# Ranked Subtree Prune and Regraft

Lena Collienne ✉[1], Chris Whidden [2], and Alex Gavryushkin [1,3]

[1]Biological Data Science Laboratory, School of Mathematics and Statistics, University of Canterbury, New Zealand
[2]Faculty of Computer Science, Dalhousie University, Canada
[3]Biomathematics Research Centre, University of Canterbury, New Zealand

✉lena@lenacoll.de
December 5, 2023

**Abstract**

Phylogenetic trees are a mathematical formalisation of evolutionary histories between organisms, species, genes, cancer cells, etc. For many applications, e.g. when analysing virus transmission trees or cancer evolution, (phylogenetic) time trees are of interest, where branch lengths represent times. Computational methods for reconstructing time trees from (typically molecular) sequence data, for example Bayesian phylogenetic inference using Markov Chain Monte Carlo (MCMC) methods, rely on algorithms that sample the treespace. They employ tree rearrangement operations such as SPR (Subtree Prune and Regraft) and NNI (Nearest Neighbour Interchange) or, in the case of time tree inference, versions of these that take times of internal nodes into account. While the classic SPR tree rearrangement is well-studied, its variants for time trees are less understood, limiting comparative analysis for time tree methods.

In this paper we consider a modification of the classical SPR rearrangement on the space of ranked phylogenetic trees, which are trees equipped with a ranking of all internal nodes. This modification results in two novel treespaces, which we propose to study. We begin this study by discussing algorithmic properties of these treespaces, focusing on those relating to the complexity of computing distances under the ranked SPR operations as well as similarities and differences to known tree rearrangement based treespaces. Surprisingly, we show the counterintuitive result that adding leaves to trees can actually decrease their ranked SPR distance, which may have an impact on the results of time tree sampling algorithms given uncertain "rogue taxa".

This work was partially supported by Royal Society Te Apārangi through a Rutherford Discovery Fellowship (UOC1702) and a Marsden grant (21-UOC-057), and by Ministry of Business, Innovation, and Employment of New Zealand through an Endeavour Smart Ideas grant (UOOX1912) and a Data Science Programmes grant (UOAX1932).

# 1 Introduction

Phylogenetic trees are used to display evolutionary relationships, for example between organisms, species, or genes, and are usually inferred from DNA, RNA, amino acid, or other types of sequence data. Typical applications include reconstructing the evolutionary history of a set of species, e.g. to construct the tree of life [24], analysing transmission patterns of viruses [30, 34], and investigating cancer evolution [3]. The goal is the same in all applications: finding a phylogenetic tree that best explains evolutionary relationship between given (sequence) data, represented by the leaves of the tree.

Many methods for inferring phylogenetic trees from sequence data use tree sampling or treespace search algorithms. These include Bayesian methods, for example the software packages BEAST [14], BEAST2 [7], MrBayes [23], and RevBayes [22], and maximum likelihood methods like IQ-TREE [27], PhyML [18], or RAxML [33]. All these methods rely on tree proposals: for a given tree, a new tree similar to the current tree is proposed and is accepted if it fulfils certain conditions. If accepted, the current tree is updated to be this new tree and the procedure repeated. Tree rearrangement operations, which apply local changes to a tree, are commonly used for these tree proposals. Most popular are the Subtree Prune and Regraft (SPR) and Nearest Neighbour Interchange (NNI) tree rearrangements, the latter being a version of SPR restricted to being more local.

An SPR operation (or move) cuts an edge of a phylogenetic tree and reattaches the thereby detached subtree at a different position in the tree. This tree rearrangement (or a version of it) is implemented in many tree inference software packages, including maximum likelihood methods [18, 29, 33], Bayesian inference methods [14, 22, 28] and also recent parsimony-based methods that are able to infer large-scale phylogenies [39]. One reason for the popularity of SPR moves in tree search algorithms is that, unlike very local NNI moves, SPR moves can be used to jump across wider regions of treespace, preventing tree search algorithms from getting stuck in local optima [18, 27]. Similarly, it has been shown for Bayesian inference methods that SPR moves, or modifications thereof, can speed up convergence of Markov chains and improve their mixing when used in combination with other operators [20]. There is however a major obstacle for interpreting how well treespace is traversed when using SPR moves for tree search and tree sampling algorithms: computing the SPR distance, i.e. the minimum number of SPR moves required to transform one tree into another, is $\mathcal{NP}$-hard [5, 19]. Despite this $\mathcal{NP}$-hardness result, fixed-parameter tractable algorithms for computing SPR distances exist [35], which can be used for analysing modes of posterior distributions [36] or for computing supertrees [35, 37]. Understanding properties of the SPR treespace, which can be viewed as a graph where vertices represent trees that are connected by edges if the trees are connected by an SPR operation, has proven useful for getting a better understanding of phylogenetic inference methods [37].

Analysing the geometry of the classic SPR treespace has lead to numerous results for SPR on rooted and unrooted trees [37, 38]. Although SPR operations are defined in the same way for both types of trees and the two resulting treespaces share most of their

properties, the techniques to prove those properties differ significantly. $\mathcal{NP}$-hardness of the problem of computing distances, for example, has first been shown for the rooted SPR [5] and only three years later for the unrooted case [19]. An important mathematical structure for both of these proofs is the Maximum Agreement Forest (MAF). A MAF for two trees results from deleting a minimum number of edges in each tree so that the two resulting graphs (forests) are isomorphic. The number of connected components of a MAF for two rooted trees coincides with their rooted SPR distance [5]. For unrooted trees, however, this relationship between the SPR distance and the number of connected components of the MAF does not hold [2, 37], which was the reason some erroneous proofs of $\mathcal{NP}$-hardness made it to the literature. It also has been found that the maximum distance under both rooted and unrooted SPR is linear in the number of leaves [4, 12], while the number of trees in the 1-neighbourhood, i.e. the number of trees resulting from one SPR move, is quadratic in the number of leaves [2, 31]. The latter is an important property that has been used to determine the curvature of the treespace by Whidden and Matsen [38] in order to analyse mixing properties of Markov Chain Monte Carlo methods, which are used for Bayesian tree inference.

