Most Compact Parsimonious Trees

Dimitris Papamichail\textsuperscript{1}, Angela Huang\textsuperscript{1}, Andrew Miller\textsuperscript{1}, Nathan Gould\textsuperscript{1}, and
Georgios Papamichail\textsuperscript{2}

\textsuperscript{1} The College of New Jersey, Ewing, NJ, USA
papamicd@tcnj.edu
\textsuperscript{2} New York College, Athens, Greece

Abstract Construction of phylogenetic trees has traditionally focused on binary rooted trees where all species are placed on leaves. Certain application domains though, such as viral evolution and transmission, paleontology, linguistics, and phylogenetic stemmatics, often require phylogeny inference that involves placing extant ancestors on internal tree nodes, and polytomies/multifurcations. In this article we describe algorithms to compute the most compact parsimonious trees that infer the phylogeny of a set of species, without resorting to exhaustive enumeration of all tree topologies.

Keywords: phylogenetics, maximum parsimony

1 Introduction

Phylogeny is the evolutionary history of a set of species whose relationships are often represented by a tree. Phylogenetic trees can be rooted or unrooted, and their branches are labelled with lengths that correspond to evolutionary distances between species. Theoretical methods for inferring phylogenies have existed for many decades and have been applied to data spanning a wide range of domains. The motivation for our work stems from its perceived applications in virology, paleontology, and stemmatics research.

There are several prominent tree construction methods for evolutionary inference. Distance Matrix methods involve the generation of dissimilarity matrices based on well-defined distances, and the application of clustering algorithms such as Neighbor-Joining \cite{28}. Maximum Parsimony is a method that uses events (characters) instead of distances, associates a cost with each event, and aims to build a tree with the smallest possible cost. Maximum Likelihood and Bayesian methods involve assigning probabilities to events and probability distributions to possible tree configurations, aiming to create trees whose observed phylogeny has the highest likelihood. In this study we will use the Maximum Parsimony method with unordered character states, a well studied, simple, and intuitive criterion.

In the past, Maximum Parsimony has been used to successfully infer phylogenies in empirical studies where the true phylogeny was known and observed.
Using this method, Bush et al. were successful at predicting the correct phylogeny for the human Influenza A virus 83 percent of the time for flu seasons spanning 1983-1994 [7]. In a study of the propagation of Bacteriophage T7 in the presence of a mutagen, the parsimony method effectively identified the correct lineage out of 135,135 possible phylogenetic trees [22]. Phylogenetic analysis has also successfully predicted the viral ancestor of 111 modern HIV-1 (AIDS virus) sequences. The predicted ancestor was matched with an actual ancestor from a sample collected and archived in 1959 [39].

Existing phylogenetic methods have primarily focused on fully bifurcating trees, where all extant species are placed on the leaves of the tree. However, it is often the case that extant species can appear as ancestors on internal nodes. The ability to identify known common ancestors using molecular data has been successfully demonstrated with the Ebolavirus and Marburgvirus genera [9]. Patterns of evolution of HIV within patients have been shown to detect emergence of specific strains [6], using serial evolution networks, which resemble trees with extant ancestor nodes. In the area of paleontology, it is often the case that ancestors of species may be known and well characterized, prompting the need for phylogenetic reconstruction methods that account for labeled internal nodes. Notably, the fossil record is incomplete, and it does not provide a high guarantee of recording the common ancestor of species [17]. However, there are certain species where the fossil record has been extensively studied and extinct common ancestors are highly known, such as the case for graptolites (e.g. [24,36]). Existing efforts to build trees which incorporate known ancestors, such as the paleotree package [4], can greatly benefit from the algorithmic methods presented in this paper.

In the area of textual criticism, attempts are often made to reconstruct an archetype of a given work, given multiple versions from different regions and time periods [23]. The relations of these documents can be represented as a tree, called “stemma”, where the root is the original archetype, and branching represents the creation of a copy. Textual criticism naturally resembles phylogenetic analysis, but there are some interesting differences. One is the lack of a uniform rate of mutation over time, which can be partially ameliorated by estimation of the time period of the creation of a copy. Polytomy (multifurcation) is also common in stemmata, as is the presence of extant documents (witnesses) as ancestors of other extant as well as hypothetical documents. Exploration of traditional phylogenetic methods for stemmatics has been done in recent years (e.g. [29,25,2], including generation of faithful stemmata using maximum parsimony [32] for stemmata with 21 witnesses, and maximum likelihood [27], using structural Expectation Maximization (EM). This last work also represents the first effort to handle multifurcation, albeit using split decomposition [12], which has been shown to represent manuscript relationships ambiguously, creating unclear stemmata [32].

Polytomies (multifurcation) and species placement on internal tree nodes are features that, even when desirable, are often ignored in existing phylogenetic methods. One reason may be the additional complexity they add to an already
hard computational problem [20,10,11,8]. With this work, we aim to explore the construction of maximum parsimony trees that allow for multifurcation and internal “species” nodes. The promotion of species from leaves to internal nodes leads to more compact trees. Minimization of the number of nodes in a tree with \( n \) species becomes now an additional parsimony criterion to the number of character state mutations along the edges, as we aim to create the most compact parsimonious tree.

