Journal of Graph Algorithms and Applications
http://jgaa.info/ vol. 20, no. 2, pp. 247–268 (2016)
DOI: 10.7155/jgaa.00392

A navigation system for tree space

Sarah Mark[1] *Jeanette C. McLeod*[1,2] *Mike Steel*[1]

[1]Department of Mathematics and Statistics, University of Canterbury, Christchurch, New Zealand
[2]Te Pūnaha Matatini, New Zealand

Abstract

The reconstruction of evolutionary trees from data sets on overlapping sets of species is a central problem in phylogenetics. Provided that the tree reconstructed for each subset of species is rooted and that these trees fit together consistently, the space of all parent trees that 'display' these trees was recently shown to satisfy the following strong property: there exists a path from any one parent tree to any other parent tree by a sequence of local rearrangements (nearest neighbour interchanges) so that each intermediate tree also lies in this same tree space. However, the proof of this result uses a non-constructive argument. In this paper we describe a specific, polynomial-time procedure for navigating from any given parent tree to another while remaining in this tree space. The results are of particular relevance to the recent study of 'phylogenetic terraces'.

Submitted:	Reviewed:	Revised:	Reviewed:	Accepted:
January 2016	February 2016	March 2016	March 2016	March 2016

Final:	Published:
March 2016	April 2016

Article type:	Communicated by:
Regular Paper	T. Warnow

We thank the Biomathematics Research Centre for partial funding of the work presented in this paper.

E-mail addresses: mark.sarahjayne@gmail.com (Sarah Mark) jeanette.mcleod@canterbury.ac.nz (Jeanette C. McLeod) mike.steel@canterbury.ac.nz (Mike Steel)

1 Introduction

A central goal in systematic biology is to reconstruct and analyze a (phylogenetic) tree to describe the evolutionary relationships among present-day species, based on a comparison of their genetic data [3]. This activity has accelerated greatly in recent years due to the rapid advances in new genomic sequencing technology. While biologists in the 1970s might have reconstructed a tree for a dozen species using a single gene, today, phylogenetic trees are routinely constructed for hundreds or thousands of species, often based on large numbers (hundreds or thousands) of genes. These trees reveal how species today trace back to a common ancestor by displaying the branching pattern and timing of separation events. These trees, in turn, provide insights into how particular evolutionary innovations arose that are present in the group of species under study (e.g. multicellularity, photosynthesis, wings, large brains, etc). Phylogenetic trees can also shed light on the amount of biodiversity captured by different subsets of species and how much of this biodiversity may be at risk from extinction in the near future (a recent example is the analysis in [4] of the estimated tree for all $\sim$10,000 species of birds).

Tree reconstruction methods often attempt to combine the evolutionary signal across many different genes. One of the problems with such an approach is that each gene may be present in only a subset of the species. This may be because the gene simply does not exist in some species or because the gene, though present, is yet to be sequenced for those species. Moreover, the set of species that lack a given gene typically varies from gene to gene.

Patchy taxon coverage has a direct combinatorial consequence for tree reconstruction methods, which often seek to optimize (e.g. minimize) some objective function based on how well the data 'fit' each tree. The result can be large collections of equally-optimal trees (i.e. a flat landscape of trees), that form a (phylogenetic) 'terrace' [7], which we define more precisely below. To describe this connection with terraces, we first describe some properties of commonly used scoring functions. Let X be a set of species and let X_G be the subset of X of species for which gene G is present, and suppose that T is a fully-resolved (binary) tree with leaf set X. Consider a scoring function s that assigns a positive real value for each such pair (G,T). In biological applications, s will generally satisfy the following equation:

$$s(G,T) = s(G,T|X_G), \tag{1}$$

where $T|X_G$ refers to the phylogenetic tree with leaf set X_G obtained from T by deleting all species in X for which G is not present. This condition essentially says that species for which the gene is not present should not affect how well the data for the gene 'fits' the tree under consideration.

Now suppose the data consists of a sequence of genes $\mathcal{G} = (G_1, G_2, \ldots, G_k)$. Given the score $s(G_i,T)$ for each i, how might we combine them to obtain a score $s(\mathcal{G},T)$ for how well this collection of genes 'fits' T? A natural option is simply to form a linear sum and let

$$s(\mathcal{G}, T) = \sum_{i=1}^{k} s(G_i, T). \tag{2}$$

We say that any such scoring scheme is *linear*. Given the data $\mathcal{G}$, we seek to find a tree that minimizes $s(\mathcal{G}, T)$. While linearity may seem a strong condition to impose, it turns out that some standard phylogenetic methods select a tree that minimizes a linear scoring scheme. One such method is minimum evolution (also called 'maximum parsimony') and another is (partitioned) maximum likelihood (in which parameters such as the branch lengths of a tree are optimized independently from gene to gene), for which $s(G_i, T)$ is equal to minus the log-likelihood of T for G_i; for further details see [7, 6]. Both of these methods also satisfy Eqn.(1), as do several others that fail linearity (e.g. maximum likelihood with branch length parameters linked across genes).

Now suppose that T^* is a fully-resolved tree that has some particular score (e.g. the optimal score) for $\mathcal{G}$ under a scoring function s that satisfies Eqn. (1) and is linear (Eqn. (2)), and let $X_i = X_{G_i}$, for $i = 1, \ldots, k$. Suppose that T is any other tree for which $T|X_i = T^*|X_i$ for all i. The tree T then has the same score as T^* for $\mathcal{G}$. To see this, simply observe that

$$s(\mathcal{G}, T) = \sum_{i=1}^{k} s(G_i, T) = \sum_{i=1}^{k} s(G_i, T|X_i) = \sum_{i=1}^{k} s(G_i, T^*|X_i) =$$

$$\sum_{i=1}^{k} s(G_i, T^*) = s(\mathcal{G}, T^*).$$

The set of phylogenetic X–trees T for which $T|X_i = T^*|X_i$ for all i is refereed to as the *terrace* containing T^*, and all trees on this terrace have the same s–score (when Eqns. (1) and (2) hold). In real applications, a terrace can be very large, for example, 61 million equally-optimal (maximum likelihood) trees for a data set consisting of 298 species of grasses on three genes [7]. The existence of large flat landscapes of trees can make the search for optimal trees by hill-climbing approaches more problematic, and ways in which search times on terraces can be improved are still being sought [2].

In this paper, we explore a further combinatorial consequence of patchy taxon coverage: namely, for any terrace of trees (which thus have the same s-score), it is possible to move from any one tree on the terrace to any other tree on the terrace by making a series of local elementary re-arrangments (called 'nearest neighbour interchange' (NNI) operations, defined below), while always remaining on that terrace (i.e. not altering the s-score). This result follows from a theorem, first stated and proved in the PhD thesis of Magnus Bordewich [1], based on an inductive argument. The motivation for the current paper is to provide an explicit algorithm for constructing a sequence of NNI operations to move from any one tree T on the terrace to any other tree T' on the terrace. In our approach, the details of the scoring function s play no real role since a terrace is the set of binary trees displaying the set of rooted trees $\{T|X_1, \ldots, T|X_k\}$, where T is some tree on that terrace. Thus we deal simply with sets of rooted trees on overlapping leaf sets as our input.

It is important to note that, although the trees comprising a terrace have the same s–score (when Eqns. (1) and (2) hold), there may be other trees with the same s–score that are not on this terrace. We can see this even in the simple case where $X_i = X$ for all i (i.e. we have complete taxon coverage). Then for any fully-resolved tree T, the terrace containing T is just T itself, yet for certain data there may be many maximum parsimony trees. Moreover, in this setting of complete taxon coverage it has been known since the early 1990s that for particular data there may be two or more optimal trees with optimal parsimony scores that cannot be connected by a sequence of NNI operations passing only through optimal trees (this leads to 'islands of trees' as studied in [5]). The result described in the previous paragraph concerns connectivity under NNI within a single terrace (not between terraces) and so its relevance is particular to the setting of partial taxon coverage. In general, the set of trees having a given s–score will be a union of one or more terraces.

The structure of our paper is as follows. We first define some terms and operations on trees in Section 2, and in Section 2.3, we summarize a result from [1]. In Section 3, we state the main result of this paper, then present and prove some preliminary results before we provide a proof of the main result at the end of this section. This is followed, in Section 4, by an algorithm and an analysis of its complexity. We end with some brief concluding comments in Section 5.

2 Preliminaries

Our terminology follows that of Semple and Steel [8]. A *rooted phylogenetic tree* T is a semi-labeled rooted tree in which the leaves of T are labeled and the root has outdegree at least two. Let $RP(X)$ be the set of all such trees with leaf label set X. A *rooted binary phylogenetic tree* is a rooted phylogenetic tree for which all interior vertices have outdegree two. Let $RB(X)$ denote the set of all such trees with leaf label set X. Note that $RB(X) \subseteq RP(X)$. A tree $T \in RP(X)$ is a *star* if it has only one interior vertex (which is the root).