The majority of these results have been derived for trees that do not contain timing information of evolutionary events. We refer to trees where branch lengths represent times, meaning that the evolutionary events represented by internal nodes are dated, as time trees. These time trees are of interest in many applications. Software packages like BEAST [14] and BEAST2 [7], for example, infer time trees and rely on tree proposals that incorporate timing information of evolutionary events. A version of SPR moves for time trees has been introduced by Höhna, Defoin-Platel, and Drummond [20]. The authors analysed the suitability of this move as a tree proposal operator for Bayesian inference using MCMC algorithms and showed it performed better than some of the previous operators in isolation but it was still better to use a combination. As of May 2023, this move is default in BEAST[1]. A guided version of this proposal was introduced in [21], where guiding the choice of destination improved the acceptance ratio. Another version of SPR moves, which works simultaneously on a species tree and multiple gene trees, can be found in the Stacey package for BEAST2 [25]. Even though SPR moves are widely used as tree proposals and have been adapted to work for time trees, little research has gone into SPR treespaces that take times of evolutionary events into account.

A version of SPR moves for time trees that has been studied mathematically is the one introduced by Song [32], where ranked trees are considered. Ranked trees (which are called ordered trees in [32]) are rooted phylogenetic trees with internal nodes ordered according to times of corresponding evolutionary events and leaves are assumed to be sampled at the same time (ultrametric). The SPR moves defined by Song [32] can move a subtree to a different place in the tree under the condition that the rank of the reattachment node is greater than the rank of the root of the moved subtree. Bounds for neighbourhood sizes

[1] `https://beast.community/xml_reference#ltfixednodeheightsubtreepruneregraftgt-element`

and diameters are provided [32], but no further results are known. Despite these efforts to investigate SPR moves for ranked trees, there is still a gap for analysing tree inference methods using SPR for time trees, as the version of SPR moves as defined by Song [32] is currently not used in phylogenetic inference.

In this paper we fill this gap by considering an alternative definition of SPR moves for ranked trees, where we require the height (i.e. rank) at which a subtree is cut and reattached to be the same. This restriction on the rank reattachment is inspired by ranked SPR moves in phylogenetic inference software: The Fixed Node Prune and Regraft move introduced in Höhna, Defoin-Platel, and Drummond [20] as tree proposal for Bayesian inference methods is the extension of our ranked SPR moves to time trees. We call these tree rearrangements Horizontal SPR (HSPR) moves and the corresponding treespace HSPR. Motivated by the importance of this move in computational phylogenetics, we study mathematical properties of this treespace focusing on the metric space of ranked trees that is given by the HSPR move. Studying this treespace helps us understand its fundamental properties, which is important to analyse tree inference algorithms using HSPR for tree proposals as well as posterior distributions of time trees output by BEAST or BEAST2, similar to how it has been done in [36]. By additionally allowing rank moves, which swap the order of two nodes in a ranked tree, we define a further metric space called RSPR (Ranked SPR). We include rank moves in the same way as it has been done in a variation of NNI introduced in [16] for ranked trees (RNNI) that allows distances to be computed in polynomial time [10], unlike in the classical NNI treespace, where this problem is $\mathcal{NP}$-hard. This suggests that the complexity of computing distances can be different in classical SPR and its ranked version.

This paper is structured as follows. We introduce notations and define the new treespaces in Section 2 before discussing some of its fundamental properties in Section 3. This especially includes the cluster property, which a treespace possesses if shortest paths between trees preserve shared information in the form of common clusters. This property has proven to be important for the polynomial time algorithm in RNNI [10] and for fixed-parameter tractable algorithms in SPR [26, 37]. The discussion of the cluster property is followed by some observations on the shape of shortest paths and the relationship of RSPR and HSPR shortest paths (Section 4). Finally, we establish a surprising result on how the distance between two trees can change after adding a new leaf to them (Section 5). We provide an open source implementation of horizontal SPR moves [8], which we use to show some properties of ranked SPR spaces computationally.

## 2 Preliminaries

A *rooted binary phylogenetic tree* is a pair $T = (t, \phi)$ where $t$ is a rooted binary tree and $\phi : L(t) \to X$ is a bijective map from the set of leaves $L(t)$ of $t$ to a set of labels $X = \{l_1, l_2, \ldots, l_n\}$. Throughout this paper, we refer to rooted binary phylogenetic tree simply as *rooted trees*, and we assume that all trees have $n$ leaves, unless stated otherwise.

Let $T$ be a rooted tree that contains an edge $e = (u, v)$ and let $T|_v$ be the subtree of $T$ rooted in $v$. A *subtree prune and regraft* (SPR) move on $e$ in $T$ transforms this tree into a new rooted binary rooted phylogenetic tree by the following three steps:

(i) *Prune the subtree* $T|_v$: delete the edge $e$ from $T$, resulting in two connected components $T|_v$ and $T_\rho$, where $T_\rho$ contains the root $\rho$ of $T$.

(ii) Suppress the resulting node of degree two in $T_\rho$ (node $u$), so that $T_\rho$ is a rooted tree.

(iii) *Reattach* $T|_v$: either by introducing a new node $w$ on an edge $f$ in $T_\rho$ and adding an edge $(w, v)$, or by adding a new root $\rho'$ and adding edges $(\rho', \rho)$ and $(\rho', v)$.

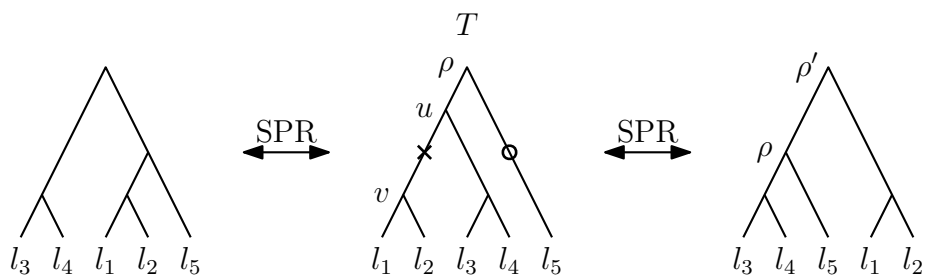


Figure 1: Two rooted SPR moves. The cross marks the edge that is cut to prune the subtree $T|_v$ of $T$ with leaf set $\{l_1, l_2\}$. In the tree on the left this subtree is reattached on the edge highlighted by a circle, connecting $l_5$ with its parent, and in the tree on the right the pruned subtree is reattached as child of a newly introduced root $\rho'$.