The rest of the paper is structured as follows: In section 2 we provide terminology for most terms encountered in this paper. Section 3 examines the number of phylogenetic trees with \( n \) species that make up our search space, and compares its magnitude to the number of cubic trees with \( n \) species, which is explored in traditional phylogenetic algorithms. In section 4 we describe the algorithms of Fitch and Hartigan, which solve the small parsimony problem, and adapt the latter from rooted to unrooted trees. In section 6 we describe an algorithm to find the most compact parsimonious tree using edge contraction. We conclude with observations and discussion in section 7.

2 Definitions

2.1 Graph theoretical terms

Graph theoretical definitions often vary in literature, so we define most of the terms used in this manuscript below:

An undirected graph is a pair \( G = (V, E) \), where \( V \) is a set of nodes (vertices) and \( E \) is a set of edges (branches) that connect nodes. An edge \( e \) is modelled as a set of two nodes \( \{v_1, v_2\} \), with \( v_1 \neq v_2 \). Edge \( e \) is then said to be incident to \( v_1 \) and \( v_2 \), and the nodes \( v_1 \) and \( v_2 \) are adjacent. The degree of a node \( v \) is the number of edges incident to \( v \). A path is a sequence of nodes \( v_1, v_2, \ldots, v_n \), where \( v_i \) and \( v_{i+1} \) are connected by an edge \( \forall i : 1 \leq i \leq n - 1 \).

For the purposes of our study, a tree is a connected acyclic undirected graph. A leaf is a node of degree one. All other nodes are internal and have a degree of at least two. An unrooted (or free) tree is as defined above, where a rooted tree has one distinguishable vertex called the root. The process of adding a root to an unrooted tree induces a hierarchical relationship on the nodes, implying a direction for each edge, which points away from the root. The hierarchy establishes an ancestor-descendant relationship among nodes. Every node on the path from a node \( v \) to the tree root is an ancestor of \( v \), and the first such node (which is incident to \( v \)) is the parent of \( v \). Nodes on the paths from a node \( v \) away from the root are descendants of \( v \). The children of \( v \) are descendants of \( v \) which are incident to \( v \). Each edge divides the tree into two subtrees. Given a node \( v \) other than the root in a rooted tree and disconnecting the edge incident to \( v \) and its parent, we derive the subtree rooted at \( v \). The subtree rooted at the root is the complete original tree.

A rooted tree where all nodes have a maximum degree of 3 is called a binary or bifurcating tree. If all internal nodes except for the root have a degree of 3
(one parent and two children) then the rooted tree is called a full binary tree. An unrooted tree where all nodes have either a degree of 1 (leaves) or 3 (internal nodes) is called a cubic tree, following the terminology of [13]. A tree whose nodes can have degrees higher than 3 is called multifurcating. Nodes in a tree can be labelled, i.e., assigned values. A tree with labelled leaves has values assigned to all of its leaves. In our study we will define a mixed labelled tree (or mixed tree) to be a tree where all leaves are labelled, and internal nodes may be labelled.

2.2 Node labels, characters and states

The following definitions follow to a large extent the terminology in [21]. Let $S$ be a set of $n$ objects $\{S_1, S_2, \ldots, S_n\}$. We will refer to these objects as species, independent of whether they represent organisms, documents, or any other kind of objects whose evolutionary relationships we are attempting to infer.

Each species has a set of $m$ ordered features $C$, which we will call characters. In organisms, such characters can be observed heritable traits, often morphological or genomic. In the comparison of documents, characters often are words or sections of manuscripts. Each character can take a constant number of values, called states. For example, a DNA character can have four states, $\{A, C, G, T\}$, where an amino acid character can have 20. Each species $S_i$ will be a fixed tuple of $m$-character states $(C(i)_1, C(i)_2, \ldots, C(i)_m)$. For our study each character $C_i$, $1 \leq i \leq m$ can have a different fixed number of states. Character states are unordered (Fitch parsimony).

Species can be assigned to nodes in a tree, which are then labelled. As such, every labelled (species) node in a tree will have an $m$-tuple of character states associated with it, which will be the value $V$ of the node. Each labelled node $v$ will also have a root set $VV$ associated with it, which is an $m$-tuple of character state singleton sets. For example, a labelled node $v_i$ corresponding to species $S_j$ will have a value $V(i) = (A, B, \ldots, Z)$ and a root set $VV(i) = (\{A\}, \{B\}, \ldots, \{Z\})$, where $C(j)_1 = A, C(j)_2 = B, \ldots, C(j)_m = Z$. Unlabelled nodes in the tree will also have root sets $VV$, whose state sets can contain more than one states. If an unlabelled node $u$ is assigned a single state for each character, then this assignment will become the value of the unlabelled node, called a fit, with $f \in VV(u)_1 \times VV(u)_2 \times \cdots \times VV(u)_m$. An unlabelled node with an assigned value will be called a fitted node. A tree fit is a assignment of values to all unlabelled nodes in the tree. When representing the root set $VV$ of a node, we will use the shorthand notation $A/BCD/EF/\ldots/YZ$ instead of the set notation $\{(A), \{B, C, D\}, \{E, F\}, \ldots, \{Y, Z\}\}$.