Consider $\mathcal{P} = \{T_1, \ldots, T_k\}$, where $T_i \in RP(X_i)$ for each $1 \leq i \leq k$. Then

$$\mathcal{L}(\mathcal{P}) = \bigcup_{i=1}^{k} X_i$$

denotes the leaf set of $\mathcal{P}$. For a single tree T_i, for ease of notation, we write $\mathcal{L}(\{T_i\}) = \mathcal{L}(T_i) = X_i$.

Consider $T \in RP(X)$. A *rooted phylogenetic subtree* t_v of T is a subtree of T whose vertex set consists of a vertex v of T and all descendants of v in T. The vertex v is the root of the subtree t_v. If v is not the root of T, then t_v is a *proper rooted phylogenetic subtree* of T. If v is a child of the root of T, then

t_v is a *maximal proper rooted phylogenetic subtree* of T. Throughout the rest of this paper, 'subtree' will refer to a rooted phylogenetic subtree unless otherwise specified.

A *cluster* of T is a subset of X consisting of all leaves that are descendants of a given vertex v in T, that is, the set $\{x \in X : x \text{ is a descendant of } v\}$. The collection of all clusters of T, denoted $\mathcal{C}(T)$, determines T. A *maximal proper cluster* of T is the leaf set of a maximal proper subtree of T.

We define two trees $T, T' \in RB(X)$ to be *equivalent* if there is a map $\phi : V(T) \to V(T')$ such that $\phi(l) = l$ for all $l \in X$ and a map $\psi : E(T) \to E(T')$ such that adjacency is preserved. If T has subtree t and T' has subtree t', where t is equivalent to t', then, to aid exposition, we say that T' has subtree t. The *distance between two vertices* u and v in T, $\text{dist}_T(u, v)$, is the length of the shortest path between u and v in T. The *distance between two arcs* $e = (u_1, u_2)$ and $e' = (v_1, v_2)$ in T is $\min\{\text{dist}_T(u_i, v_j) : i, j \in \{1, 2\}\}$.

2.1 Operations on trees

Consider a tree $T = (V, E) \in RP(X)$, and an arc $e = (v, u)$ of T. The graph $T \setminus e = (V, E \setminus \{e\})$ is said to be obtained from T by *deleting* e. The graph obtained from T by deleting e and replacing its endpoints u and v with a new vertex (so that all arcs incident to u or v are now incident to this new vertex) is denoted T/e and is said to be obtained from T by *contracting* e. Consider vertex x of T and suppose that all but one of its outgoing arcs has been deleted, giving a tree in which x has outdegree one. The method used to suppress x depends on whether or not x is the root of T. If x is not the root of T, let $e_1 = (w, x)$ and $e_2 = (x, y)$ be arcs of T. The tree T' obtained from T by deleting vertex x and arcs e_1 and e_2 from T and inserting arc (w, y) is said to be obtained from T by *suppressing* x. Note that a tree equivalent to T' can also be obtained as T/e_1 or T/e_2. If x is the root of T (and x has outdegree one, as before), then x is suppressed by deleting x and its incident arc, making the child of x the root of the resulting tree.

Let v be a vertex of T and let $e_1, \ldots, e_k$ be the arcs of T incident to v. The graph $T \setminus v = (V \setminus \{v\}, E \setminus \{e_1, \ldots, e_k\})$ is said to be obtained from T by *deleting* v. Let $e = (v, u)$ be an arc of T and let t_u be the subtree of T with root u. We define $T \setminus t_u$ to be the tree obtained from T by deleting t_u and the arc e.

Suppose $T \in RB(X)$ and let $e = (v, u)$ be an arc of T. The tree T' given by introducing a new vertex w (so that $V(T') = V(T) \cup \{w\}$), deleting arc e, and inserting arcs (v, w) and (w, u) into T is said to be obtained from T by *subdividing e with w*.

The following two operations allow us to "prune" a subtree and "regraft" it elsewhere on the tree. The second operation is simply a special case of the first operation.

rSPR (rooted subtree prune and regraft) operation: Let $T \in RB(X)$ and let $e = (v, u)$ be an arc of T. We say that $T' \in RB(X)$ is an *rSPR-neighbour* of T

if T' can be obtained from T by the following procedure. Let t_u be the subtree of T rooted at u. Delete arc e, *pruning* the subtree t_u. To *regraft* t_u, either:

(i) Choose an arc f of $T \setminus t_u$ and subdivide f with a vertex w, then insert the arc (w, u), *regrafting* the subtree t_u, or
(ii) Introduce a vertex r', insert the arc (r', r_T) where r_T is the root of T, and then insert the arc (r', u), *regrafting* the subtree t_u. Note that r' is the root of the resulting tree.

Lastly, suppress v. We have now obtained a tree T' that is an rSPR-neighbour of T and we write $T \overset{\text{rSPR}}{\sim} T'$. Note that $T \overset{\text{rSPR}}{\sim} T$. Also, if $T \overset{\text{rSPR}}{\sim} T'$, then $T' \overset{\text{rSPR}}{\sim} T$. We say that this rSPR operation is performed *with respect to* t_u.

NNI (nearest neighbour interchange) operation on a rooted tree:
Let $T \in RB(X)$ and let $e = (v, u)$ be an arc of T. We say that $T' \in RB(X)$ is an *NNI-neighbour* of T if T' can be obtained from T by the following procedure. Let t_u be the subtree of T rooted at u. Let w be a vertex of T adjacent to v, where $w \neq u$. Delete arc e. Then:
If w is not the root of $T \setminus t_u$,

(i) Choose an arc f incident to w, and subdivide f with a vertex x.

If w is the root of $T \setminus t_u$, either do (i) or

(ii) Introduce a vertex x and insert the arc (x, w). Note that x is the root of the resulting tree.

Now insert the arc (x, u) into T, and, lastly, suppress v. We have now obtained a tree T' which is an NNI-neighbour of T and we write $T \overset{\text{NNI}}{\sim} T'$. Note that $T \overset{\text{NNI}}{\sim} T$. Also, if $T \overset{\text{NNI}}{\sim} T'$, then $T' \overset{\text{NNI}}{\sim} T$. We say that the NNI operation is performed *with respect to* t_u.

We define a *sequence of NNI-related trees* to be a sequence of trees, say $(T_1, \ldots, T_n)$, for which $T_i \overset{\text{NNI}}{\sim} T_{i+1}$ for all $1 \leq i < n$; that is, each tree in the sequence can be obtained from the previous tree by a single NNI operation (not necessarily with respect to the subtree t_u). We refer to these NNI operations as a *sequence of NNI operations*. A sequence of NNI operations from T_1 to T_n is called a *minimum sequence of NNI operations* if it is the shortest sequence of NNI operations that starts with the tree T_1 and produces the tree T_n. If each of the NNI operations in a sequence is performed with respect to the subtree t_u, then we refer to this as a *sequence of NNI operations with respect to* t_u. We define an analogous set of terms for rSPR operations.

When we perform an operation on a tree T, some vertices may be deleted or inserted to produce the tree T'. All other vertices retain the same labels in both T and T', although we note that arcs may have been deleted or inserted and so the connections between these vertices may be different. Figure 1 shows an example of this for an NNI operation. In this example, the subtree t_u rooted at u is pruned and regrafted. Vertex v is suppressed and vertex x is inserted, so $V(T') = (V(T) \setminus \{v\}) \cup \{x\}$.

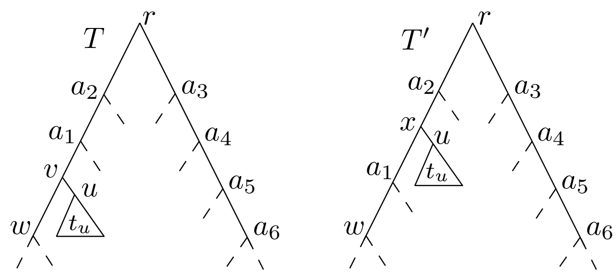

Figure 1: An example of the labeling of vertices before (T) and after (T') an NNI operation with respect to t_u.

2.2 Triples, refinement, display, and compatibility

Let $T \in RP(X)$ and let $X' \subseteq X$. Then $T|X' \in RP(X')$, called the *restriction of* T *to* X', is the tree for which

$$\mathcal{C}(T|X') = \{C \cap X' : C \in \mathcal{C}(T) \text{ and } C \cap X' \neq \emptyset\}.$$

We can obtain $T|X'$ from T by deleting all maximal subtrees containing only leaves that are not in X' and then suppressing all vertices with outdegree one.

Let $T' \in RP(X)$. We say that T *refines* T' (or *is a refinement of* T') if $\mathcal{C}(T') \subseteq \mathcal{C}(T)$.

Let $X'' \subseteq X$, and let $T'' \in RP(X'')$. We say that T *displays* T'' if $T|X''$ is a refinement of T''.

