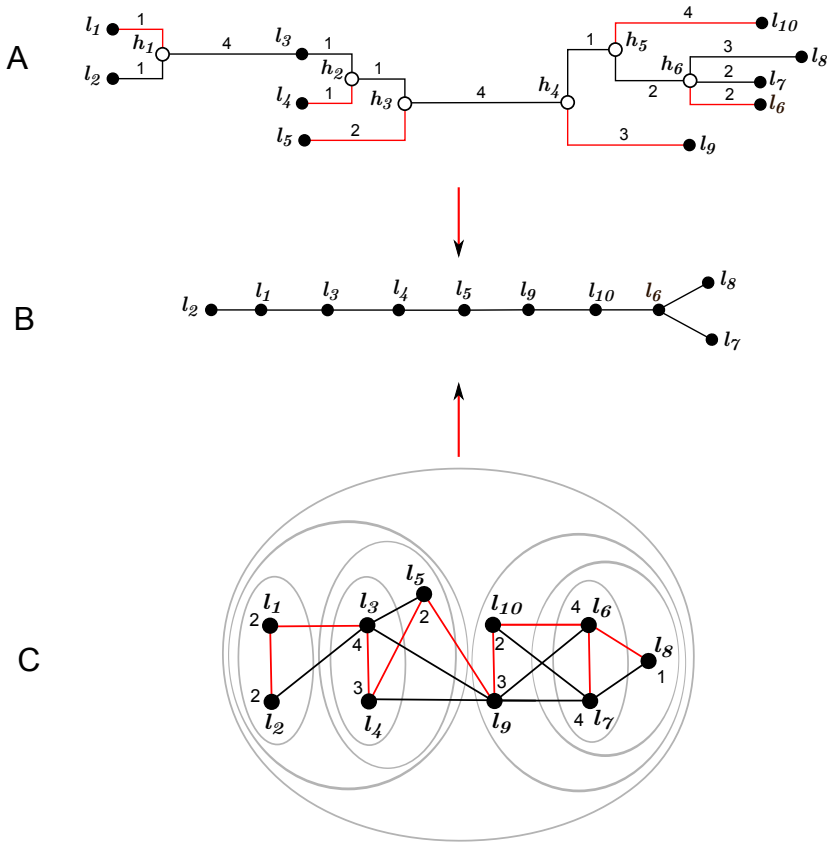

Figure 6: Panel A shows a generally labeled phylogenetic tree T with surrogate vertices selected such that the edge contraction would construct the VMST with the minimum number of leaves shown in Panel B. Panel C shows the VMST (in red) superimposed with the common laminar family and the MST union graph. Additionally, each vertex has been labeled with the corresponding $\delta_{\max}$.

6 A common laminar family

In this section we will prove the existence of a so-called common laminar family over the vertex set of an edge-weighted graph G. A collection $\mathcal{F}$ of subsets of a set S is a laminar family over S if, for any two intersecting sets in $\mathcal{F}$, one set contains the other. That is to say, for each pair S_1, S_2 in $\mathcal{F}$ such that $|S_1| \leq |S_2|$, either $S_1 \cap S_2 = \varnothing$, or $S_1 \subset S_2$.

The common laminar family defines a representation of a tree structure that is common to each MST of G. The notion of a laminar family has been utilized previously by Ravi and Singh [20] for designing an approximation algorithm for constructing a minimum-degree MST.

Semple and Steel [25] note that each rooted phylogenetic tree can be uniquely described as a laminar family over the set of labeled vertices. Laminar family representations of rooted phylogenies are used for comparing and combining information from multiple rooted phylogenetic trees. Later in this section we show that the laminar family representation of an ultrametric tree is equivalent to the common laminar family.

6.1 A structure that is common to all MSTs of a graph

Lemma 2 *Given an edge-weighted graph $G = (V, E)$ with k distinct weight classes $W = \{w_1, w_2, \ldots, w_k\}$, and an MST M of G, let F_i be the forest that is formed by removing all edges in G that are heavier than w_i. Let C_i be the collection comprising the vertex set of each component of F_i. Consider the collection $\mathcal{F}$ which is constructed as follows: $\mathcal{F}_C = \left\{\cup_{i=1}^{k} C_i\right\} \cup V$. The following is true:*

1. $\mathcal{F}_C$ is a laminar family over V
2. Each vertex set in $\mathcal{F}_C$ induces a connected graph in each MST of G

Proof: (i). Consider any two vertex sets V_1 and V_2 in $\mathcal{F}$. Let w_1 and w_2 be the weights of the heaviest edges in the subgraphs of M that are induced by V_1 and V_2, respectively. Let F_1 and F_2 be the forests that are formed by removing all edges in M that are heavier than w_1 and w_2, respectively. Let C_1 and C_2 be the collections comprising the vertex set of each component in F_1 and F_2, respectively.