A mutation is a change between states of a character. A single mutation will carry a unit cost. A distance $d(v, u)$ between two connected nodes $v$ and $u$ in a tree is defined as the sum of all mutations among all characters in the values of the nodes. The distance between two adjacent nodes $v$ and $u$ is assigned to the edge $(v, u)$. Let $nei(x, y): X_i \times X_i \to \{0, 1\}$, where $X_i$ is the powerset of the states of character $C_i$, be a function such that

$$nei(x, y) = \begin{cases} 
0 & \text{if } x \cap y \neq \emptyset \\
1 & \text{otherwise}
\end{cases}$$
The minimum distance $md(u, v)$ between two adjacent nodes $u$ and $v$ is defined as

$$md(u, v) = \sum_{1 \leq i \leq m} nei_i(VV(u)_i, VV(v)_i)$$

The potential cost of an edge $(v, u)$ is the number of mutations between a pair of fits for $v$ and $u$. The cost of an edge $(v, u)$ is the number of mutations between the values of $v$ and $u$. The minimum cost or min-cost of an edge $(v, u)$, denoted by $mc(v, u)$, is defined as the minimum number of mutations between all pairs of fits between $v$ and $u$, and is equal to $md(v, u)$. The cost of a tree for a given fit is the sum of costs along the tree’s edges. The most parsimonious cost (MP-cost) of a tree is the minimum sum of potential costs along all of its edges for any tree fit. An MP-cost tree fit is called a best fit.

3 Enumerating mixed trees

According to Flight [16], there are $\sum_{m=0}^{n-2} T(n, m)$ mixed labelled trees, where all leaf nodes are labelled, and internal nodes may be labelled, where $T(n, m)$ is the number of unique trees with $n$ labelled nodes and $m$ unlabelled nodes. Observably, there are four different ways to construct a tree with $n$ labelled species from a tree with $n-1$ labelled species, allowing multifurcations:

1. Insert an unlabelled node into any of the $n + m - 3$ edges of any of the $T(n-1, m-1)$ trees and have the $n$th labelled node descend from it.
2. Insert the labelled node directly into any of the $n + m - 2$ edges of any of the $T(n-1, m)$ trees.
3. Make the labelled node the child of any of the $n + m - 1$ available nodes belonging to any of the $T(n-1, m)$.
4. Label any of the $m + 1$ unlabelled nodes in any of the $T(n+1, m+1)$ trees.

This leads to the following recurrence:

$$T(n, m) = \begin{cases} 
(m + n - 3) \cdot T(n-1, m-1), & m > 0 \\
(2n + 2m - 3) \cdot T(n-1, m) & + \\
(m + 1) \cdot T(n-1, m+1), & n > m + 2
\end{cases}$$

The base case of this recurrence is: $T(1, 0) = 1$ and $T(1, i) = 0 \forall i, i > 0$. Utilizing N.J.A. Sloane’s Online Encyclopedia of Integer Sequences [31] we identified the following closed form formula for the number of trees as a function of the labelled nodes $n$:

$$\frac{n^{n-2}}{\sqrt{2e^2 \left(2 - e^{\frac{1}{2}}\right)^{n-\frac{1}{2}}}$$

From observing the growth rate of this function, it is clear that an exhaustive search on all possible trees is only practical for small numbers of $n$. Already, the exhaustive search method on cubic trees with labelled leaves and
unlabelled internal nodes is computationally impractical even for small values of $n$ [35]. Comparatively, the number of mixed multifurcating trees grows at a hyper-exponentially faster rate, as can be seen in Fig. 1. This motivates a need for an alternative method to exhaustive enumeration of all $n$-species trees.

4 Maximum Parsimony for trees with labelled leaves

According to the parsimony criterion, we seek a tree that explains divergence of species with the fewest number of evolutionary events. As such, we seek to identify a tree with $n$ labelled nodes and fitted unlabelled nodes such that the tree cost, which is the sum of edge costs and therefore the total number of mutations, is minimized. This problem can be broken into two subproblems.

- **Small parsimony problem (SPP):** Given a tree $\tau$ with $n$ species nodes and a specified topology, identify a tree fit such that the tree cost is minimized (MP-cost).
- **Large parsimony problem (LPP):** Given a set of $n$ species, find the tree(s) with the MP-cost among all possible tree topologies. Such tree(s) is/are called the most parsimonious tree(s) (MP-trees).

4.1 Fitch’s algorithm

Fitch described an algorithm to solve the SPP and calculate the MP-cost and a best fit of a given $n$-species rooted binary tree where all species are placed
on leaf nodes [15]. The MP-cost is calculated in a bottom-up fashion, whereas a best fit is determined in a top-down traversal. Procedurally, in the bottom-up step, for every node \( u \) with children \( v \) and \( w \) in an input tree, the algorithm calculates a state set \( VV( u ) \) and the MP-cost \( c_u \) of the subtree rooted at \( u \) as seen in Algorithm 1.

**Algorithm 1** Fitch’s bottom up procedure

1: procedure \textsc{Fitch} \\
2: \hspace{1em} count = 0; \\
3: \hspace{1em} for each \( C_i, 1 \leq i \leq m \) do \\
4: \hspace{2em} if \( u \) is a leaf then \\
5: \hspace{3em} Initialize \( VV(u) \), \\
6: \hspace{3em} count += 0; \\
7: \hspace{2em} \text{else} \\
8: \hspace{3em} \hspace{1em} \hspace{1em} c_v = \textsc{Fitch}(v); \\
9: \hspace{3em} \hspace{1em} \hspace{1em} c_w = \textsc{Fitch}(w); \\
10: \hspace{3em} \hspace{1em} \hspace{1em} \text{if } VV(u)_i \cap VV(w)_i \neq \emptyset \text{ then} \\
11: \hspace{3em} \hspace{3em} \text{count } + = c_v + c_w; \\
12: \hspace{3em} \hspace{1em} \hspace{1em} \text{else} \\
13: \hspace{3em} \hspace{3em} \hspace{1em} VV(u)_i = VV(v)_i \cup VV(w)_i; \\
14: \hspace{3em} \hspace{3em} \text{count } + = c_v + c_w + 1; \\
15: \hspace{1em} \text{return count;}

The top down step assigns a fit to the tree. It starts by fitting the root with an \( m \)-tuple value populated with arbitrary states from its root set, then updating the subtrees recursively. The value of a node \( u \) with parent \( p \) is determined in Algorithm 2.