**Theorem 1.** *The decision problem* SPR*:*

*Instance: Two rooted trees $T$ and $R$ and an integer $k$*

*Question: Is $d_{\mathrm{SPR}}(T, R) \leq k$?*

*is $\mathcal{NP}$-complete.*

A proof for this theorem can be found in Bordewich and Semple [5]. This proof relies on the equivalence of the SPR distance between two trees and the size of a maximum agreement forest. A *maximum agreement forest* (MAF) of two rooted trees $T$ and $R$ can be interpreted as a forest that results from cutting the minimum number of edges from both $T$ and $R$ to result in the same forest, and its size $|\mathrm{MAF}(T, R)|$ is defined as the number of its connected components. For a formal definition, see [5]. The equality $|\mathrm{MAF}(T, R)| = d_{\mathrm{SPR}}(T, R)$ implies that the same edge is only pruned once on a shortest SPR path. Furthermore, distance computation can be broken down into smaller problems if two trees share some information in the form of common clusters [26], which we will formally define later.

## 2.1 Ranked Trees

A *ranked binary phylogenetic tree* is a pair $T = (T_u, \text{rank})$ consisting of a rooted tree $T_u$ and a function $rank : V \to \{0, 1, \ldots, n-1\}$, where $V$ is the set of nodes of $T_u$, such that: (i) $\text{rank}(v) = 0$ if and only if $v$ is a leaf, (ii) $\text{rank}(v) \neq rank(w)$ for all internal nodes $v \neq w$, and (iii) $\text{rank}(v) < \text{rank}(w)$ if $w$ is on the path from the root of $T_u$ to $v$. We refer to $\text{rank}(v)$ as the *rank of* $v$, and we denote node of rank $i$ in a ranked binary phylogenetic tree $T$ by $(T)_i$. For simplicity of notation, we say *tree* or *ranked tree* to refer to ranked binary phylogenetic trees, unless stated otherwise. The assumption that all leaves have rank 0 can be interpreted as requiring ranked trees to be ultrametric.

If $T = (T_u, \text{rank})$ is a tree, we call the rooted tree $T_u$ the *unranked version* of $T$, as it can be interpreted as $T$ without ranks, but with leaf labels. Two trees $T$ and $R$ are *identical* if there is a graph isomorphism between them that preserves leaf labels and ranks. We then write $T \simeq R$, and if the trees are not identical we write $T \not\simeq R$. An example of a tree $T$ with annotated ranks and its unranked version $T_u$ is given in Figure 2.

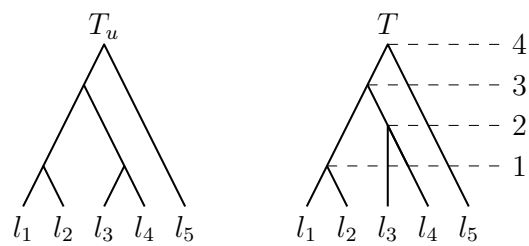


Figure 2: A rooted tree $T_u$ on the left and a ranked tree $T = (T_u, \text{rank})$ on the right.

A *subtree* $T|_v$ of a tree $T$ is a tree rooted in a node $v$ of $T$ that contains exactly the nodes that descend from $v$ in $T$, annotated by the same ranks as in $T$. We then say that $T|_v$ is *induced* by the node $v$. Note that by this definition of subtrees, the subtree of a ranked tree is not necessarily a ranked tree itself. We denote the parent of a node $v$ by $\text{parent}(v)$ and if a subtree $T|_v$ is induced by $v$, we call $\text{parent}(v)$ *parent of the subtree* $T|_v$. To emphasise that we are considering the parent of a subtree $T|_v$ in a tree $T$ we might write $\text{parent}_T(T|_v)$.

Since we consider trees where internal nodes are assigned unique ranks and leaves are assigned unique leaf labels, we can uniquely identify nodes in a tree by their labels, using the label function

$$l(v) = \begin{cases} \text{rank}(v) & \text{if } v \text{ is internal node} \\ \phi(v) & \text{if } v \text{ is leaf.} \end{cases} \tag{1}$$

Throughout this paper we assume $\{1, \ldots, n-1\} \cap \{l_1, \ldots, l_n\} = \emptyset$, which results in $l$ being a bijective function. We can therefore uniquely identify a node $v$ by its label $l(v)$, so we will refer to a node $v$ simply by $l(v)$. For example, we might refer to the internal node

with rank $i$ as *node* $i$ and to the leaf with label $l_j$ as *node* $l_j$ or *leaf* $l_j$. We define a strictly partial order $\prec_T$ on the co-domain of $l$ so that $l(u) \prec_T l(v)$ if $\text{rank}_T(u) < \text{rank}_T(v)$. If it is clear that we consider the tree $T$ we might simply write $\prec$ for $\prec_T$.

Using the label function (1) allows us to represent edges $(u, v)$ in a tree as $(l(u), l(v))$. We call the set $E(T) = \{(l(v), (l(w)) \mid (v, w) \text{ is an edge in the tree } T)\}$ the *edge set of* $T$. $E(T)$ uniquely defines $T$. We say that an edge $(l^+, l^-)$ *covers* a rank $i \in \{1, \ldots, n-1\}$ if $l^- \prec k \prec l^+$.

A *cluster* $C$ of a tree $T$ is the set of leaves descending from an internal node $v$ in $T$. We then say that the node $v$ *induces* the cluster $C$. A *cherry* is a subtree consisting of one internal node with two leaves as children, and we refer to a cherry by its cluster containing these two leaves, e.g. $\{c_1, c_2\}$. If the internal node of such a cherry has rank $i$, we say that the cherry $\{c_1, c_2\}$ has rank $i$. Given a set of leaves $S$ and a tree $T$, the subtree $T|_S$ of $T$ *induced by* $S$ is the subtree of $T$ with minimum number of leaves that contains all leaves of $S$. If $S$ is a cluster of $T$, $T|_S$ contains exactly the leaves of $S$.

We can uniquely represent a tree $T$ by a list of its clusters sorted according to increasing rank. This representation is called *cluster representation* and has been introduced by Collienne and Gavryushkin [10]. The leftmost tree in Figure 3 for example has cluster representation $[\{l_1, l_2\}, \{l_3, l_4\}, \{l_1, l_2, l_5\}, \{l_1, l_2, l_3, l_4, l_5\}]$. We include the cluster induced by the root in the cluster representation of a tree, even though this cluster is simply the set of all leaf labels $\{l_1, l_2, \ldots, l_n\}$ for all trees on $n$ leaves.