By construction, we have $V_1 \in C_1$ and $V_2 \in C_2$. Consider the case where $w_1 = w_2$. Since $C_1 = C_2$, it follows that $V_1 \cap V_2 = \varnothing$. If $w_1 \neq w_2$, then without loss of generality, let $w_1 < w_2$. F_2 can be constructed by adding to F_1 all edges in M that are no heavier than w_2. The vertex set of each component in F_1 that is not in F_2 induces a connected subgraph in exactly one component of F_2. If $V_1 \in C_1 \cap C_2$ then $V_1 \cap V_2 = \varnothing$. Otherwise, if $V_1 \in C_1 \backslash C_2$, then V_1 is a subset of exactly one set in C_2. This implies that either $V_1 \subset V_2$, or $V_1 \cap V_2 = \varnothing$. Thus $\mathcal{F}_C$ is a laminar family over V.

(ii). Let V_i be the vertex set of a component in the graph G_i of G that is created by removing all edges in G_i that are heavier than w_i. It follows that V_i

induces a connected graph in each minimum spanning forest of G_i. Consider an MST M of G. Removing all edges in M that are heavier that w_i constructs a minimum spanning forest F of G. Thus V_i induces a connected graph in M. It follows that V_i induces a connected graph in each MST of G. By construction $V_i \in \mathcal{F}_C$. □

6.2 Ultrametric trees

Ultrametric trees are rooted phylogenetic trees that satisfy the condition that the root is equidistant from the leaves. All ultrametric trees are leaf-labeled. Semple and Steel [25] note that the hierarchical structure of a rooted tree can be represented using a laminar family. We show that the laminar family $\mathcal{F}_T$ that represents an ultrametric tree T is equivalent to the laminar family $\mathcal{F}_C$ that common to all the MSTs of the distance graph associated with T.

Lemma 3 *We are given an ultrametric tree T and the corresponding distance graph G. Let $\mathcal{F}_C$ be the laminar family that is common to each MST of G. Let $\mathcal{F}_T$ be the laminar family representation of T. The following is true.*

$$\mathcal{F}_T = \mathcal{F}_C.$$

Proof: Consider a vertex set $S \subset \mathcal{F}_T$. Let w be the largest distance between vertices in S. Consider the forest F that is constructed by removing all edges in G that are heavier than w. S induces a connected component C in F since each pairwise distance between vertices in S is not larger than w. Since the distance between each vertex in S and each vertex in $V_T \backslash S$ is larger than w, it follows that C does not contain any vertex that is not in S. Since the common laminar family $\mathcal{F}_C$ contains the vertex set of each component in F, it follows that $S \subset \mathcal{F}_C$. Since this is true for each set in $\mathcal{F}_T$, it follows that $\mathcal{F}_T = \mathcal{F}_C$. □

Note that the laminar family representation $\mathcal{F}_T$ of a rooted tree, and the corresponding common laminar family $\mathcal{F}_C$, are not equivalent in general. See Fig. 7 for an example.

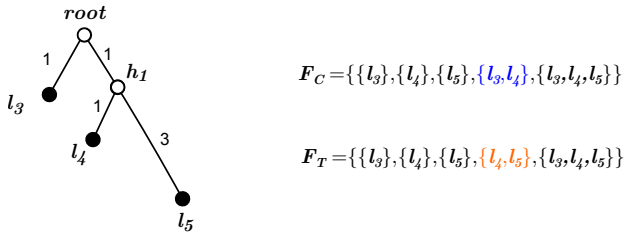

Figure 7: The equivalence between the laminar family representation $\mathcal{F}_T$ of a rooted phylogenetic tree, and the common laminar family $\mathcal{F}_C$, is not true in general.