**Algorithm 2** Fitch’s top-down fitting

1: for each \( C_i, 1 \leq i \leq m \) do \\
2: \hspace{1em} \text{if } VV(u)_i \cap VV(p)_i \neq \emptyset \text{ then} \\
3: \hspace{2em} Select arbitrary \( V(u)_i \in VV(u)_i \cap VV(p)_i; \) \\
4: \hspace{1em} \text{else} \\
5: \hspace{2em} Select arbitrary \( V(u)_i \in VV(u)_i; \)

An example of Fitch’s steps in evaluating the root sets of the nodes of a given tree is shown in Fig. 2(a).

4.2 Advantages and Disadvantages of Fitch’s algorithm

Fitch’s algorithm optimally solves the SPP for full binary rooted trees with labelled leaves [15]. It also identifies a most parsimonious fit for a given full binary
Figure 2. (a) Computation of character state sets of all unlabelled nodes. (b) Node $u_2$ is assigned value B, not in its state set, resulting in a most parsimonious tree fit.

tree topology. It is simple, efficient, and amenable to bit-operation optimization and other improvements that can speed up the large parsimony search [26, 38].

Fitch’s algorithm cannot solve the SPP on multifurcating trees, or unrooted trees without the addition of an arbitrary root of degree 2. Furthermore, computed state sets at each unlabelled node are not necessarily the root sets of these nodes, meaning there can exist best tree fits that the Fitch algorithm does not identify. An example of such a fit in a single character tree is shown in Fig. 2(b), where node $u_2$ can be assigned state B for a best fit, but that state is not identified by Fitch’s algorithm (and is not included in the node’s state set, as seen in Fig. 2(a)). As we will see in the next subsection, Hartigan’s algorithm allows for the identification of all most parsimonious tree fits, as well as handles multifurcation, with only constant time and space additional worst case complexity requirements.

4.3 Hartigan’s algorithm

Hartigan’s algorithm provides a more powerful framework for calculating a best fit for a given tree. It solves the SPP for multifurcating rooted trees with $n$ labelled leaves [21]. The bottom-up procedure of Hartigan’s algorithm processes every unlabelled internal node $u_i, 1 \leq i \leq n - 2$ that has children $v_i, i \geq 2$. The procedure recursively calculates upper $VU(u_i, p_{u_i})$ and lower $VL(u_i, p_{u_i})$ sets on every unlabelled node as shown in Algorithm 3.

Hartigan’s top down refinement allows the computation of optimal assignments to each node. For the root node, selecting any of the candidate states from the root set would yield a most parsimonious labelling. The algorithm then proceeds to compute the root sets of internal nodes $u_i$ as determined using Algorithm 4.

By storing all optimal and next-to-optimal values in sets $VU(u)$ and $VL(u)$ respectively, and by computing $VV(u)$, Hartigan’s algorithm can be used to find
**Algorithm 3** Hartigan’s bottom up procedure

1: procedure HARTIGAN
2: count = 0;
3: for each $C_i, 1 \leq i \leq m$ do
4:    if $v_i$ is a leaf then
5:        Initialize $VU(v_i)$;
6:        $VL(v_i) = \emptyset$;
7:        count += 0;
8:    else
9:        $num(v_i) = 0$;
10:       for each state $x_j \in C_i$ do
11:           $num(x_j) = 0$;
12:          for each child $v_i$ do
13:             if $x_j \in VU(v_i)$ then
14:                $num(x_j) += 1$;
15:             $K = \max(num(x_j))$;
16:             $VU(v_i) = \{x_j | num(x_j) = K\}$;
17:             $VL(v_i) = \{x_j | num(x_j) = K - 1\}$;
18:             count += $Hartigan(v_i) + num(v_i) - K$;
19: return count;

all co-optimal solutions to the SPP. An example of Hartigan’s algorithm can be seen in Fig. 3. In this example we can observe the same tree as in Fig. 2, where state $B$ is now included in the root set of node $u_2$.

It is important to note that Fitch’s and Hartigan’s algorithms are restricted to unit cost character mutations. Other algorithms exist for weighted costs, such as Sankoff’s algorithm [30], which allows for edge independent, arbitrary metric costs that may vary for each evolutionary event.

4.4 Unrooting trees

Tree enumeration for the LPP involves the systematic generation of cubic trees, for which MP-costs are computed by arbitrarily rooting the trees. To maintain bifurcation, a root can be added to a tree by replacing an edge $(v_1, v_2)$ with a

**Algorithm 4** Hartigan’s top down fitting

1: for each $C_i, 1 \leq i \leq m$ do
2:    if $VV(u_i) \in VU(v_i)$ then
3:        $VV(v_i) = VV(u_i)$;
4:    else
5:        if $VV(u_i) \in VL(v_i)$ then
6:            $VV(v_i) = VV(u_i) \cup VU(v_i)$;
7:        else
8:            $VV(v_i) = VU(v_i)$;

new unlabelled root node $r$ and two edges $(r, v_1)$ and $(r, v_2)$. It is evident that the cost of the new tree will remain unaltered, since the root node can be assigned the same root set and value as one of either $v_1$ or $v_2$.