A set $\mathcal{P}$ of rooted phylogenetic trees is *compatible* if there exists a tree $T \in RP(X)$ such that T displays each tree in $\mathcal{P}$. We then say that T *displays* $\mathcal{P}$. Let $\langle \mathcal{P} \rangle$ (respectively $\langle \mathcal{P} \rangle_B$) denote the set of all rooted phylogenetic trees (respectively rooted binary phylogenetic trees) that display $\mathcal{P}$. Note that $\langle \mathcal{P} \rangle_B \subseteq \langle \mathcal{P} \rangle$.

A *rooted triple* is a tree in $RB(X)$ where $|X| = 3$. A rooted triple with $X = \{a, b, c\}$ is denoted $ab|c$ if the path from a to b does not intersect the path from c to the root of the tree. Let $r(T)$ denote the set of all rooted triples displayed by $T \in RB(X)$. Figure 2 shows an example of a set $\mathcal{R}$ of rooted triples and two trees $T, T' \in \langle \mathcal{R} \rangle_B$.

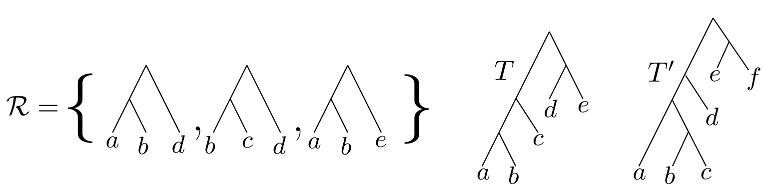

Figure 2: An example of a set $\mathcal{R}$ of rooted triples and two rooted binary phylogenetic trees T and T' that both display $\mathcal{R}$.

Note that we only need to consider a set of rooted triples $\mathcal{R}$ rather than a more general set of rooted phylogenetic trees $\mathcal{P}$, since the latter can be converted into the former so that the trees displaying $\mathcal{R}$ are exactly those which display $\mathcal{P}$. Let $\mathcal{P}'$ be a set of rooted phylogenetic trees such that each tree in $\mathcal{P}'$ has at least one internal arc (there are no stars). Then $\langle \mathcal{P}' \rangle_B = \langle \mathcal{R}_{\mathcal{P}'} \rangle_B$, where $\mathcal{R}_{\mathcal{P}'}$ is the set of rooted triples such that, for each rooted triple $ab|c \in \mathcal{R}_{\mathcal{P}'}$, there is some tree in $\mathcal{P}'$ which displays $ab|c$. The case in which $\mathcal{P}$ contains at least one tree that is a star will be dealt with later.

2.3 The Bordewich result

For any two trees $T, T' \in RB(X)$, there is a sequence of trees $(T_1, T_2, \dots, T_n)$ such that $T = T_1$, $T' = T_n$, and $T_i \overset{\text{NNI}}{\sim} T_{i+1}$ for all i $(1 \leq i < n)$. In this paper, we consider two trees, T and T', that display a set of rooted triples $\mathcal{R}$, and find a sequence of trees satisfying the aforementioned conditions, along with the additional condition that each tree in the sequence displays $\mathcal{R}$.

The following result was stated and proved in the PhD Thesis of Bordewich [1].

Theorem 1 *Let $\mathcal{R}$ be a set of rooted triples. Suppose that $T, T' \in \langle \mathcal{R} \rangle_B$ and $\mathcal{L}(T) = \mathcal{L}(T')$. Then there is a sequence of trees $(T_1, T_2, \dots, T_n)$ such that:*
1. *$T_1 = T$ and $T_n = T'$,*
2. *$T_i \overset{\text{NNI}}{\sim} T_{i+1}$ for $1 \leq i < n$, and*
3. *$T_i \in \langle \mathcal{R} \rangle_B$ for $1 \leq i \leq n$ (i.e. each T_i displays $\mathcal{R}$).*

3 Main result

We first note that Theorem 1 is equivalent to the following:

Theorem 2 *Let $T, T' \in \langle \mathcal{R} \rangle_B$, where $\mathcal{R} = r(T) \cap r(T')$ and $\mathcal{L}(T) = \mathcal{L}(T')$. Then there is a sequence of trees $(T_1, T_2, \dots, T_n)$ such that Properties (1)–(3) of Theorem 1 hold.*

To see why these two theorems are equivalent, first assume that Theorem 1 holds. Let $\mathcal{R}'$ be the set of rooted triples in Theorem 1 (for clarity of notation). If we let $\mathcal{R}' = r(T) \cap r(T')$ in Theorem 1, we have Theorem 2. Now assume that Theorem 2 holds. Once again let $\mathcal{R}'$ be the set of rooted triples in Theorem 1 (for clarity of notation). Since $\mathcal{R}' \subseteq r(T) \cap r(T')$ and each T_i displays $r(T) \cap r(T')$, then T, T', and each T_i will also display $\mathcal{R}'$, so Theorem 1 holds.

Figure 3 shows an example to illustrate Theorem 2. In this example, $\mathcal{R} = r(T) \cap r(T') = \{ac|d, ac|e, ac|f, de|a, de|c, ef|d\}$. It is easy to check that $T_1, \dots, T_4$ all display $\mathcal{R}$, and each tree can be obtained from the previous tree by a single NNI operation.

We are only considering rooted trees in this paper because there is no result directly analogous to Theorem 1 for unrooted trees and quartet trees (the unrooted analogue of rooted triples). To see this, consider the following counterexample. The two unrooted trees in Figure 4 both display the quartet trees

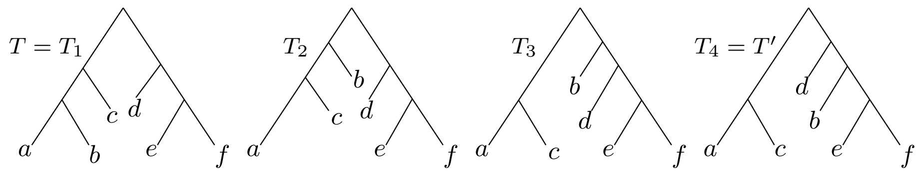

Figure 3: An example to illustrate Theorem 2 with $\mathcal{R} = \{ac|d, ac|e, ac|f, de|a, de|c, ef|d\}$.

12|45, 34|16, and 56|23. However, it is straightforward to check that the two trees in Figure 4 are the only two trees which display these quartet trees and that they are not one (unrooted) NNI operation apart. See [8] for further information and definitions.

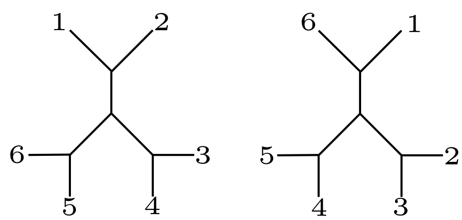

Figure 4: The two unrooted trees that display the quartets 12|45, 34|16, and 56|23.

The proof of Theorem 1 in [1] is based on an inductive argument. We present an alternative proof that provides an explicit procedure for obtaining the sequence of trees $(T_1, \ldots, T_n)$.

The main result of this section is the following theorem.

Theorem 3 *Given $T, T' \in \langle \mathcal{R} \rangle_B$, where $\mathcal{R} = r(T) \cap r(T')$ and $\mathcal{L}(T) = \mathcal{L}(T')$, one can construct (in polynomial time) a sequence of trees $(T_1, T_2, \ldots, T_n)$ such that:*
1. *$T_1 = T$ and $T_n = T'$,*
2. *$T_i \overset{\text{NNI}}{\sim} T_{i+1}$ for $1 \leq i < n$, and*
3. *$T_i \in \langle \mathcal{R} \rangle_B$ for $1 \leq i \leq n$ (i.e. each T_i displays $\mathcal{R}$).*

This takes into consideration the case in which $\mathcal{L}(\mathcal{R}) \subset \mathcal{L}(T) = \mathcal{L}(T')$, that is, there are leaves in T and T' that are not in any rooted triple in $\mathcal{R}$. We call these leaves (the leaves in $\mathcal{L}(T) \setminus \mathcal{L}(\mathcal{R})$) *free leaves.*

Let $\mathcal{P}''$ be a set of rooted phylogenetic trees such that at least one of these trees is a star. Let $\mathcal{P}_S$ be the subset of $\mathcal{P}$ consisting of all trees in $\mathcal{P}$ which are stars and let $\mathcal{P}' = \mathcal{P}'' \setminus \mathcal{P}_S$. All leaves which are in $\mathcal{L}(\mathcal{P}_S) \setminus \mathcal{L}(\mathcal{P}')$ (i.e. all leaves that are in a tree in $\mathcal{P}_S$ and are not in any tree in $\mathcal{P}'$) are free leaves.

So $\langle P'' \rangle_B = \{T \in \langle \mathcal{R}_{P'} \rangle_B : \mathcal{L}(P'') \subseteq \mathcal{L}(T)\}$ where $\langle \mathcal{R}_{P'} \rangle_B$ is defined as in Section 2.2.