6.3 An algorithm for constructing the common laminar family and the MST union graph

In this subsection we present an algorithm for constructing the common laminar family and the MST union graph. The MST union graph of a graph G is the subgraph of G that contains all the edges that are present in at least one MST of G.

Algorithm 2 Construct the common laminar family $\mathcal{F}_C$ and the MST union graph G_U.

Input: $G = (V_G, E_G)$
Initialize:
$M = (V_M, E_M) \leftarrow$ singleton graph over V_G
$G_U = (V_U, E_U) \leftarrow$ singleton graph over V_G
$\mathcal{F}_C \leftarrow V_G$
$E_{G\leq} \leftarrow$ edges in E_G that are sorted in order of increasing weight
$w_{\text{previous}} \leftarrow$ weight of the lightest edge in E_G
$V_w \leftarrow \varnothing$
Functions:
$C_M(v)$: Returns the vertex set of the component of M containing v
$F_M(v)$: Returns id of the component of graph M containing v
$U_M(u, v)$: Adds edge $\{u, v\}$ to E_M and updates component ids
for $\{u, v\}$ in $E_{G\leq}$
  $w_{\text{current}} \leftarrow$ weight of $\{u, v\}$
  if $w_{\text{current}} > w_{\text{previous}}$
    for $\{u, v\}$ in E_w
      if $F_M(u) \neq F_M(v)$
        $U_M(u, v)$
    $E_w \leftarrow \varnothing$
    for v in V_w
      $\mathcal{F}_C \leftarrow \mathcal{F}_C \cup C_M(v)$
    $V_w \leftarrow \varnothing$
  else if $F_M(u) \neq F_M(v)$
    $E_U \leftarrow E_U \cup \{\{u, v\}\}$# ensures that G_U contains all the edges that are present in at least one MST of G
    $E_w \leftarrow E_w \cup \{\{u, v\}\}$
    $V_w \leftarrow V_w \cup \{u\}$
  $w_{\text{previous}} \leftarrow w_{\text{current}}$
Output: $\mathcal{F}_C, G_U = (V_U, E_U)$

Lemma 4 *Given an edge-weighted graph $G = (V_G, E_G)$ with k distinct weight classes $W = \{w_1, w_2, \ldots, w_k\}$, the outputs $\mathcal{F}_C$ and G_U of Algorithm 2 are the common laminar family of G, and the MST union graph of G, respectively.*

Proof: Algorithm 2 adds edges to the singleton graph M in order of increasing weight, in such a way that M does not contain any cycles. From Kruskal [16] we know that M is an MST of G.

Consider the forest F_i that is constructed by removing all edges in M that are heavier than w_i. By construction, $\mathcal{F}_C$ includes the vertex set of each component of F_i. Let C_i be the collection comprising the vertex set of each component of F_i. It follows that $\mathcal{F}_C = \left\{\cup_{i=1}^{k} C_i\right\} \cup V$. From Lemma 2, we know that $\mathcal{F}_C$ is the common laminar family of G.

E_U is constructed by adding the lightest edges that are incident to vertices in different components. The cut property of MSTs states that given a graph $G = (V, E)$, for each pair V_1, V_2 of disjoint sets such that $V_1 \cup V_2 = V$, each MST of G contains one of the lightest edges which have one endpoint in V_1 and the other endpoint in V_2. It follows that each edge in E_U is present in at least one MST of G. □

7 Minimum leaves VMSTs

7.1 Implicitly selecting optimal surrogate vertices

Lemma 5 *We are given a phylogenetic tree T, the corresponding distance graph $G = (V, E)$. Let $\mathcal{F}_C$ be the common laminar family of G. Let $G_U = (V_U, E_U)$ be the MST union graph of G. Let h be a hidden vertex in T such that there is a leaf l in $S(h)$, and h is incident to l. Let V_i be a vertex set in $\mathcal{F}$ and let w_i be the corresponding edge weight. Then the following is true:*

1. Let $N(v)$ be the set of all vertices that are adjacent to vertex v in G_U. Let $C(v)$ be a smallest sub-collection of $\mathcal{F}$ that covers $N(v)$ but not v. Among all MSTs, the maximum vertex degree $\delta_{\max}(v)$ of v is $|S(v)|$.