Conversely, the following theorem also holds true:

**Theorem 1.** Removing the root of a binary tree, as well as any unlabelled node of degree 2, does not change the most parsimonious cost of the tree.

**Proof.** Executing Hartigan’s algorithm on a rooted binary tree will compute the root sets of all internal nodes, including the root set $VV(r)$ of the root node $r$. Let $VV(x)$ and $VV(y)$ be the corresponding root sets of the root’s children $x$ and $y$. Any assignment of a state $S_i \in VV(r)_j$ to the character $C_j$ of the root node will result in

1. A cost of 0 for mutating this character from the root to both children, if $VV(r)_j \in (VV(x)_j \cap VV(y)_j)$ or
2. A cost of 1 otherwise (when $VV(r)_j \in (VV(x)_j \cup VV(y)_j)$).

Removing the root and connecting nodes $x$ and $y$ directly with an edge will not cause an increase to the MP-cost of the tree, as the same assignments that would minimize the cost of the transitions from the root to its children will now be maintained in the transition from $x$ to $y$, meaning 0 for each character $j$ whose state does not mutate ($VV(x)_j \cap VV(y)_j \neq \emptyset$), and 1 when the state mutates.

Therefore, cubic trees with $n$ labelled leaves share the same MP-cost with binary rooted counterparts (not necessarily full).
5 Hartigan’s algorithm implementation for unrooted trees

Hartigan’s algorithm can compute a most parsimonious fit for a given multifurcating rooted tree with $n$ labelled leaves, as well as the root set of all unlabelled nodes in $\Theta(n)$-time steps. Extending Hartigan’s algorithm to unrooted trees involves only rooting the tree arbitrarily at an internal unlabelled node $v_r$, without splitting any node. The algorithm then performs two depth first (DFS) traversals of the given unrooted tree. Using Hartigan’s bottom up procedure, it post-orderly calculates the upper $V_U(u_i, p_{u_i})$ and lower $V_L(u_i, p_{u_i})$ sets of every unlabelled node $u_i$, $1 \leq u_i \leq n - 2$ from the view of each node’s parent $p_{u_i}$. This first traversal ends with the calculation of the root set of the root node $V_R(v_r)$, since upper and lower sets for all children of $v_r$ from the view of $v_r$ have been computed.

During the second DFS traversal, $V_V$ parent information can flow down the tree from the root $v_r$. In pre-order we can now compute now the $V_V(u_i)$ sets of every unlabelled internal node $u_i$. The computation of sets is depicted graphically in Fig. 4.

6 Towards a compact most parsimonious tree

Our ultimate goal is to find the most compact parsimonious $n$-species tree. To solve this problem, in this section we will demonstrate that it is sufficient to find the most parsimonious cubic $n$-species trees and contract them. Towards that goal we will prove that the most compact parsimonious $n$-species tree cannot have a lower cost fit than the most parsimonious $n$-species cubic tree. To prove this claim we will utilize the following lemmas:
Lemma 1. An $n$-species MP-tree with labelled internal nodes cannot have a lower cost than an $n$-species MP-tree with $n$ labelled leaves.

Proof. We will prove by construction, while maintaining the invariant of lowest tree cost. Consider an $n$-species MP-tree with labelled internal nodes. Let $u_i$ be one of these nodes. Let $(u_i, v)$ be an edge connecting $u_i$ with another node $v$. We will create a new internal node $u_k$ with the same root set as $u_i$, meaning $VV(u_k) = VV(u_i)$. We will then remove the $(u_i, v)$ edge and connect $u_k$ to $u_i$ and $v$ with two edges $(u_i, u_k)$ and $(u_k, v)$. We will then create a new leaf node $u_l$ with $VV(u_l) = VV(u_i)$ and connect it to node $u_k$ with an edge $(u_k, u_l)$. Finally we will move the label from $u_i$ to $u_l$, effectively removing a labelled internal node and creating a labelled leaf. The construction is shown in Fig. 5.

The tree cost remains unchanged during these operations, since edge $(u_k, v)$ will have the same potential cost (for the same fit of $v$) as edge $(u_i, v)$ had, where the other new edges $(u_i, u_k)$ and $(u_k, u_l)$ will have potential costs of 0, since they connect nodes with the same single-fit root sets. We can repeat this process independently for every internal labelled node, until the only labelled nodes are leaves, while the MP-cost of the tree remains the same. \qed

Lemma 2. In leaf-labelled trees, a multifurcating $n$-species MP-tree cannot have a lower cost fit than an $n$-species cubic MP-tree.

Proof. We will prove this lemma by construction, once again without modifying the MP-tree cost. A multifurcating tree has two types of nodes that do not appear in a cubic tree, nodes of degree 2 and nodes of degrees $\geq 4$. We have seen how to remove unlabelled nodes of degree 2 in Theorem 1 without increasing the MP-tree cost. To remove tree nodes with degrees greater than 3 we will introduce a split operation that will create a new node, reduce the degree of an existing node by 1, and conserve the tree cost.