Before we prove Theorem 3, we need some preliminary results.

Lemma 1 *Let $\mathcal{R}$ be a set of rooted triples and let $L = \mathcal{L}(\mathcal{R})$. Suppose that $T, T' \in \langle \mathcal{R} \rangle_B$ and $\mathcal{L}(T) = \mathcal{L}(T')$. For $T^* \in \{T, T'\}$, it is possible to construct, in polynomial time, a sequence of NNI-related trees from T^* to a tree $\widetilde{T^*}$ with subtree $T^*|L$ such that each tree in the sequence displays $\mathcal{R}$ and $\widetilde{T} \setminus (T|L)$ is equivalent to $\widetilde{T'} \setminus (T'|L)$.*

Informally, this means that we can disregard the free leaves, transform $T|L$ into $T'|L$, and then reinstate the free leaves in T'. As these free leaves do not appear in any of the rooted triples in $\mathcal{R}$, this process will not affect whether a particular tree displays $\mathcal{R}$.

Proof: [Proof of Lemma 1] First note that T and T' have the same leaf set, and let $\mathcal{L}(T) = \mathcal{L}(T') = \{x_1, \ldots, x_m\}$ such that $x_1, \ldots, x_n$ (for some $n \leq m$) are the free leaves. The following steps describe NNI operations which give a sequence of NNI-related trees from T to a tree $\widetilde{T}$ that has subtree $T|L$. An example of this can be seen in Figure 5.

(a) Consider $\mathcal{L}(T) \setminus L = \{x_1, \ldots, x_n\}$, the free leaves of T. Let $U_0 = T$ and let $S = (U_0)$.

(b) For $j = 1, \ldots, n$:
   - (i) If $j \neq 1$, let v_{j-1} be the root of the subtree $T|(\mathcal{L}(T) \setminus \{x_1, x_2, \ldots, x_{j-1}\})$ of U_{j-1}. Otherwise (if $j = 1$), let v_{j-1} be the root of U_0.
   - (ii) Consider U_{j-1}. If x_j is a child of v_{j-1}, let $U_j = U_{j-1}$. Otherwise,
     - (1) Perform a minimum sequence of NNI operations with respect to x_j that results in a tree in which x_j is the grandchild of v_{j-1}. Append the sequence of trees (each of which is the result of one NNI operation in the sequence) to S.
     - (2) If $j \neq 1$, perform the following NNI operation: prune x_j, subdivide the arc between v_{j-1} and its parent with a vertex w, insert arc (w, x_j), and suppress the vertex with outdegree one.
       Otherwise (if $j = 1$), perform the following NNI operation: prune x_1, introduce a vertex r', insert arc (r', v_0) and arc (r', x_1), and suppress the vertex with outdegree one.
     - (3) Call the resulting tree U_j and append U_j to the sequence S.

(c) Let $\widetilde{T} = U_n$, the last tree of the sequence S. The tree $\widetilde{T}$ has subtree $T|L$, as required.

The trees in S will only differ on rooted triples that contain some x_j (for $1 \leq j \leq n$), but by our assumption x_j is not in any rooted triple. Therefore S is a sequence of NNI-related trees from T to $\widetilde{T}$ for which each tree in the sequence displays $\mathcal{R}$.

Recall that $\mathcal{L}(T) = \mathcal{L}(T') = \{x_1, \ldots, x_m\}$, where $x_1, \ldots, x_n$ (for some $n \leq m$) are the free leaves. Repeating steps (a) through (c) for T', we obtain a sequence of trees S', where the first tree in the sequence is T' and the last

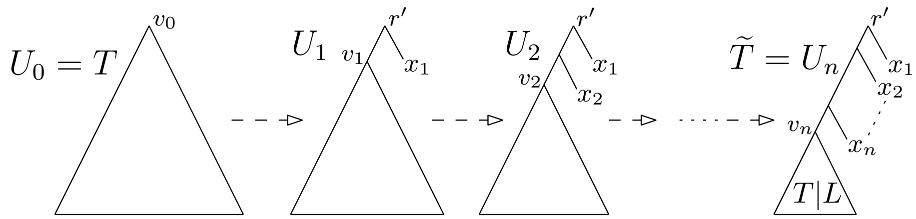

Figure 5: An example of the process in the proof of Lemma 1, where $x_1, \ldots, x_n$ are the free leaves.

tree is $\widetilde{T'}$ which has subtree $T'|L$. Now S' is a sequence of NNI-related trees from T' to $\widetilde{T'}$ for which each tree in the sequence displays $\mathcal{R}$. Since step (b) is acting on the leaf sets in T and T' in the same order, $\widetilde{T} \setminus (T|L)$ is equivalent to $\widetilde{T'} \setminus (T'|L)$, as required. $\square$

This last result simplifies our analysis so that we can always consider $T|L$ instead of T, where L is the set of leaves that are not free leaves. We have a sequence of trees from T to $\widetilde{T}$ and a sequence of trees from T' to $\widetilde{T'}$ (which can be reversed to give a sequence of trees from $\widetilde{T'}$ to T' as NNI operations are invertible). We now need a sequence of trees from $\widetilde{T}$ to $\widetilde{T'}$. We find a sequence of NNI operations which transforms $T|L$ into $T'|L$ and apply these NNI operations to $\widetilde{T}$, transforming the subtree $T|L$ into the subtree $T'|L$ and giving the tree $\widetilde{T'}$.

The following result further simplifies our analysis.

Lemma 2 *Suppose that $\mathcal{R}$ is a set of rooted triples and $T, T' \in \langle\mathcal{R}\rangle_B$, with $\mathcal{L}(T) = \mathcal{L}(T') = \mathcal{L}(\mathcal{R})$. Suppose that T' is obtained from T by one rooted subtree prune-and-regraft operation. Then there is a sequence of trees $(T_0, T_1, \ldots, T_n)$ such that:*

1. *$T_0 = T$ and $T_n = T'$,*
2. *$T_i \overset{\text{NNI}}{\sim} T_{i+1}$ for $0 \leq i < n$, and*
3. *$T_i \in \langle\mathcal{R}\rangle_B$ for $0 \leq i \leq n$ (i.e. each T_i displays $\mathcal{R}$).*

The proof of Lemma 2 is given in the Appendix.

Lemma 2 allows us to convert a sequence of rSPR operations into an equivalent sequence of NNI operations.

Lemma 3 *Let $\mathcal{R}$ be a set of rooted triples and suppose that $T \in \langle\mathcal{R}\rangle_B$ and $ab|c \in \mathcal{R}$. Then $a, b \in C$ for some maximal proper cluster C of T.*

Proof: Let r be the root of T and let C and $\bar{C}$ be the maximal proper clusters of T. For $ab|c \in \mathcal{R}$ suppose, without loss of generality, that $a \in C$ and $b \in \bar{C}$. The most recent common ancestor of a and b is then r, so the path from a to b will contain r and so this path will intersect the path from c to r. Therefore $ab|c \notin \mathcal{R}$, a contradiction. Hence if $ab|c \in \mathcal{R}$, then either $a, b \in C$ or $a, b \in \bar{C}$. $\square$

Lemma 4 *Let $\mathcal{R}$ be a set of rooted triples and let $T \in \langle\mathcal{R}\rangle_B$, where $\mathcal{L}(T) = \mathcal{L}(\mathcal{R})$. Let C and $\bar{C}$ be non-empty subsets of $\mathcal{L}(T)$ for which $C \cup \bar{C} = \mathcal{L}(T)$, $C \cap \bar{C} = \emptyset$, and C and $\bar{C}$ are maximal proper clusters of at least one tree in $\langle\mathcal{R}\rangle_B$. Consider $T|C$ and $T|\bar{C}$, with roots r_C and $r_{\bar{C}}$ respectively. Let $\hat{T} \in \langle\mathcal{R}\rangle_B$ be the tree rooted at $\hat{r}$ composed of exactly the subtrees $T|C$ and $T|\bar{C}$ and the arcs $(\hat{r}, r_C)$ and $(\hat{r}, r_{\bar{C}})$. Then there is a sequence of trees $(T_1, T_2, \ldots, T_n)$ such that:*

1. *$T_1 = T$ and $T_n = \hat{T}$,*
2. *$T_i|C = T|C$ and $T_i|\bar{C} = T|\bar{C}$ for $1 \leq i \leq n$,*
3. *$T_i \overset{\text{rSPR}}{\sim} T_{i+1}$ for $1 \leq i < n$, and*
4. *$T_i \in \langle\mathcal{R}\rangle_B$ for $1 \leq i \leq n$.*

Proof: We first show how to obtain, from T, a tree T' with subtree $T'|C = T|C$. For any i $(1 \leq i \leq n)$, the tree T_i contains one or more maximal subtrees whose leaf sets are subsets of C. If T_i contains only one such subtree, then that subtree must be $T|C$ and so $T' = T_i$. Now consider the case in which T_i contains two or more such subtrees. Let t_v be a minimal subtree of T_i containing exactly two maximal subtrees, t_x and t_y, whose leaf sets are subsets of C. (Note that t_v may contain leaves in $\bar{C}$.) We assume $i \leq n-2$ and apply the following two rSPR operations, starting at T_i, to produce a tree in which there is a subtree with leaf set $\mathcal{L}(t_x) \cup \mathcal{L}(t_y)$.