2. $\delta_{\max}(l) \leq \delta_{\max}(v)$ for each vertex v in $S(h)$

Proof: (i). Let $N(v) = \{j_1, j_2, \ldots, j_k\}$ be the neighbors of v in G_U. Let M be an MST of G. Let $C(v) = \{c_1, c_2, \ldots, c_m\}$ be a smallest sub-collection of $\mathcal{F}$ that covers $N(v)$ and does not include v.

Let $C(v)$ contain a set c_i that covers multiple vertices in $N(v)$. Let j_1 and j_2 be any two vertices in c_i. Let w_i be the heaviest weight on the path between j_1 and j_2 in M. The edges $\{v, j_1\}$ and $\{v, j_2\}$ are heavier than w_i. If they were not, then we would have $v \in c_i$. Since v, j_1 and j_2 are on a common cycle, each MST of G can only contain one of the two edges $\{v, j_1\}$, and $\{v, j_2\}$. It follows that, for each set $c_i \in C(v)$, each MST can contain at most one edge which is incident to v and to a vertex in c_i. Thus the maximum number of edges that can be incident to v in any MST is the number of vertex sets in $C(v)$, i.e., $\delta_{\max}(v) = |C(v)|$.

(ii). Let $N(l)$ and $N(v)$ be the set of all vertices that are incident to l and v in G_U, respectively. Let $j \in N(l) \backslash S(h)$. The weight of the edge $\{j, l\} \in E_U$ is given by d_{jl}. $d_{jh} > d_{vh}$ since $j \notin S(h)$. Thus $d_{lj} > d_{lv}$, and consequently $v \in N(l)$. We have $d_{jl} = d_{jh} + d_{hl} = d_{jh} + d_{hv} = d_{jv}$. Consider the MST $M = (V_M, E_M)$ that contains the edges $\{l, v\}$ and $\{l, h\}$. Consider the spanning tree M' that is formed by removing $\{l, h\}$ from E_M and adding $\{v, h\}$. M' and M have the same sum of edge weights. Thus we also have $j \in N(v)$. Consequently $N(l) \subseteq N(v)$. Let $C(l)$ and $C(v)$ be the smallest sub-collections of $\mathcal{F}$ such that $C(l)$ covers $N(l)$ but does not contain l, and $C(v)$ covers $N(v)$ but does not contain v. $C(v)$ covers both $N(l)$ and $N(v)$ since $N(l) \subseteq N(v)$. Thus $|C(l)| \leq |C(v)|$. From part (i), we know that $|C(l)| = \delta_{\max}(l)$ and $|C(v)| = \delta_{\max}(v)$. Thus $\delta_{\max}(l) \leq \delta_{\max}(v)$. □

7.2 Constructing a VMST with the minimum number of leaves

Algorithm 3 Construct a minimum leaves VMST (MLVMST)

Input: $G = (V, E)$
$\mathcal{F}_C \leftarrow$the common laminar family of G
$\mathcal{F}_C^{\supseteq} \leftarrow$sets of $\mathcal{F}_C$ ordered in order of decreasing size
$G_U \leftarrow$the MST union graph of G
$\delta_{max} \leftarrow$empty array
for i in V
  $N_i \leftarrow$neighbors of i in G_U
  $\delta_{max}(i) \leftarrow 0$
  for C in $\mathcal{F}_C^{\supseteq}$:
    if $C \cap N_1 \neq \varnothing$ and $C \cap \{i\} = \varnothing$
      $\delta_{max}(i) \leftarrow \delta_{max}(i) + 1$
      $N_i \leftarrow N_i \backslash C$
$<_* \leftarrow$ A total order over V such that $u <_* v \implies \delta_{max}(u) \leq \delta_{max}(v)$
$M_* \leftarrow$ VMST constructed by applying Algorithm 1 to $(G, <_*)$
Output: M_*

Theorem 1 *We are given a phylogenetic tree T and the corresponding distance graph G. Let M be the vertex order based MST that is computed using Algorithm 3. Among all VMSTs of G, M has the minimum number of leaves.*

Proof: Let $S(h)$ be the set of vertices that are closest to h wrt the tree metric d_T that is associated with T. From Lemma 5(ii), we know that, if there is a leaf l in $S(h)$, then among all vertices in $S(h)$, $\delta_{\max}(l)$ is smallest. By construction of $<_*$, among all vertices in $S(h)$, l is the smallest vertex wrt $<_*$. It follows that Algorithm 3 implicitly selects l as the surrogate vertex of h. Since each leaf in T is adjacent to at most one hidden vertex, the vertex order that is selected by Algorithm 3, maximizes the number of distinct leaves that are selected as

surrogate vertices. Contracting the path in T between a hidden vertex and the corresponding surrogate vertex increases the degree of the surrogate vertex. Thus, among all vertex order based MSTs, M has the minimum number of leaves. □