Consider a node $u_i$ with degree $d > 3$. Node $u_i$ will be adjacent to $d$ other nodes $v_1, v_2, \ldots, v_d$. We will create a new unlabelled node $u_k$ with the same root set as $u_i$, which we will connect to $u_i$. Then we will disconnect nodes $v_{d-1}$ and $v_d$ from $u_i$, and connect them to $u_k$. The modified node $u_i$ is now connected to nodes $v_1, v_2, \ldots, v_{d-2}, u_k$ and has degree $d - 1$, where node $u_k$ is adjacent to $u_i, v_{d-1}$ and $v_d$, and has degree 3. The degrees of all other nodes are unchanged.
The MP-tree cost remains the same, as new edge \((u_i, u_k)\) has a potential cost of 0 with an original tree best fit, and removed edges \((u_i, v_{d-1})\) and \((u_i, v_d)\) carry the same potential cost with added edges \((u_k, v_{d-1})\) and \((u_k, v_d)\) respectively. The split operation is shown in Fig. 6.

Repeating the split operation on all nodes with degrees \(\geq 4\) until their degrees are reduced to 3 will produce a cubic tree with the same MP-cost as the original multifurcating tree.

\[\square\]

**Lemma 3.** Unlabelled nodes with degrees \(d < 3\) can be removed from an \(n\)-species tree without increasing its MP-cost.

**Proof.** We have seen how to remove unlabelled nodes of degree 2 in theorem Theorem 1 without increasing the tree MP-cost. To remove an unlabelled leaf \(v\), we can notice that its incident edge can always have a cost of 0 for any given fit, since we can always set \(V V(v) = V V(u)\), where \(u\) is the single neighbor of \(v\). As such, removal of \(v\) and its incident edge does not increase the tree cost.

\[\square\]

**Lemma 4.** A most compact parsimonious \(n\)-species tree will have at most \(n - 2\) unlabelled nodes.

**Proof.** Based on Lemma 3, all leaves of a most compact MP-tree will be labelled. Thus, such a tree will have \(n\) leaves. Assume to the contrary of our stated lemma that a most compact \(n\)-species MP-tree has \(k \geq n - 1\) internal nodes, all of which have degrees \(\geq 3\), as per Lemma 3. Then the total number of nodes of the tree is \(n + k\). A tree with \(n + k\) nodes has \(n + k - 1\) edges. The sum of the node degrees then will be \(2n + 2k - 2\), since every edge contributes 2 to the total sum.

The sum of the degrees of the \(n\) leaves is \(n\), which means that the sum of degrees of the internal nodes \(S = n + 2k - 2\). Since every internal node has a degree of at least 3, the \(k\) internal nodes will have a sum of degrees \(S \geq 3k < n + 2k - 2 \geq 3k \leq k \leq n - 2\), which contradicts our assumption \(k \geq n - 1\).

\[\square\]

Now we can proceed with the proof of our main theorem:
Theorem 2. The most compact parsimonious \( n \)-species tree cannot have a lower cost fit than the most parsimonious \( n \)-species cubic tree.

Proof. Assume to the contrary that there exists a tree \( \tau_c \) on \( n \) species \( S \) that has a lower cost than the most parsimonious cubic tree \( \tau \) on \( S \). Using the construction in Lemma 1 we can move all labelled internal nodes to leaves without any increase to the MP-cost of \( \tau_c \). Based on Lemma 4 we could remove all unlabelled nodes with degrees \( \leq 2 \) without changing the MP-cost of \( \tau_c \) as well. Now \( \tau_c \) has only nodes with degree 1 or degree \( \geq 3 \). Using the construct of Lemma 2 we can convert \( \tau_c \) to a cubic tree, by successively splitting nodes of degree higher than 3, again without increasing the MP-cost of \( \tau_c \). The resulting tree is cubic, has all species in \( S \) associated with leaves, and a lower cost than \( \tau \), which is a contradiction. \( \square \)

Theorem 2 enables us to build the most compact MP-tree without enumerating all \( n \)-species trees, but only cubic trees with \( n \) species. It also supplies us with a systematic procedure to create the most compact parsimonious tree by reversing the process described in Theorem 2. Starting with the \( n \)-species cubic MP-trees, we can contract edges with 0-min-cost, effectively reversing the split operation. But which is the right order to contract edges, so that we can produce the most compact parsimonious tree? The relation \( RV \rightarrow V : (u, v) \in R \iff md(u, v) = 0 \) is not transitive, and edge contraction order can matter. Therefore we will consider all possible orders of edge contractions.

Our algorithm for contracting a cubic tree to the most compact tree on \( n \) species with the same cost is described in Algorithm 5.

| Algorithm 5 | Tree contraction algorithm |
|-------------|----------------------------|
| 1: loop For each 0-min-cost edge |
| 2: Contract edge |
| 3: Update root set of new node |
| 4: Update root sets of all unlabelled tree nodes and list of 0-min-cost edges |

The contraction of an edge \((u, v)\) creates a new vertex \( w \), whose root set is computed as follows: \( VV(w)_i = VV(u)_i \cap VV(v)_i \), \( 1 \leq i \leq m \).

The root sets of all remaining unlabelled nodes in the tree are updated recursively via a DFS traversal starting at node \( w \). For every node \( u \) whose initial root set was \( VV(pu) \), whose parent’s previous root set was \( VV(pp) \) and whose parent’s current root set is \( VV(cp) \), the new root set \( VV(u) \) is calculated as follows: \( VV(u)_i = (VV(pu)_i \setminus VV(pp)_i) \cup (VV(pu)_i \cap VV(cp)) \), \( 1 \leq i \leq m \).