## 2.2 Subtree Prune and Regraft for Ranked Trees

A *horizontal* SPR *move* (HSPR move) on an edge $e$ of a tree $T$, which cannot be incident to the root, transforms this tree into a tree $T'$ by shifting the top node of $e$ horizontally to another branch. Formally, the move is performed in the following three steps:

(i) *prune the subtree* $T|_v$: delete the edge $e$ to obtain two subtrees $T|_v$ and $T_\rho$, where $T_\rho$ has the same root $\rho$ as $T$,

(ii) suppress the resulting node of degree 2 in $T_\rho$ (top node of edge $e$), whose rank we denote by $i$,

(iii) *reattach* $T|_v$ *on an edge* $f$: reattach the root of $T|_v$ to a newly introduced node $v$ at rank $i$ on an edge $f$ that covers rank $r$ in $T_\rho$.

Since the changes done to the tree $T$ move node $i$, we call this an HSPR *move at rank* $i$. A *rank move* on a tree $T$ can be applied to two nodes with consecutive ranks if they are not connected by an edge, and swaps the ranks of these two nodes. An example of an HSPR move and a rank move can be found in Figure 3.

Note that HSPR moves and rank moves are reversible and in contrast to SPR moves for rooted trees, HSPR moves do not allow subtrees to be reattached above the root of a tree.

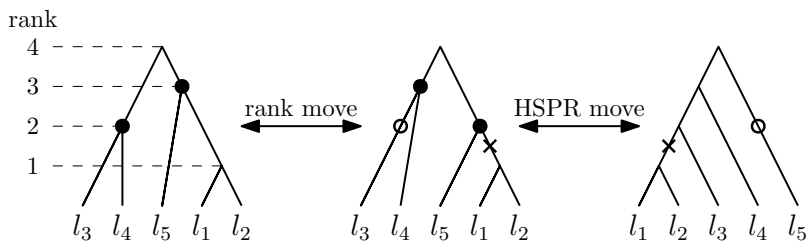


Figure 3: Rank move on the left, swapping the ranks of the nodes highlighted the leftmost tree. On the right an HSPR move at rank 2 is illustrated, moving the subtree induced by $\{l_1, l_2\}$ by cutting the edge highlighted by a cross and re-attaching it at the edge highlighted by a circle.

We are now ready to introduce our main objects of study in this paper, two treespaces extending the classical SPR treespace to ranked trees.

The Ranked SPR (RSPR) space is a graph where vertices represent trees on $n$ leaves that are connected by an edge if one tree can be transformed into the other by an HSPR move or a rank move. The Horizontal SPR (HSPR) space is a graph where vertices represent trees on $n$ leaves that are connected by an edge if one can be transformed into the other by an HSPR move.

We refer to the moves allowed in RSPR space, i.e. rank moves and HSPR moves, as RSPR *moves*.

A *path* between two trees in HSPR (RSPR) is a sequence of trees $p = [T_0, T_1, \ldots, T_d]$ such that $T_i$ and $T_{i+1}$ are connected by an HSPR (RSPR) move for all $i$. If a path $p$ contains $d+1$ trees, we say it has *length* $|p| = d$. A *shortest path* between trees $T$ and $R$ is a path of minimal length connecting $T$ and $R$. The length of such a path is called the *distance* between trees $T$ and $R$, and we refer to this distance as $d_{\text{HSPR}}(T, R)$ in HSPR space and $d_{\text{RSPR}}(T, R)$ in RSPR space.

### HSPR **moves on edge sets**

Let us consider how an HSPR move changes the set of edges of a tree. By the definition of HSPR moves, there are four edges involved in an HSPR move at rank $i$ on a tree $T$: The three edges incident to the node of rank $i$, and the edge $f$ on which the pruned subtree gets reattached. All other edges stay the same in the tree $T$ and its HSPR neighbour $T'$. Let $e = (i, l)$ be the edge that is cut by the HSPR move between $T$ and $T'$, let $(i^+, i)$ and $(i, i^-)$ be the other two edges incident to $i$, and let $f = (j^+, j^-)$ be the reattachment edge in $T$. Then the HSPR move between $T$ and $T'$ changes these edges as follows:

(i) *prune the subtree* $T|_v$: delete the edge $e = (i, l)$

(ii) suppress the resulting node of degree 2: replace edges $(i^+, i)$ and $(i, i^-)$ by an edge

$(i^+, i^-)$.

(iii) *reattach* $T|_v$: replace edge $f = (j^+, j^-)$ by $(j^+, i)$ and $(i, j^-)$ and add edge $(i, l)$.

An illustration of the trees $T$ and $T'$ with these node labels is provided in Figure 4. The difference between the edge sets of $T$ and $T'$ can be summarised to

$$\begin{aligned} E(T') = E(T) &\setminus \{(i^+, i), (i, i^-), (j^+, j^-)\} \\ &\cup \{(i^+, i^-), (j^+, i), (i, j^-)\}. \end{aligned}$$

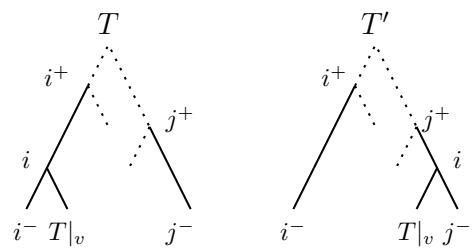


Figure 4: HSPR move pruning the subtree $T|_v$, moving it to the edge $(i^+, i^-)$. Dotted edges represent parts of the tree that potentially contain further nodes.

Conversely, if an edge set $E(T')$ can be described in this way, then $T$ and $T'$ are connected by an HSPR move:

**Theorem 2.** *Let $E(T)$ be the set of edges of a tree $T$. A tree $T'$ with*

$$\begin{aligned} E(T') = E(T) &\setminus \{(i^+, i), (i, i^-), (j^+, j^-)\} \\ &\cup \{(i^+, i^-), (j^+, i), (i, j^-)\}. \end{aligned}$$

*is* HSPR *neighbour of $T$ if and only if $(i^+, i), (i, i^-), (j^+, j^-) \in E(T)$ and $j^- \prec i \prec j^+$.*

Note that $j^- \prec i \prec j^+$ is required in Theorem 2, as the reattachment edge needs to cover the rank $i$ of the HSPR move.