7.3 Implementation details and time complexity analysis

Algorithm 3 takes as input an edge-weighted graph $G = (V, E)$ and performs the following actions. First, the common laminar family $\mathcal{F}_C$ and the MST union graph G_U are constructed by applying Algorithm 2 to G. Subsequently, a vertex order $<_V$ is computed on the basis of $\mathcal{F}_C$ and G_U. Finally, a VMST is constructed by applying Algorithm 1 to $(G, <_V)$.

Algorithms 1 and 2 are variants of Kruskal's algorithm and were implemented using a disjoint-set data structure with balanced Union, and Find with path compression [26]. The functions F_M and U_M correspond to a Find operation and a Union operation, respectively. A disjoint-set data structure can be represented as a forest with self-loops and directed edges. Each vertex points to its parent. The root of a component points to itself. A Find operation on a vertex repoints the edge to its former parent to the root of the component containing the vertex. A Union operation takes as input the roots of two components and creates an edge pointing from the root of the smaller component to the root of the larger component. The function $C_M(u)$ is designed to return the set of vertices that are in the same component as u. C_M is implemented as follows. We store the vertex set of a component in the root of the component. Each time we perform a union operation $U_M(r_1, r_2)$ we combine the vertex sets and store the combined vertex set in the root of the component containing $r_1$and r_2.

The main steps of Algorithms 1 and 2 are (i), sorting $O(n^2)$ edges and, (ii), performing $O(n^2)$ Find operations and $O(n)$ Union operations, where n is the number of vertices in V. Step (i) can be done using mergesort in time $O(n^2 \log n^2)$=$O(n^2 \log n)$. Step (ii) takes time $O(n^2\alpha(n^2, n))$ where α is the inverse of Ackermann's function [26]. Since $\alpha(n^2, n) < \log n$, both the algorithms complete their computations in time $O(n^2 \log n)$.

In addition to calling Algorithms 1 and 2, Algorithm 3 sorts the sets in $\mathcal{F}_C$ and computes $\delta_{\max}$ for each vertex in V. $\mathcal{F}_C$ has $O(n)$ sets which can be sorted using mergesort in time $O(n \log n)$. For each vertex, $\delta_{\max}$ can be computed in time $O(n)$.

Thus the total time complexity of Algorithm 3 is $O(n^2 \log n)$.

Acknowledgements

We thank Erik Jan van Leeuwen and Davis Isaac for helpful discussions during the early stages of this work. We thank the reviewers for their constructive criticism of the first version of the manuscript.

References

[1] S. Bastkowski, V. Moulton, A. Spillner, and T. Wu. The minimum evolution problem is hard: a link between tree inference and graph clustering problems. *Bioinformatics*, 32(4):518–522, 2016. doi:10.1093/bioinformatics/btv623.

[2] M. J. Choi, V. Y. F. Tan, A. Anandkumar, and A. S. Willsky. Learning latent tree graphical models. *Journal of Machine Learning Research*, 12:1771–1812, 2011.

[3] B. Chor and T. Tuller. Finding a maximum likelihood tree is hard. *Journal of the Association for Computing Machinery*, 53(5):722–744, 2006. doi:10.1145/1183907.1183909.

[4] C. Chow and C. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, 14(3):462–467, 1968. doi:10.1109/TIT.1968.1054142.

[5] W. H. E. Day. Computational complexity of inferring phylogenies from dissimilarity matrices. *Bulletin of Mathematical Biology*, 49(4):461–467, 1987. doi:10.1007/BF02458863.

[6] R. Desper and O. Gascuel. Theoretical foundation of the balanced minimum evolution method of phylogenetic inference and its relationship to weighted least-squares tree fitting. *Molecular Biology and Evolution*, 21(3):587–598, 2004. doi:10.1093/molbev/msh049.

[7] M. dos Reis, P. C. J. Donoghue, and Z. Yang. Bayesian molecular clock dating of species divergences in the genomics era. *Nature Reviews Genetics*, 17(2):71–80, 2016. doi:10.1038/nrg.2015.8.