The proof of correctness of Algorithm 5 follows from the reversal of the conversion of the most compact MP-tree to a cubic MP-tree. We are exhaustively enumerating all \( n \)-species cubic trees and, for the most parsimonious of them, we are considering all possible orders of edge contractions. Contracting edges reverses the node split operation that was utilized in Theorem 2.
Most Compact Parsimonious Trees

The initial cubic tree (before any contractions) has $n$ labelled and $n-2$ unlabelled nodes, therefore $2n-2$ nodes and $2n-3$ edges. The minimum number of nodes a compact tree can have is $n$, so the maximum number of consecutive contractions that can be performed is $n-2$. In the worst case our algorithm can iterate $(2n-3)(2n-4)\cdots n = \binom{2n-3}{n-2}$ times. Each iteration involves a DFS traversal that takes linear time as a function of the size of the tree. The worst time complexity of the algorithm is $O((2n-3)!)$ for a single tree topology (SPP), and $o((2n-3)!!)$ for finding the most compact parsimonious tree for $n$ species (LPP).

7 Conclusion

We have created an algorithm to generate the most compact parsimonious tree with $n$ species, by enumerating all cubic $n$-species trees, finding the most parsimonious ones, and optimally contracting them. Although contraction requires hyper-exponential time as a function of the number of species, the running time of the algorithm is superior to the enumeration of all multifurcating trees with $n$ species, even in the worst case. On average we expect the contraction algorithm to be very efficient, as the probability of contracting an tree edge decreases exponentially as a function of the number of characters examined. Furthermore, cubic tree enumeration has been refined in several existing phylogenetic software suites for many years [14,34], and a large number of heuristics, approximations, and parallel algorithms have been developed and used effectively to speed up enumeration [5,19,38,3,33,18,1,37], advancements which our algorithm can easily take advantage of to further improve its efficiency.

It is our hope that our algorithms will enable the creation of new software that handles multifurcation and extant species in internal nodes natively. Such tools are expected to further enhance studies in evolutionary virology, paleontology, and phylogenetic stemmatics.

Acknowledgments This work has been supported by NSF Grant CCF-1418874 and The College of New Jersey Mentored Undergraduate Summer Experience (MUSE) program.

References

1. Alon, N., Chor, B., Pardi, F., Rapoport, A.: Approximate maximum parsimony and ancestral maximum likelihood. IEEE/ACM Transactions on Computational Biology and Bioinformatics 7(1), 183–187 (2010)
2. Andrews, T.L., Macé, C.: Beyond the tree of texts: Building an empirical model of scribal variation through graph analysis of texts and stemmata. Literary and Linguistic Computing 28(4), 504–521 (2013)
3. Bader, D.A., Chandu, V.P., Yan, M.: ExactMP: An efficient parallel exact solver for phylogenetic tree reconstruction using maximum parsimony. In: Proceedings of the International Conference on Parallel Processing. pp. 65–73 (2006)
4. Bapst, D.W.: paleotree: an R package for paleontological and phylogenetic analyses of evolution. Methods in Ecology and Evolution 3(5), 803–807 (2012), http://dx.doi.org/10.1111/j.2041-210X.2012.00223.x
5. Bonet, M., Steel, M., Warnow, T., Yooseph, S.: Better methods for solving parsimony and compatibility. J Comput Biol 5(3), 391–407 (1998), http://www.ncbi.nlm.nih.gov/pubmed/9773340
6. Buendia, P., Narasimhan, G.: Serial evolutionary networks of within-patient HIV-1 sequences reveal patterns of evolution of X4 strains. BMC Systems Biology 3, 62 (2009), http://bmcsystbiol.biomedcentral.com/articles/10.1186/1752-0509-3-62
7. Bush, R., Bender, C., Subbarao, K., Cox, N., Fitch, W.: Predicting the evolution of human influenza A. Science 286(5446), 1921-1925 (1999), http://www.ncbi.nlm.nih.gov/pubmed/10583948
8. Carmel, A., Musa-Lempel, N., Tsur, D., Ziv-Ukelson, M.: The worst case complexity of maximum parsimony. In: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). vol. 8486 LNCS, pp. 79–88 (2014)
9. Carroll, S.A., Towner, J.S., Sealy, T.K., McMullan, L.K., Khristova, M.L., Burt, F.J., Swanepoel, R., Rollin, P.E., Nichol, S.T.: Molecular evolution of viruses of the family filoviridae based on 97 whole-genome sequences. Journal of Virology 87(5), 2608–2610 (2013), http://jvi.asm.org/content/87/5/2608.abstract
10. Day, W.H.E.: Computationally difficult parsimony problems in phylogenetic systems. Journal of Theoretical Biology 103(3), 429–438 (1983)
11. Day, W.H.E., Johnson, D.S., Sankoff, D.: The computational complexity of inferring rooted phylogenies by parsimony. Mathematical Biosciences 81(1), 33–42 (1986)
12. Dopazo, J., Dress, a., von Haezeler, a.: Split decomposition: a technique to analyze viral evolution. Proceedings of the National Academy of Sciences of the United States of America 90(21), 10320–10324 (1993)
13. Exoo, G.: A simple method for constructing small cubic graphs of girths 14, 15, and 16. The Electronic Journal of Combinatorics 3(1), 30 (1996), http://eudml.org/doc/119250
14. Felsenstein, J.: Phylip: phylogeny inference package (version 3.2). Cladistics 5, 164–166 (1989), http://evolution.genetics.washington.edu/phylip/faq.html#citation
15. Fitch, W.M.: Toward defining the course of evolution: Minimum change for a specific tree topology. Systematic Zoology 20(4), 406–416 (1971), http://www.jstor.org/stable/2412116
16. Flight, C.: How many stemmata? Manuscripta 34(2), 122–128 (1990), http://dx.doi.org/10.1484/J.MSS.3.1335
17. Foote, M.: On the probability of ancestors in the fossil record. Paleobiology 22, 141–151 (1996), http://journals.cambridge.org/article_S0094837300016146
18. Goeffon, A., Richer, J.M., Hao, J.K.: Progressive tree neighborhood applied to the maximum parsimony problem. IEEE/ACM Transactions on Computational Biology and Bioinformatics 5(1), 136–145 (2008)
19. Goloboff, P.A.: Optimization of polytomies: state set and parallel operations. Molecular phylogenetics and evolution 22(2), 269–275 (2002)
20. Graham, R.L., Foulds, L.R.: Unlikelihood that minimal phylogenies for a realistic biological study can be constructed in reasonable computational time. Mathematical Biosciences 60(2), 133–142 (1982)
21. Hartigan, J.A.: Minimum mutation fits to a given tree. Biometrics 29(1), 53–65 (1973), http://www.jstor.org/stable/2529676
22. Hillis, D., Bull, J., White, M., Badgett, M., Molineux, I.: Experimental phylogenetics: generation of a known phylogeny. Science 255(5044), 589–592 (1992), http://www.ncbi.nlm.nih.gov/pubmed/1736360