(a) Consider v, the most recent common ancestor of x and y. If v is not the root of T_i, let v' be the parent of v, and subdivide the arc (v', v) with a vertex u. Otherwise (if v is the root of T_i), introduce a vertex u and insert arc (u, v).

(b) Prune t_x, insert arc (u, x), regrafting t_x, and suppress the vertex with outdegree one. Call the resulting tree T_{i+1}. Note that $T_{i+1}|C = T|C$ (and similarly for $\bar{C}$).

(c) Subdivide the arc (u, x) with a vertex u'. Prune t_y, insert arc (u', y), regrafting t_y, and suppress the vertex with outdegree one. Call the resulting tree T_{i+2}. We now have a subtree $t_{u'}$ of T_{i+2} containing exactly the leaves in $\mathcal{L}(t_x) \cup \mathcal{L}(t_y)$. Note that $T_{i+2}|C = T|C$ (and similarly for $\bar{C}$).

We now prove by induction on $|C|$ that we can repeatedly perform the above sequence of rSPR operations to obtain, from T, a tree T' with subtree $T'|C = T|C$. In the case $|C| = 1$, $T' = T$ and so the result holds. Consider the case $|C| = 2$. Let $C = \{x, y\}$ and $\bar{C} = \mathcal{L}(T) \setminus \{x, y\}$. Starting with T, apply the above steps (a) through (c), where t_x and t_y are each a single leaf (x and y respectively), to obtain a tree $T_3 = T'$ with a subtree containing exactly the leaves in $\mathcal{L}(t_x) \cup \mathcal{L}(t_y) = \{x\} \cup \{y\} = C$, as required.

Assume that, for tree T with $2 \leq |C| \leq k$ (for some $k \geq 2$), we can perform a sequence of rSPR operations to obtain a tree T' with subtree $T'|C = T|C$. Now consider a tree T where $|C| = k+1$, and let t_v be the minimal subtree of T such that $C \subseteq \mathcal{L}(t_v)$. Then t_v has two maximal proper subtrees, say t' and t'', each of which must contain at least one leaf in C, and so each contains no more than k leaves in C. By the induction assumption, we obtain a tree T^* with subtree t_v^* containing subtrees $t'|C$ and $t''|C$. Now t_v^* is the minimal subtree of T^* containing $t'|C$ and $t''|C$, so we apply steps (a) through (c) above, starting with the tree T^*, to obtain a tree T' with a subtree containing exactly the leaves in $\mathcal{L}(t'|C) \cup \mathcal{L}(t''|C) = C$, as required.

We now prove that neither of the rSPR operations in steps (a) through (c) violate any rooted triples in $\mathcal{R}$. Note that t_x or t_y (or both) may consist of only a single leaf. Consider the rSPR operation with respect to t_x (given in steps (a) and (b)) used to obtain T_{i+1} from T_i $(1 \leq i \leq n-2)$. Assume that T_i displays $\mathcal{R}$, but suppose, without loss of generality, that T_{i+1} does not display rooted triple $ab|c \in \mathcal{R}$. Consider T_i and $a, b, c \in \mathcal{L}(T_i)$, and the possible locations of the leaves a, b, and c in T_i with respect to the subtrees t_x, t_y, and t_v. The scenarios in which $a, b \in \mathcal{L}(t_x)$ or $a, b \notin \mathcal{L}(t_x)$ result in contradictions. So, without loss of generality, consider the scenarios in which $a \in \mathcal{L}(t_x)$ and $b \notin \mathcal{L}(t_x)$. The cases in which $c \notin \mathcal{L}(t_v)$ or $b \notin \mathcal{L}(t_v)$ also result in contradictions, so assume that $b, c \in \mathcal{L}(t_v)$. If $b \in \mathcal{L}(t_y)$, then $c \notin \mathcal{L}(t_v)$, which is a contradiction, hence $b \notin \mathcal{L}(t_y)$. However, since $b \notin \mathcal{L}(t_x)$ and $b \notin \mathcal{L}(t_y)$, $b \notin C$, which is a contradiction of Lemma 3 because C is a maximal proper cluster of some tree in $\langle \mathcal{R} \rangle_B$. This concludes the case analysis. All possibilities result in contradictions, hence T_{i+1} displays $ab|c$. Now consider the rSPR operation with respect to t_y (given in step (c)) used to obtain T_{i+2} from T_{i+1}. Assume that T_{i+1} displays $\mathcal{R}$, but suppose, without loss of generality, that T_{i+2} does not display rooted triple $ab|c \in \mathcal{R}$. Consider T_{i+1} and $a, b, c \in \mathcal{L}(T_{i+1})$, and the possible locations of the leaves a, b, and c in T_{i+1} with respect to the subtrees t_x, t_y, and t_u. Similar reasoning, replacing t_v with t_u (recalling u is the parent of v in T_{i+1}) and swapping t_x and t_y, again leads to contradictions. Hence, T_{i+2} displays $ab|c$.

We now show how to obtain $\hat{T}$ from T'. In T', let w be the root of subtree $T|C$, and let x be the parent of w (x always exists as w is not the root of T', otherwise $|\bar{C}| = 0$). If x is the root of T', then $T|C$ is a maximal proper subtree of T' and so $\hat{T} = T'$. Otherwise, the following rSPR operation is performed to obtain $\hat{T}$ from T'. Let r' be the root of T'. Prune subtree $T|C$, introduce a vertex $\hat{r}$, insert arc $(\hat{r}, r')$ and arc $(\hat{r}, w)$, regrafting $T|C$, and suppress the vertex with outdegree one. We have now obtained a tree $\hat{T}$ (with root $\hat{r}$) with maximal proper subtrees $T|C$ and $T|\bar{C}$, as required.

We now prove that this rSPR operation does not violate any rooted triples in $\mathcal{R}$. Assume that T' displays $\mathcal{R}$ but suppose, without loss of generality, that $\hat{T}$ does not display rooted triple $ab|c \in \mathcal{R}$. Consider T' and $a, b, c \in \mathcal{L}(T')$, and the possible locations of the leaves a, b, and c in T' with respect to the subtree $T|C$. The scenarios in which $a, b \in \mathcal{L}(T|C)$ or $a, b \notin \mathcal{L}(T|C)$ result in contradictions. So, without loss of generality, consider the scenario in which $a \in \mathcal{L}(T|C)$ and $b \notin \mathcal{L}(T|C)$. Then $a \in C$ and $b \notin C$, which is a contradiction of Lemma 3. Hence $\hat{T}$ displays $ab|c$.

We have established that each of $(T_2, T_3 \ldots, T_n = \hat{T})$ display $\mathcal{R}$. Furthermore, since $T = T_1$ displays $\mathcal{R}$ by definition, we have shown that T_i displays $\mathcal{R}$ for all $1 \leq i \leq n$. $\square$

The process described in Lemma 4 will be referred to as *disentangling C*

from T.

We now have all the necessary preliminary results, so we return to the proof of Theorem 3.

Proof: [Proof of Theorem 3] We first prove the special case in which $\mathcal{L}(T) = \mathcal{L}(T') = \mathcal{L}(\mathcal{R})$ (i.e. there are no free leaves). We prove that we can obtain a sequence of NNI-related trees from T to T' for which each tree in the sequence displays $\mathcal{R}$. We use strong induction on $m = |\mathcal{L}(T)|$.

Consider the case $m = 2$. Then the two children of the root of T' are leaves, and since $\mathcal{L}(T) = \mathcal{L}(T')$, then $T = T'$ and so the result holds.

Assume that the result holds for trees T and T' with at most $k-1$ leaves. Now consider two trees $T, T' \in \langle\mathcal{R}\rangle_B$ with $|\mathcal{L}(T)| = |\mathcal{L}(T')| = |\mathcal{L}(\mathcal{R})| = k$. Let C and $\bar{C}$ be the maximal proper clusters of T'. Note that $\mathcal{L}(T) = C \cup \bar{C}$. Consider T and apply Lemma 4 to disentangle C from T, giving a sequence of rSPR-related trees from T to a tree T_i $(1 < i \leq n)$ such that each tree displays $\mathcal{R}$, and the tree T_i has maximal proper subtrees $T|C$ and $T|\bar{C}$. Applying Lemma 2 to this sequence of trees, we obtain a sequence S of NNI-related trees from T to T_i for which each tree in the sequence displays $\mathcal{R}$.