[8] S. Fiorini and G. Joret. Approximating the balanced minimum evolution problem. *Operations Research Letters*, 40(1):31–35, 2012. doi:10.1016/j.orl.2011.10.003.

[9] L. R. Foulds and R. L. Graham. The Steiner problem in phylogeny is NP-complete. *Advances in Applied Mathematics*, 3(1):43–49, 1982. doi:10.1016/S0196-8858(82)80004-3.

[10] O. Gascuel and M. Steel. Neighbor-joining revealed. *Molecular Biology and Evolution*, 23(11):1997–2000, 2006. doi:10.1093/molbev/msl072.

[11] S. B. Hedges, J. Dudley, and S. Kumar. TimeTree: a public knowledge-base of divergence times among organisms. *Bioinformatics*, 22(23):2971–2972, 2006. doi:10.1093/bioinformatics/btl505.

[12] S. B. Hedges, J. Marin, M. Suleski, M. Paymer, and S. Kumar. Tree of life reveals clock-like speciation and diversification. *Molecular Biology and Evolution*, 32(4):835–845, 2015. doi:10.1093/molbev/msv037.

[13] F. Huang, U. N. Niranjan, I. Perros, R. Chen, J. Sun, and A. Anandkumar. Scalable latent tree model and its application to health analytics. *arXiv preprint*, arXiv:1406.4566, 2014.

[14] T. H. Jukes and C. R. Cantor. Evolution of protein molecules. In H. N. Munro, editor, *Mammalian Protein Metabolism*, pages 21–132. Academic Press, New York, 1969. doi:10.1016/B978-1-4832-3211-9.50009-7.

[15] P. Kalaghatgi, N. Pfeifer, and T. Lengauer. Family-joining: a fast distance-based method for constructing generally labeled trees. *Molecular Biology and Evolution*, 10(33):2720–2734, 2016. doi:10.1093/molbev/msw123.

[16] J. B. Kruskal. On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings of the American Mathematical Society*, 7(1):48–50, 1956. doi:10.2307/2033241.

[17] S. Kumar and S. B. Hedges. TimeTree2: species divergence times on the iPhone. *Bioinformatics*, 27(14):2023–2024, 2011. doi:10.1093/bioinformatics/btr315.

[18] L. Pozzi, J. A. Hodgson, A. S. Burrell, K. N. Sterner, R. L. Raaum, and T. R. Disotell. Primate phylogenetic relationships and divergence dates inferred from complete mitochondrial genomes. *Molecular Phylogenetics and Evolution*, 75:165–183, 2014. doi:10.1016/j.ympev.2014.02.023.

[19] M. N. Price, P. S. Dehal, and A. P. Arkin. Fasttree 2 –approximately maximum-likelihood trees for large alignments. *PloS ONE*, 5(3):e9490, 2010. doi:10.1371/journal.pone.0009490.

[20] R. Ravi and M. Singh. Delegate and conquer: an LP-based approximation algorithm for minimum degree MSTs. In *Proceedings of the 33rd International Colloquium on Automata, Languages and Programming*, pages 169–180, 2006. doi:10.1007/11786986_16.

[21] D. F. Robinson and L. R. Foulds. Comparison of phylogenetic trees. *Mathematical Biosciences*, 53(1-2):131–147, 1981. doi:10.1016/0025-5564(81)90043-2.

[22] S. Roch. A short proof that phylogenetic tree reconstruction by maximum likelihood is hard. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 3(1):92–94, 2006. doi:10.1109/TCBB.2006.4.

[23] N. Saitou and M. Nei. The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Molecular Biology and Evolution*, 4(4):406–425, 1987. doi:10.1093/oxfordjournals.molbev.a040454.

[24] G. Salamon and G. Wiener. On finding spanning trees with few leaves. *Information Processing Letters*, 105:164–169, 2008. doi:10.1016/j.ipl.2007.08.030.

[25] C. Semple and M. Steel. *Phylogenetics*, volume 24 of *Oxford Lecture Series In Mathematics And Its Applications*. Oxford University Press, 2003.

[26] R. E. Tarjan. Efficiency of a good but not linear set union algorithm. *Journal of the Association for Computing Machinery*, 22(2):215–225, 1975. doi:10.1145/321879.321884.