23. Maas, P.: Textual criticism. Clarendon Press (1958), https://books.google.com/books?id=1-xnAAAAAMAAJ

24. Mierzejewski, P.: A new graptolite, intermediate between the tuboidea and the camaridea. Acta Palaeontologica Polonica 46(3), 367–376 (2001), https://www.app.pan.pl/archive/published/app46/app46-367.pdf

25. Roelli, P., Bachmann, D.: Towards generating a stemma of complicated manuscript traditions: Petrus alfonsi’s dialogus. Revue d’histoire des textes 5, 307–321 (2010), http://dx.doi.org/10.5167/uzh-34542

26. Ronquist, F.: Fast fitch-parsimony algorithms for large data sets. Cladistics 14(4), 387–400 (1998), http://www.sciencedirect.com/science/article/pii/S0748303798900822

27. Roos, T., Zou, Y.: Analysis of textual variation by latent tree structures. In: Proceedings - IEEE International Conference on Data Mining, ICDM. pp. 567–576 (2011)

28. Saitou, N., Nei, M.: The neighbor-joining method: a new method for reconstructing phylogenetic trees. Molecular Biology and Evolution 4(4), 406–425 (1987), http://mbe.oxfordjournals.org/content/4/4/406.abstract

29. Salemans, B.: Building Stemmas with the Computer in a Cladistic, Neo-Lachmannian, Way: The Case of Fourteen Text Versions of Lancelot Van Denemerken. University Press (2000), https://books.google.com/books?id=Gr6THAAACAAJ

30. Sankoff, D.: Minimal mutation trees of sequences. SIAM Journal on Applied Mathematics 28(1), 35–42 (1975), http://dx.doi.org/10.1137/0128004

31. Sloane, N.: The On-Line Encyclopedia of Integer Sequences (2010), http://oeis.org

32. Spencer, M., Davidson, E.A., Barbrook, A.C., Howe, C.J.: Phylogenetics of artificial manuscripts. Journal of Theoretical Biology 227(4), 503–511 (2004)

33. Sridhar, S., Lam, F., Blelloch, G.E., Ravi, R., Schwartz, R.: Mixed integer linear programming for maximum-parsimony phylogeny inference. In: IEEE/ACM Transactions on Computational Biology and Bioinformatics. vol. 5, pp. 323–331 (2008)

34. Swofford, D.L.: Phylogenetic Analysis Using Parsimony. Options 42, 294–307 (2003), http://www.springerlink.com/index/10.1007/BF02198556

35. Swofford, D.L., Sullivan, J.: Phylogeny inference based on parsimony and other methods using PAUP. In: Lemey, P., Salemi, M., Vandamme, A.M. (eds.) The Phylogenetic Handbook, pp. 267–312. Cambridge University Press, second edn. (2009), http://dx.doi.org/10.1017/CBO9780511819049.010, cambridge Books Online

36. Urbanek, A.: Oligophyly and evolutionary parallelism: A case study of silurian graptolites. Acta Palaeontologica Polonica 43(4), 549–572 (1998)

37. White, W.T.J., Hollard, B.R.: Faster exact maximum parsimony search with XMP. Bioinformatics 27(10), 1359–1367 (2011)

38. Yan, M., Bader, D.A.: D.a.: Fast character optimization in parsimony phylogeny reconstruction. Tech. rep., Georgia Institute of Technology (2003)

39. Zhu, T., Korber, B., Nahmias, A., Hooper, E., Sharp, P., Ho, D.: An african hiv-1 sequence from 1959 and implications for the origin of the epidemic. Nature 391(6667), 594–597 (1998), http://www.nature.com/nature/journal/v391/n6667/full/391594a0.html

All links were last followed on February 10, 2016.