Using Theorem 2, we can define an HSPR move between trees $T$ and $T'$ by providing the difference in edge sets $E(T)$ and $E(T')$. We say that a change in edge sets describes a *valid* HSPR move, if the edges that are removed from $E(T)$ fulfil the conditions of Theorem 2.

### HSPR **moves on cluster representation**

Similar to using edge sets to describe HSPR moves, we can also use the cluster representation to describe an HSPR move between two trees. Let $T$ and $R$ be trees connected by an HSPR move at rank $i$ as described in Theorem 2 and let the clusters induced by $i^-$, $T|_v$ (the subtree that is moved), and $j^-$ be $A$, $B$, and $C$, respectively. Any of $A, B$, and $C$

could be just a leaf, e.g. $A = \{l_k\}$ for some $k$, but for simplicity we refer to those sets as clusters, too. Let furthermore $T = [C_1, C_2, \ldots, C_{i-1}, A \cup B, C_{i+1}, \ldots, C_{n-1}]$ be the cluster representation of $T$. The HSPR move described in Theorem 2 then creates a new cluster $B \cup C$ at rank $i$, as in $T'$ the node $i$ has $T|_v$ and $j^-$ as children. All clusters induced by nodes with rank less than $i$ remain unchanged between $T$ and $T'$. Since the subtree induced by $B$ becomes sibling of the subtree induced by $C$ in $T'$, the move between $T$ and $T'$ removes $B$ from every cluster of $T$ that contains $B$ but not $C$ and is induced by a node with rank greater than $i$. On the other side, $B$ is added to every cluster induced by a node with rank greater than $i$ that contains $C$ in $T$. All remaining clusters induced by nodes with rank greater than $i$ that do not contain $B$ or $C$ remain unchanged between $T$ and $T'$.

We can summarise this to describe $T'$ by its cluster representation

$$T' = [C_1, \ldots, C_{i-1}, B \cup C, C'_{i+1}, \ldots, C'_{n-1}]$$

with

$$C'_m = \begin{cases} C_m \setminus B & \text{if } B \subset C_m \text{ and } C \not\subset C_m \\ C_m \cup B & \text{if } C \subset C_m \\ C_m & \text{otherwise.} \end{cases}$$

Conversely, if the difference between the cluster representation of $T$ and $T'$ can be described in this way, $T$ and $T'$ are connected by an HSPR move:

**Theorem 3.** *Let $T = [C_1, C_2, \ldots, C_{i-1}, A \cup B, C_{i+1}, \ldots, C_{n-1}]$ be a tree and $C = C_l$ for some $l \in \{1, \ldots, i-1\}$. A tree*

$$T' = [C_1, \ldots, C_{i-1}, B \cup C, C'_{i+1}, \ldots, C'_{n-1}]$$

*is connected to $T$ by an* HSPR *move if and only if*

$$C'_m = \begin{cases} C_m \setminus B & \textit{if } B \subset C_m \textit{ and } C \not\subset C_m \\ C_m \cup B & \textit{if } C \subset C_m \\ C_m & \textit{otherwise.} \end{cases}$$

## 3 Basic Properties of HSPR and RSPR

The first question we want to answer is whether our newly defined treespaces are connected (Theorem 4), that is, whether a tree can be transformed into any other tree by a sequence of moves in HSPR or RSPR. Connectedness is essential for these treespaces, as tree rearrangements are used for tree proposals in (MCMC) random walks, which should be able to reach any tree in treespace from any starting tree. For developing and interpreting such random walks it is furthermore important to know how many trees have distance one from a given tree, as well as the maximum distance between any two trees. We establish

neighbourhood size (Theorem 5) and maximum distance (diameter) in RSPR and HSPR in Section 3.1. We then investigate the cluster property for RSPR and HSPR and explain the significance of neither of the two spaces having this property (Section 3.2).

**Theorem 4.** *The treespaces* HSPR *and* RSPR *are connected.*

*Proof.* We show that any pair of trees $T'$ and $R$ are connected by a path of only HSPR moves. As all HSPR moves are RSPR moves, it then follows that both spaces are connected.

We construct an HSPR path from $T'$ to $R$ by the following bottom up approach, iterating through ranks $k = 1, \ldots, n-1$ of $R$. In every iteration $k$, we perform HSPR moves so that all nodes with ranks less than or equal to $k$ induce the same clusters in the tree after iteration $k$ and $R$. Let $T$ be the tree before iteration $k$, let $R|_i$ and $R|_j$ be the subtrees that are children of the node of rank $k$ in $R$, and let $T|_l$ and $T|_m$ be the subtrees that are children of the node of rank $k$ in $T$, i.e. $\text{parent}_R(R|_i) = \text{parent}_R(R|_j) = k$ and $\text{parent}_T(T|_l) = \text{parent}_T(T|_m) = k$. We can assume that $R|_i$ and $R|_j$ are subtrees in $T$, because we use a bottom up approach that results in all nodes of rank less than $k$ inducing the same cluster in $R$ and the tree $T$ before iteration $k$. Note that in the first iteration $k = 1$, $R|_i$ and $R|_j$ will contain a single leaf only. Consider the following path transforming $T$ into a tree $T_2$ with $\text{parent}_{T_2}(R|_i) = \text{parent}_{T_2}(R|_j) = k$ (see Figure 5):

First, perform an HSPR move at rank $k$ that prunes $T|_l$ from $T$ and reattaches it on the edge between $R|_i$ and $\text{parent}_T(R|_i)$. In the resulting tree $T_1$, $\text{parent}_{T_1}(R|_i) = \text{parent}_{T_1}(T|_l) = k$. In a second step, we perform an HSPR move pruning the subtree $R|_i$ from $T_1$ and reattaching it on the edge between $R|_j$ and $\text{parent}_{T_1}(R|_j)$, resulting in a tree $T_2$ with $\text{parent}_{T_2}(R|_i) = \text{parent}_{T_2}(R|_j) = k$. Since all moves between $T$ and $T_2$ are HSPR moves at rank $k$, no clusters induced by nodes less than $k$ change between $T$ and $T_2$, while the cluster induced by the node of rank $k$ changes so that it coincides between $T_2$ and $R$. Hence, all cluster induced by nodes with rank less than or equal to $k$ are identical in $T_2$ and $R$. Note that if $\{R|_i, R|_j\}$ and $\{T|_l, T|_m\}$ intersect, the trees $T_1$ or $T_2$ might be equal to $T$, and fewer than two steps are required to reach $T_2$. At the end of iteration $k$, we update $T := T_2$ and continue with the next iteration $k+1$. After iteration $n-2$, we reach $R$, as after each iteration $k$ all nodes with rank less than or equal to $k$ induce the same clusters in $T$ and $R$.