The tree T' has maximal proper subtrees $T'|C$ and $T'|\bar{C}$. Let $\mathcal{R}_C = \{ab|c \in \mathcal{R} : a, b, c \in C\}$ (i.e. the set of rooted triples for which the leaves are all in C). Define $\mathcal{R}_{\bar{C}}$ similarly. Now note that $|\mathcal{L}(T|C)| = |\mathcal{L}(T'|C)| < k$ so, by the induction assumption, there is a sequence of NNI-related trees $(T|C, \ldots, T'|C)$ such that each tree in the sequence displays $\mathcal{R}_C$. Since $|\mathcal{L}(T|\bar{C})| = |\mathcal{L}(T'|\bar{C})| < k$, the same applies to $\bar{C}$.

Consider the sequence of NNI operations above that create the sequence $(T|C, \ldots, T'|C)$. Starting with tree T_i with subtree $T|C$, perform this sequence of NNI operations to obtain a tree T_j $(1 < i \leq j \leq n)$ with subtree $T'|C$, where the rest of the tree remains unchanged (that is, $T_i \setminus (T|C)$ is equivalent to $T_j \setminus (T'|C)$). Since each tree in the sequence $(T_i, \ldots, T_j)$ displays $\mathcal{R}_C$, and $T_i \setminus T|C$ is equivalent to $T_j \setminus T'|C$, each tree this sequence displays $\mathcal{R}$. Repeating this process for $\bar{C}$ (with set of rooted triples $\mathcal{R}_{\bar{C}}$), starting with the tree T_j, gives the tree T' with maximal proper subtrees $T'|C$ and $T'|\bar{C}$, as required. We now have a sequence of NNI-related trees from T_i to T' such that each tree in the sequence displays $\mathcal{R}$.

Combining this sequence of trees with the earlier sequence S, we obtain a sequence of NNI-related trees $(T, \ldots, T')$ such that each tree in the sequence displays $\mathcal{R}$, as required.

We now turn to the general case in which $\mathcal{L}(\mathcal{R}) \subseteq \mathcal{L}(T) = \mathcal{L}(T')$. By Lemma 1, there is a sequence of NNI-related trees from T to a tree $\widetilde{T}$ with subtree $T|\mathcal{L}(\mathcal{R})$, and there is a sequence of trees from T' to a tree $\widetilde{T'}$ with subtree $T'|\mathcal{L}(\mathcal{R})$, such that each tree in these sequences displays $\mathcal{R}$ and $\widetilde{T} \setminus (T|\mathcal{L}(\mathcal{R}))$ is equivalent to $\widetilde{T'} \setminus (T'|\mathcal{L}(\mathcal{R}))$.

Next, we need a sequence of NNI-related trees from $\widetilde{T}$ to $\widetilde{T'}$ such that each tree in the sequence displays $\mathcal{R}$. By the special case (proved above), there is a sequence of NNI-related trees from $T|\mathcal{L}(\mathcal{R})$ to $T'|\mathcal{L}(\mathcal{R})$ such that each tree in the sequence displays $\mathcal{R}$. Performing the corresponding sequence of NNI operations, starting with the tree $\widetilde{T}$, transforms the subtree $T|\mathcal{L}(\mathcal{R})$ into $T'|\mathcal{L}(\mathcal{R})$, giving the tree $\widetilde{T'}$. We now have a sequence of NNI-related trees from $\widetilde{T}$ to $\widetilde{T'}$ such that each tree displays $\mathcal{R}$, as required. □

4 Algorithm

In this section, we take the steps from the proofs of Lemmas 1, 2,and 4 and create an algorithm which takes two trees $T, T' \in \langle\mathcal{R}\rangle_B$ as input and produces a sequence of NNI-related trees from T to T', such that each tree in the sequence displays $\mathcal{R}$. The algorithm consists of the following five procedures.

The first procedure, FreeLeaves, takes a tree T and a set of rooted triples $\mathcal{R}$ as input and uses steps (a) through (c) in the proof of Lemma 1 to create a sequence of NNI-related trees ending with a tree $\widetilde{T}$which has a subtree containing exactly the leaves in $\mathcal{R}$ (as illustrated in Figure 5). It returns the sequence of NNI-related trees.

Procedure FreeLeaves:
Input: A set $\mathcal{R}$ of rooted triples; a tree $T \in \langle\mathcal{R}\rangle_B$.
Output: A sequence F of NNI-related trees from T to a tree with subtree $T|\mathcal{L}(\mathcal{R})$.

1. Label the free leaves $\mathcal{L}(T) \setminus \mathcal{L}(\mathcal{R}) = \{x_1, x_2, \dots x_n\}$.
2. Apply steps (a) through (c) in the proof of Lemma 1 where T is the starting tree and the free leaves are labeled as in step 1 to produce F.
3. Return F.

The second procedure, ToNNI, uses steps (a) through (e) in the proof of Lemma 2 to produce a sequence of NNI-related trees from a sequence of rSPR-related trees.

Procedure ToNNI:
Input: A sequence $S = (S_1, \dots, S_k)$ of rSPR-related trees; a set $\mathcal{R}$ of rooted triples displayed by each tree in S.
Output: A sequence $\widetilde{S}$ of NNI-related trees.

1. Let $\widetilde{S} = (\ )$.
2. For $i = 1, \dots, k-1$:
   - (i) Apply steps (a) through (e) from the proof of Lemma 2 (given in the appendix) to the trees S_i and S_{i+1} to obtain a sequence U_i of NNI-related trees, where S_i is the first tree in the sequence U_i and S_{i+1} is the last tree in the sequence U_i. Each tree in the sequence U_i displays $\mathcal{R}$ by Lemma 2.
   - (ii) If $i \neq k$, remove the last tree (S_{i+1}) from U_i (so that it will not be repeated) and append U_i to $\widetilde{S}$.
3. Return $\widetilde{S}$.

The third procedure, Disentangle, takes a tree $T_{current}$ as input and uses steps (a) through (c) in the proof of Lemma 4 to disentangle a given leaf set from a specified subtree of $T_{current}$, and returns the resulting sequence of rSPR-related trees.

Procedure Disentangle:

Input: A set $\mathcal{R}$ of rooted triples; a tree $T_{current} \in \langle\mathcal{R}\rangle_B$; the root w of the subtree of $T_{current}$ to disentangle; the leaf set C to disentangle.
Output: A sequence S of rSPR-related trees.

1. Let t_w be the subtree of $T_{current}$ rooted at w. Let $S = (\ )$ and let $T_{working} = T_{current}$.
2. While $T_{working}$ does not have subtree $T_{current}|C = T_{working}|C$:
   Let t_w be the subtree of $T_{working}$ rooted at w. There is a minimal subtree of t_w containing exactly two maximal subtrees t_x and t_y, whose leaf sets are subsets of C (note that t_x or t_y may contain only a single leaf). Perform steps (a) through (c) in the proof of Lemma 4, starting with the tree $T_{working}$, to obtain two trees T^* and T^{**}, where T^{**} has a subtree with leaf set $\mathcal{L}(t_x) \cup \mathcal{L}(t_y)$. Append T^* and T^{**} to S. Let $T_{working} = T^{**}$.
3. The tree $T_{working}$ now has subtree $T_{current}|C$. Consider the root r_C of the subtree $T_{current}|C$ of $T_{working}$ and the root r_L of the subtree $T_{working}|\mathcal{L}(\mathcal{R})$ of $T_{working}$. If r_C is not a child of r_L, perform one more rSPR operation to prune the subtree $T_{current}|C$ and regraft it to a vertex subdividing the arc between r_L and its parent, giving tree $\hat{T}$. Append $\hat{T}$ to S.
4. Return S.

The fourth procedure, TraverseTree, is a recursive procedure that traverses a tree depth-first, calls the procedure Disentangle for each subtree, and combines the resulting sequences of trees. It then returns the entire sequence of rSPR-related trees produced by all of the recursive calls.
Procedure TraverseTree:
Input: a set $\mathcal{R}$ of rooted triples; a tree $T_{current} \in \langle\mathcal{R}\ \rangle_B$ to traverse; the root w of a subtree of $T_{current}$; a tree $T' \in \langle\mathcal{R}\rangle_B$.
Output: A sequence S of rSPR-related trees from $T_{current}$ to a tree with subtree T'.

1. Let $S = (\ )$ and let t_w be the subtree of $T_{current}$ rooted at w. If $|\mathcal{L}(t_w)| \in \{1, 2\}$, go to step 5 (i.e. if t_w consists of only a single leaf or a cherry, return an empty sequence.)
2. Let C be a maximal proper cluster of T'.
3. Do $S = S$+Disentangle$(\mathcal{R}, T_{current}, w, C)$. If $|S| \neq 0$, let $T_{current}$ be the last tree in the sequence S.
4. For $A \in \{C, \bar{C}\}$ (where $\bar{C}$ is the complement of C with respect to $\mathcal{L}(t_w)$):
   (i) Do $S = S$+TraverseTree$(\mathcal{R}_A, T_{current}, r_A, T'|A)$, where $\mathcal{R}_A = \{ab|c \in \mathcal{R} : a, b, c \in A\}$ and r_A is the root of the subtree $T_{current}|A$ of $T_{current}$ (the subtree containing exactly the leaves in A).
   (ii) If $|S| \neq 0$, let $T_{current}$ be the last tree in the sequence S.
5. Return S.