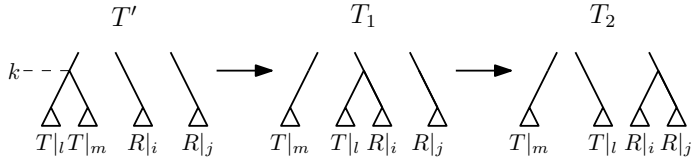


Figure 5: Path between $T$ and $T_2$ as described in the proof of Theorem 4. Only subtrees involved in the moves given in the theorem are shown in this illustration.

This procedure can be applied to any two trees to compute a path connecting them by

a sequence of HSPR moves, which proves the theorem. □

The algorithm used to prove Theorem 4 produces a path between any two trees in HSPR, and can hence be used to approximate HSPR distances. An implementation can be found on GitHub [8].

Since HSPR and RSPR are connected undirected graphs, we obtain the following corollary.

**Corollary 1.** *$d_{\text{HSPR}}$ and $d_{\text{RSPR}}$ are metrics.*

The *(1-)neighbourhood* of a tree $T$ in a treespace with distance measure $d$ is defined as $NH(T) := \{T' \mid d(T, T') = 1\}$ and a tree $T' \in NH(T)$ is called *neighbour* of $T$. We use $NH_{\text{HSPR}}(T)$ and $NH_{\text{RSPR}}(T)$ to refer to the neighbourhood of $T$ in HSPR and RSPR, respectively. Because tree inference algorithms often require sampling tree neighbourhoods, it is important to know the number of *neighbours* of a tree $T$ under a tree rearrangement, i.e. $|NH(T)|$. In rooted and unrooted SPR the number of neighbours of a tree is quadratic in the number of leaves $n$ [2, 31].

For counting the number of neighbours of a tree in RSPR, we need the following notion: If a tree $T$ has two nodes $r$ and $r+1$ with rank difference one that are not connected by an edge, we say that $[r, r+1]$ is a *rank interval*. The leftmost tree $T$ in Figure 3 for example has two rank intervals: $[3, 2]$ and $[2, 1]$. We now show that the number of neighbours in RSPR and HSPR is quadratic in the number of leaves, with the number of neighbours in RSPR naturally depending on the shape of the tree. We derive both of these numbers explicitly.

**Theorem 5.** *The number of neighbours of a tree $T$ with $k$ rank intervals is $|NH_{\text{RSPR}}(T)| = (n-1)(n-2) + k$ in* RSPR *and $|NH_{\text{HSPR}}(T)| = (n-1)(n-2)$ in* HSPR.

*Proof.* The number of neighbouring trees resulting from a rank move on a tree with $k$ rank intervals is $k$, as there is one unique rank move for every such interval.

We now count the number of HSPR moves at rank $i$ for $i \in \{1, \ldots, n-1\}$. Since every node has two children, two different subtrees can be pruned by an HSPR move at rank $i$. There are $n-1-i$ edges that cover rank $i$, excluding the one on which the node of rank $i$ is placed in $T$, so there are $n-1-i$ potential reattachment edges for the pruned subtree. This gives $2(n-1-i)$ RSPR neighbours of $T$ resulting from an HSPR move at rank $i$. And because HSPR moves can be performed at any rank between 1 and $n-1$, the number of RSPR moves possible on $T$ is:

$$2\sum_{i=1}^{n-1}(n-1-i) = (n-1)(n-2)$$

Since all rank moves and HSPR moves result in different trees, $T$ has $(n-1)(n-2)$ neighbours in HSPR and $(n-1)(n-2)+k$ neighbours in RSPR. □

### 3.1 Diameter

The maximum distance between any two trees in a treespace with distance measure $d$, $\max_{T,R} d(T,R)$, is called the *diameter* of the treespace. When measuring the similarity of trees using a distance metric, knowing the maximum possible distance is essential for interpreting distances. The diameter of both unrooted and rooted SPR space is $n - \Theta(\sqrt{n})$ [4, 12] and hence linear in $n$. We show in this section that the diameters of HSPR space and RSPR space are linear in the number of leaves $n$, too, by establishing lower and upper bounds.

**Corollary 2.** *The diameters of* HSPR *space and* RSPR *space have an upper bound of* $2(n-2)$.

*Proof.* We can use the algorithm introduced in the proof of Theorem 4 to compute a path between any two trees $T$ and $R$ in HSPR. Since every HSPR move is an RSPR move, the length of this path is an upper bound of the diameter of both HSPR and RSPR space. The algorithm uses a bottom-up approach that, starting at tree $T$, constructs in iteration $k = 1, \ldots, n-2$ the cluster induced by the node of rank $k$ in $R$, using at most two HSPR moves. After iteration $n-2$, we receive the destination tree $R$ after at most $2(n-2)$ HSPR moves. Because the path $p$ constructed by this algorithm has length at most $2(n-2)$, this provides an upper bound to the diameter of HSPR and RSPR. □

It is important to note that the algorithm described in the proof of Theorem 4 approximates HSPR distances and does not compute the exact distance for all pairs of trees, which we can show using our implementations [8].

We can also prove a lower bound for the distance between any two trees in HSPR, but first we need the following lemma.