The last procedure, TreeSequence, takes two trees, T and T', as input and uses all of the above procedures to produce a sequence of NNI-related trees from T to T' such that each tree in the sequence displays $\mathcal{R}$. TreeSequence first calls FreeLeaves with input T (respectively T') to produce a sequence of NNI-related trees from T to a tree $\widetilde{T}$ (respectively from T' to a tree $\widetilde{T'}$). TraverseTree is then applied to produce a sequence of rSPR-related trees from $\widetilde{T}$ to $\widetilde{T'}$, which ToNNI converts into a sequence of NNI-related trees. Lastly, these three sequences are combined to produce the required sequence of NNI-related trees from T to T'.

Procedure TreeSequence:

Input: Two rooted binary phylogenetic trees, T and T'.

Output: A sequence of NNI-related trees from T to T' such that each tree in the sequence displays $\mathcal{R}$.

1. Let $\mathcal{R} = r(T) \cap r(T')$, so $T, T' \in \langle \mathcal{R} \rangle_B$. Let $L = \mathcal{L}(\mathcal{R})$. Do $F =$ FreeLeaves$(\mathcal{R}, T)$ and $F' =$ FreeLeaves$(\mathcal{R}, T')$. Let $\widetilde{T}$ and $\widetilde{T'}$ be the last trees in the sequences F and F' respectively. Note that $\widetilde{T} \setminus (T|L)$ is equivalent to $\widetilde{T'} \setminus (T'|L))$.
2. Reverse F' and call this sequence of trees $\overleftarrow{F'}$.
3. Do $S =$ TraverseTree$(\mathcal{R}, \widetilde{T}, r_{T|L}, \widetilde{T'})$, where $r_{T|L}$ is the root of the subtree $T|L$ of $\widetilde{T}$. Now S is a sequence of rSPR-related trees from $\widetilde{T}$ to $\widetilde{T'}$ for which each tree in the sequence displays $\mathcal{R}$.
4. Do $\widetilde{S} =$ ToNNI(S).
5. Return $F + \widetilde{S} + \overleftarrow{F'}$, which is a sequence of NNI-related trees from T to T' satisfying the required properties.

4.1 Complexity

In this section we calculate the complexity of each procedure. We start by noting that one rSPR operation is $O(1)$, as is one NNI operation. Recall that $T, T' \in \langle \mathcal{R} \rangle_B$ and $\mathcal{L}(\mathcal{R}) \subseteq \mathcal{L}(T) = \mathcal{L}(T')$. Let $n = |\mathcal{L}(T)|$. Let $n_R = |\mathcal{L}(\mathcal{R})|$, the number of leaves in $\mathcal{R}$, and let $n_F = |\mathcal{L}(T)| - n_R$, the number of free leaves in T, so that $n = n_R + n_F$.

First consider the procedure FreeLeaves. This procedure uses NNI operations to produce, from T, a tree with subtree $T|\mathcal{L}(\mathcal{R})$, as described in the procedure. Let $D = d(T)$ be the depth of T. For tree T, each leaf requires $O(D)$ NNI operations. There are n_F leaves for which this must be repeated, so this procedure is $O(n_F D)$.

Next, we consider the procedure ToNNI applied to a sequence $S = (S_1, \ldots, S_k)$ of rSPR-related trees. This procedure produces from S a sequence of NNI-related trees. Let $D_S = \max\{d(S_i) \text{ for } 1 \leq i \leq k\}$. Each rSPR operation corresponds to at most $2D_S$ NNI operations since, in the worst case, arcs e and f (given in the definition of an rSPR operation) are distance $2D_S - 2$ apart. Therefore, each consecutive pair of trees in S produces a sequence of up to $2D_S$ NNI-related trees. There are $k - 1$ consecutive pairs of trees in S, so this procedure is $O(kD_S) = O(|S|D_S)$.

Consider the procedure Disentangle. Let t_w be the subtree of $T_{current}$ rooted at w. Disentangling the leaf set C from t_w requires up to $2|C| - 1$ rSPR operations. Therefore, the total number of rSPR operations required is at most $2|C| - 1$. Since $2|C| - 1 < 2|C| < 2n_{\mathcal{R}}$, the procedure Disentangle is $O(n_{\mathcal{R}})$.

Now consider the procedure TraverseTree. The maximum recursion depth is $n_{\mathcal{R}}$. The call to the procedure Disentangle in step 3 is $O(n_{\mathcal{R}})$, as described above. Step 4 is $O(|C|) \times O(n_{\mathcal{R}}) + O(|\bar{C}|) \times O(n_{\mathcal{R}}) = (O(|C|) + O(|\bar{C}|)) \times O(n_{\mathcal{R}}) = O(n_{\mathcal{R}}) \times O(n_{\mathcal{R}}) = O(n_{\mathcal{R}}^2)$. So the overall complexity of the procedure TraverseTree is $O(n_{\mathcal{R}}) + O(n_{\mathcal{R}}^2) = O(n_{\mathcal{R}}^2)$.

Lastly, consider the procedure TreeSequence. Let $D_T = \max\{d(T), d(T')\}$. Step 1 is $O(n_F D_T)$ (two calls to the procedure FreeLeaves). This gives two sequences of trees, F and F', where the length of F' is $An_F D_T$ for some constant A. Step 2 is therefore $O(n_F D_T)$, reversing the sequence F'. Step 3 is $O(n_{\mathcal{R}}^2)$ (call to the procedure TraverseTree). This gives a sequence of trees S' of length $Bn_{\mathcal{R}}^2$ for some constant B. Step 4 is $O(|S'|D_{S'}) = O(Bn_{\mathcal{R}}^2 D_{S'}) = O(n_{\mathcal{R}}^2 D_{S'})$ (call to the procedure ToNNI). Step 5 concatenates three sequences, the complexity of which can be $O(1)$ (depending on the implementation). Therefore, the procedure TreeSequence is $O(n_{\mathcal{R}}^2 D_{S'} + n_F D_T)$. Letting $D_{max} = \max\{D_{S'}, D_T\}$ and noting that $n_{\mathcal{R}} \leq n$ and $n_F \leq n$, the complexity is $O(n^2 D_{max})$.

Hence, producing a sequence of NNI-related trees from T to T' has a complexity of $O(D_{max} n^2)$.

5 Concluding comments

In this paper, we have provided an explicit polynomial-time procedure for moving between any two trees on a phylogenetic 'terrace' using elementary (NNI) operations, so that each tree in the sequence also belongs to the terrace. Thus if two trees have an optimal score under some linear scoring function satisfying Eqn.(1), each tree in the sequence is also optimal. Of course, there are likely to be many other such sequences between the two trees that also lie on the terrace, so having some way of quantifying this number would be interesting. A further question, that is particularly relevant to many applications, asks for the development of a polynomial-time approximation procedure for sampling the trees on a terrace uniformly at random (or, equivalently, the trees that display a set of rooted triples). An approach based on random NNI or rSPR walks (sequences of NNI or rSPR operations) that move between trees on the terrace may provide a way to approach this problem; this was, in part, the motivation for our study. The development of an efficient randomized sampling scheme for trees on a terrace seems a worthy topic for further study.

Acknowledgements

We thank the Biomathematics Research Centre for partial funding of the work presented in this paper, and the two anonymous reviewers for helpful comments on an earlier version of this paper.

References

[1] M. Bordewich. *The Complexity of Counting and Randomised Approximation*. PhD thesis, University of Oxford, 2003.

[2] O. Chernomor, B. Q. Minh, and A. von Haeseler A. Consequences of common topological rearrangements for partition trees in phylogenomic inference. *J. Comput. Biol.*, 22:1–14, 2015. doi:10.1089/cmb.2015.0146.

[3] J. Felsenstein. *Inferring phylogenies*. Sinauer Press, 2004.

[4] W. Jetz, G. H. Thomas, J. B. Joy, D. W. Redding, K. Hartmann, and A. O. Mooers. Global distribution and conservation of evolutionary distinctness in birds. *Curr. Biol.*, 24:1–12, 2014. doi:10.1016/j.cub.2014.03.011.

[5] D. R. Maddison. The discovery and importance of multiple islands of most-parsimonious trees. *Syst. Zool.*, 40:315–328, 1991. doi:10.1093/sysbio/40.3.315.

[6] M. J. Sanderson, M. M. McMahon, A. Stamatakis, D. Zwickl, and M. Steel. Impacts of terraces on phylogenetic inference. *Syst. Biol.*, page syv024, 2015. doi:10.1093/sysbio/syv024.

[7] M. J. Sanderson, M. M. McMahon, and M. Steel. Terraces in phylogenetic tree space. *Science*, 333:448–450, 2011. doi:10.1126/science.1206357.

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

Appendix: Proof of Lemma 2

For convenience, we restate Lemma 2 here.

Suppose that $\mathcal{R}$ is a set of rooted triples and $T, T' \in \langle\mathcal{R}\rangle_B$, with $\mathcal{L}(T) = \mathcal{L}(T') = \mathcal{L}(\mathcal{R})$. Suppose that T' is obtained from T by one rooted subtree prune-and-regraft operation. Then there is a sequence of trees $(T_0, T_1, \ldots, T_n)$ such that:

1. *$T_0 = T$ and $T_n = T'$,*
2. *$T_i \overset{\mathrm{NNI}}{\sim} T_{i+1}$ for $0 \le i < n$, and*
3. *$T_i \in \langle\mathcal{R}\rangle_B$ for $0 \le i \le n$ (i.e. each T_i displays $\mathcal{R}$).*

Proof: Consider the trees T and T'. By our assumption, $T \overset{\mathrm{rSPR}}{\sim} T'$. If $T = T'$ or $T \overset{\mathrm{NNI}}{\sim} T'$, we are done. So suppose that this is not the case. Let the rSPR operation be with respect to some subtree t_u of T. Then T and T' both have subtree t_u. Let v_0 be the parent of u in T (the vertex v_0 always exists because t_u is a proper subtree of T). Similarly, let v' be the parent of u in T'. When defining the neighbours of v' in T', there are two cases to consider. The first case is that v' has three neighbours, u, v_n, and v_{n+1}. Note that v_n and v_{n+1} are both in T, so there is a path $v_0, \ldots, v_n, v_{n+1}$ in T. The second case is that v' has two neighbours, u and v_n. In this case, v' is the root of T'. As v_n is in T, there is a path $v_0, \ldots, v_n$ in T.

We now describe the minimum sequence of NNI operations with respect to t_u that is performed to obtain T' from T.

a) In the first NNI operation, delete the arc (v_0, u) from T, subdivide the arc $\{v_1, v_2\}$ with a vertex w_1, insert arc (w_1, u), and then suppress v_0. Call the resulting tree T_1.

b) For $i = 2, \ldots, n-1$, perform the following (i^{th}) NNI operation: Delete the arc (w_{i-1}, u) from T_{i-1}, subdivide the arc $\{v_i, v_{i+1}\}$ with a vertex w_i, insert arc (w_i, u), and then suppress the vertex w_{i-1}. Call the resulting tree T_i.

c) The last (n^{th}) NNI operation is as follows. If v_n is the root of T_{n-1} and v' is the root of T', delete the arc (w_{n-1}, u) from T_{n-1}, introduce a vertex w_n, and insert arc (w_n, v_n). (Note that, in this case, w_n is the root of the resulting tree.) Otherwise, delete the arc (w_{n-1}, u) from T_{n-1}, and subdivide the arc $\{v_n, v_{n+1}\}$ with a vertex w_n.

d) Insert arc (w_n, u) and then suppress the vertex w_{n-1}. The resulting tree is T', where $w_n = v'$.

We now have a sequence of NNI-related trees from T to T'. Next, we prove that for each i ($0 \le i \le n$), $T_i \in \langle\mathcal{R}\rangle_B$. Let r_i denote the root of T_i for each i. We proceed using induction on i and note that $T_0 = T \in \langle\mathcal{R}\rangle_B$ by assumption.

Assume that $T_i \in \langle\mathcal{R}\rangle_B$ for some $i < n$. To see that $T_{i+1} \in \langle\mathcal{R}\rangle_B$, consider a rooted triple $ab|c \in \mathcal{R}$. Recall that a tree displays $ab|c$ if and only if the path from a to b does not intersect the path from c to the root of the tree. This is the case in T_i and we show that it is also the case in T_{i+1}. There are six possible cases to be considered with respect to the possible locations of a, b, and c in T_i. Let v_{ab} be the most recent common ancestor of a and b in T_i and let v_{abc}

be the most recent common ancestor of a, b, and c in T_i. Recall that all of the NNI operations are with respect to t_u.

Case 1: $a, b, c \in t_u$. In this case, in T_i, the (a-b)-path (the path from a to b) does not intersect the (c-r_i)-path, so, in t_u, the (a-b)-path does not intersect the (c-u)-path. The tree T_{i+1} contains the subtree t_u, so, as before, the (a-b)-path does not intersect the (c-u)-path and hence, in T_{i+1}, the (a-b)-path does not intersect the (c-r_{i+1})-path. Therefore, T_{i+1} displays $ab|c$.

Case 2: $a, b, c \notin t_u$. In this case, in T_i, no element of the (a-b)-path, or the (c-r_i)-path, is in t_u. Let q be the path between v_{ab} and v_{abc} in T_i and let p be a path in T_i with one endpoint in the (a-b)-path and the other endpoint in the (c-r_i)-path. Then p must contain q, so $|p| \geq |q|$. Therefore, the (a-b)- and (c-r_i)-paths intersect if and only if q has length zero, in which case $v_{ab} = v_{abc}$. Assume that T_{i+1} does not display $ab|c$, so the (a-b)- and (c-r_{i+1})-paths intersect and so, by the same argument, $v_{ab} = v_{abc}$. Then, in T_{i+1}, v_{abc} has three children and so T_{i+1} is not a binary tree. This is a contradiction because, by definition, an NNI operation on a binary tree produces another binary tree. Therefore T_{i+1} displays $ab|c$.

Case 3: $a, b \in t_u$ and $c \notin t_u$. In this case, in T_i, the (a-b)-path is contained entirely in t_u and the (c-r_i)-path contains no arcs or vertices in t_u, so these two paths do not intersect. Performing an NNI operation on T_i to obtain T_{i+1} will not affect this, so the same property will hold in T_{i+1}. In T_{i+1}, the (a-b)-path will be contained entirely in t_u and the (c-r_{i+1})-path will not contain any of these vertices or arcs, so these two paths do not intersect. Therefore, T_{i+1} displays $ab|c$.

Case 4: $b, c \in t_u$ and $a \notin t_u$ (which is analogous to the case $a, c \in t_u$, $b \notin t_u$). In this case, in T_i, the (a-b)-path and the (c-r_i)-path both contain u, so these two paths intersect. Therefore, T_i does not display $ab|c$, which is a contradiction. It is thus not possible that $b, c \in t_u$ and $a \notin t_u$.

Case 5: $a \in t_u$ and $b, c \notin t_u$ (which is analogous to the case $b \in t_u$, $a, c \notin t_u$). Let v'_{ab} be the most recent common ancestor of a and b in T_{i+1} and let v'_{abc} be the most recent common ancestor of a, b, and c in T_{i+1}. First assume that, in T_{i+1}, v'_{ab} is a proper descendant of v'_{abc}. Then T_{i+1} displays $ab|c$. Now assume that this is not the case; that is, in T_{i+1}, v'_{ab} is not a proper descendant of v'_{abc}. Then the (a-b)-path and the (c-r_{i+1})-path intersect. Furthermore, in T_{i+1}, both the (a-b)-path and the (c-r_{i+1})-path contain v_{abc}. Now for each T_k, where $k > i$, both the (a-b)-path and the (c-r_k)-path contain v_{abc}. Hence, in $T_n = T'$, both the (a-b)-path and the (c-r_n)-path contain v_{abc}, so these two paths intersect, a contradiction of the assumption that $T' \in \langle\mathcal{R}\rangle_B$. Therefore, T_{i+1} displays $ab|c$.

Case 6: $c \in t_u$ and $a, b \notin t_u$. Let v'_{ab} and v'_{abc} be defined as in case 5. First assume that, in T_{i+1}, v'_{abc} is a proper ancestor of v'_{ab}. Then T_{i+1} displays $ab|c$. Now assume that this is not the case; that is, in T_{i+1}, v'_{abc} is not a proper ancestor of v'_{ab}. Then u (and therefore c) is a descendant of v_{ab}. So the (a-b)-path and the (c-r_{i+1})-path intersect. Furthermore, in T_{i+1}, both the (a-b)-path and the (c-r_{i+1})-path contain v_{ab}. Now for each T_k, where $k > i$, both the (a-b)-path and the (c-r_k)-path contain v_{ab}. Hence, in $T_n = T'$, both the (a-b)-path and the (c-r_n)-path contain v_{ab}, so these two paths intersect, a contradiction of the assumption that $T' \in \langle \mathcal{R} \rangle_B$. Therefore, T_{i+1} displays $ab|c$.

In each of the six cases, either the scenario is not possible (case 4) or T_{i+1} displays $ab|c$. Since $ab|c$ was an arbitrary rooted triple in $\mathcal{R}$, we can extend this result to all of the rooted triples in $\mathcal{R}$, so $T_{i+1} \in \langle \mathcal{R} \rangle_B$. Therefore, by induction, $T_j \in \langle \mathcal{R} \rangle_B$ for all $0 \leq j \leq n$. $\square$