**Lemma 1.** *Let $T$ and $R$ be trees containing $x$ leaves whose parents have different ranks in $T$ and $R$, i.e. $|\{(u,v) \mid v \text{ is leaf and } (u,v) \in E(T) \text{ and } (u,v) \notin E(R)\}| = x$. Then $d_{\text{HSPR}}(T,R) \geq \lceil \frac{x}{2} \rceil$.*

*Proof.* As described in the technical introduction, an HSPR move at rank $i$ between trees $T$ and $T'$ leads to the following difference in edge sets for some edges $(i^+, i), (i, i^-), (j^+, j^-) \in E(T)$ where $(i^+, j^-)$ covers rank $i$:

$$E(T') = E(T) \setminus \{(i^+, i), (i, i^-), (j^+, j^-)\} \cup \{(i^+, i^-), (j^+, i), (i, j^-)\}.$$

Therefore, the only nodes whose parents change by this HSPR move are the nodes $i$, $i^-$, and $j^-$. Not all of them need to have different a parent after the move, since for example $i^+ = j^+$ results in the parent of $i$ having the same rank in $T$ and $T'$. With $(i^+, i)$ and $(i, i^-)$ being edges in $T$, it follows that $i$ is an internal node, so only $i^-$ and $j^-$ can be leaves. Therefore, an HSPR move can change the parents of at most two leaves.

Since the ranks of $x$ parents of leaves differ between $T$ and $R$ and any HSPR move can fix at most two of those, there are in total at least $\lceil \frac{x}{2} \rceil$ HSPR moves needed to connect $T$ and $R$. □

The lower bound given in Lemma 1 can be tight. For example, the two leftmost trees in Figure 6 are connected by one HSPR move and the parents of $x = 2$ leaves ($l_2$ and $l_5$) have different ranks in the two trees.

**Theorem 6.** *There are trees $T$ and $R$ with distance $d_{\text{HSPR}}(T, R) \geq \lceil \frac{n}{2} \rceil$.*

*Proof.* Let $T$ and $R$ be the following caterpillar trees:

$$T = [\{l_1, l_2\}, \{l_1, l_2, l_3\}, \{l_1, l_2, l_3, l_4\}, \ldots, \{l_1, l_2, \ldots, l_n\}]$$
$$R = [\{l_3, l_4\}, \{l_2, l_3, l_4\}, \{l_2, l_3, l_4, l_5\}, \ldots, \{l_2, \ldots, l_n\}, \{l_1, \ldots, l_n\}]$$

The parent of $l_1$ and $l_2$ has rank one in $T$, but not in $R$, where $l_3$ and $l_4$ are children of the node of rank one. For all other leaves $l_i$ with $i \in \{5, \ldots, n\}$, the rank of the parent also is different in $T$ than in $R$: $\text{parent}_T(l_i) = i - 1 \neq i - 2 = \text{parent}_R(l_i)$. Therefore, the parents of all $n$ leaves have different ranks in $T$ and $R$, and by Lemma 1 it follows $d_{\text{HSPR}}(T, R) \geq \lceil \frac{n}{2} \rceil$. □

By Corollary 2 and Theorem 6, the diameter of the HSPR space is linear in $n$. We will see later (Corollary 6) that Theorem 6 also applies to RSPR.

### 3.2 No Cluster property

Two trees $T$ and $R$ *share* a cluster $C$ if both of them contain $C$ as cluster. If on a path $p$ every tree contains the cluster $C$, we say that $p$ *preserves* the cluster $C$. We moreover say that a treespace has the *strong cluster property* (simply called *cluster property* by Collienne et al. [9]) if for two trees sharing a cluster $C$, every shortest path between them preserves $C$. In other words, if two trees share some evolutionary information in form of a cluster, this information is preserved along every shortest path between them if the treespace has the cluster property. If for any two trees sharing a cluster $C$ there exists a shortest path that preserves $C$ in a tree space, we say that this treespace has the *weak cluster property*. Note that the difference between the weak and the strong cluster property is that for the strong cluster property we require all shortest paths to preserve clusters, while for the weak cluster property only one shortest path needs to preserve a shared cluster.

The classic rooted (unranked) version of SPR space has the weak cluster property, which is shown as part of the proof of Theorem 2.2 in [26], where the problem of computing the SPR distance is split into the problem of computing distances for subtrees induced by shared clusters. The weak cluster property of SPR is essential for fixed-parameter tractable algorithms [26, 35] and facilitates proofs for $\mathcal{NP}$-hardness of computing SPR distances, as it is related to the formulation of this problem as an agreement forest problem.

Here we show that RSPR and HSPR space have neither the weak nor the strong cluster property. This observation is important as it suggests that the proving technique for $\mathcal{NP}$-hardness for rooted (unranked) SPR, which uses maximum agreement forests, cannot be used for ranked trees.

**Theorem 7.** HSPR *space and* RSPR *space do not have the weak cluster property.*

*Proof.* We prove this theorem by considering trees $T$ and $R$ that share a cluster but have no shortest path between them that preserves this cluster. Let $T$ and $R$ be the following two trees (see Figure 6):

$$T = [\{l_1, l_2\}, \{l_1, l_2, l_3\}, \{l_4, l_5\}]$$
$$R = [\{l_4, l_5\}, \{l_2, l_3\}, \{l_1, l_4, l_5\}]$$

On any path between $T$ and $R$ preserving the shared cluster $\{l_4, l_5\}$, the rank of the node inducing this cluster needs to decrease from 3 in $T$ to 1 in $R$. Note that originating from $T$, no HSPR move can decrease the rank of this cluster, the only possible moves preserving $\{l_4, l_5\}$ are rank moves. There is hence no shortest path between $T$ and $R$ in HSPR that preserves $\{l_4, l_5\}$, i.e. HSPR does not have the weak cluster property.

In RSPR, two rank moves are necessary to decrease the rank of the node inducing $\{l_4, l_5\}$ to 1, resulting in a tree $R' = [\{l_4, l_5\}, \{l_1, l_2\}, \{l_1, l_2, l_3\}]$. Since $R'$ is not identical to $R$, further RSPR moves are needed to receive $R$, resulting in a path of length greater than two.

There is however a path from $T$ and $R$ with only two HSPR moves, where first the leaf $l_1$ is pruned and moved to the edge $(\text{parent}(l_5), l_5)$ and then $l_5$ is pruned and moved to the edge $(\text{parent}(l_4), l_4)$ (see Figure 6). Since this path is shorter than any path in RSPR preserving the cluster $\{l_4, l_5\}$, RSPR space does not have the weak cluster property.

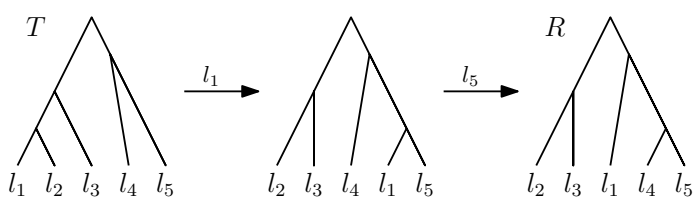


Figure 6: Trees $T$ and $R$ of the counterexample to the weak cluster property in RSPR and HSPR in Theorem 7. The labels of the arrows indicate leaves that are pruned in the corresponding HSPR moves.

□

# 4 Shortest paths

One of the main hindrances of using tree rearrangement based distance measures is that most of them are $\mathcal{NP}$-hard to compute. This applies to rooted and unrooted SPR distance [5, 19] as well as NNI distance [11]. A recent modification of NNI tree rearrangements for ranked trees has however shown to be computationally tractable [10]. Whether the same is true for ranked SPR is a natural question, and we approach it by investigating the structure of shortest paths in RSPR and HSPR treespace. Though the complexity of computing distances in HSPR and RSPR remains unknown, our results in this section are a first step towards understanding shortest paths in these treespaces. In particular, we investigate the relationship of shortest paths in HSPR and RSPR, which hints at the complexity of the shortest path problem in the two treespaces being identical. Gaining insights into the structure of shortest paths might furthermore be useful for developing algorithms for computing or approximating ranked SPR distances.

## 4.1 HSPR Shortest Paths

Shortest paths between trees in HSPR are generally not unique, and we show here that they can be arranged so that the rank at which subtrees are cut does not decrease on a shortest path (Theorem 8). We then use this result to study scenarios at which clusters are preserved along shortest paths: First, we show that if the cherry at rank one is identical in two trees, it is preserved on every shortest path (Theorem 9). Then, by further generalising this result, we show in Corollary 4 that for two trees with identical clusters up to a certain rank, all shortest paths between them preserve these clusters. These results provide insights into the shape of shortest paths in HSPR, and in future research we hope to leverage these results to prove the complexity of computing distances in this treespace.

To prove that there is a shortest path in HSPR on which ranks of HSPR moves do not decrease, we need the following lemma.

**Lemma 2.** *Let $p = [T, T', R]$ be a path in* HSPR *such that $T$ and $T'$ are connected by an* HSPR *move at rank $k$ and $T'$ and $R$ are connected by an* HSPR *move at rank $i$ with $k > i$. Then there is a path $p' = [T, T'', R]$ where an* HSPR *move at rank $i$ connects $T$ and $T''$ and an* HSPR *move at rank $k$ connects $T''$ and $R$.*

*Proof.* As described in Theorem 2, we can describe the HSPR move between trees $T$ and $T'$ by the change in the set of edges $E(T)$, compared to $E(T')$:

$$E(T') = E(T) \setminus \{(k^+, k), (k, k^-), (l^+, l^-)\} \cup \{(k^+, k^-), (l^+, k), (k, l^-)\}$$

where $(k^+, k), (k, k^-), (l^+, l^-) \in E(T)$ and $(l^+, l^-)$ covers rank $k$. Similarly, we assume that $E(T')$ and $E(R)$ are related in the following way:

$$E(R) = E(T') \setminus \{(i^+, i), (i, i^-), (j^+, j^-)\} \cup \{(i^+, i^-), (j^+, i), (i, j^-)\}$$

where $(i^+, i), (i, i^-), (j^+, j^-) \in E(T')$ and $(j^+, j^-)$ covers rank $i$. In total, we get

$$\begin{aligned} E(R) &= E(T) \cup \big(\{(k^+, k^-), (l^+, k), (k, l^-)\} \cup \{(i^+, i^-), (j^+, i), (i, j^-)\}\big) \\ &= \setminus\big(\{(k^+, k), (k, k^-), (l^+, l^-)\} \cup \{(i^+, i), (i, i^-), (j^+, j^-)\}\big). \end{aligned}$$

Note that there might be edges that are added to $T'$ and then deleted in $R$. Therefore, when summarising multiple HSPR moves as set operations, we assume that we consider multisets, even though the edge set of a tree does not contain an edge multiple times. Let

$$\begin{aligned} &E_1 = \{(k^+, k), (k, k^-), (l^+, l^-)\}, E_2 = \{(k^+, k^-), (l^+, k), (k, l^-)\} \\ &E_3 = \{(i^+, i), (i, i^-), (j^+, j^-)\}, \text{ and } E_4 = \{(i^+, i^-), (j^+, i), (i, j^-)\}. \end{aligned}$$

Then $E_3 \cap E_1 = \emptyset$, as otherwise the HSPR move between $T'$ and $R$ would not be possible.

If $E_2 \cap E_3 = \emptyset$, then $E_3 \subset E(T)$, and we can create a tree $T''$ with $E(T'') = E(T) \setminus E_3 \cup E_4$, i.e. $T$ and $T''$ are connected by an HSPR move at rank $i$. With $E_2 \cap E_3 = \emptyset$ we get $E_2 \in E(T'')$ and we can perform an HSPR move at rank $k$ on $T''$ that results in $E(R^*) = E(T'') \setminus E_2 \cup E_1$.

Hence, $E(R^*) = E(T) \cup (E_3 \cup E_1) \setminus (E_4 \cup E_2) = E(R)$. Therefore, $R^*$ and $R$ are identical and the path $p' = [T, T'', R]$ contains an HSPR move at rank $i$ followed by an HSPR move at rank $k$.

We now distinguish different cases in which $E_2 \cap E_3 \neq \emptyset$. With $i^-, j^- \prec i \prec k \prec k^+$, it is $(i, i^-) \notin E_2$ and $(l^+, k) \notin E_3$. Therefore, $E_2 \cap E_3 \neq \emptyset$ if and only if $\{(i^+, i), (j^+, j^-)\} \cap \{(k^+, k^-), (k, l^-)\} \neq \emptyset$. We distinguish six different cases, based on this intersection. In every case, we construct an alternative path $p' = [T, T'', R]$, where $T$ and $T''$ are connected by an HSPR move at rank $i$ and $T''$ and $R$ are connected by an HSPR move at rank $k$. To show that the moves that we describe by edge set changes are valid HSPR moves, we need to show that the edge set changes we provide fulfil the criteria listed in Theorem 2. By our assumption of the moves on $p$, we know that $E_1 \subset E(T)$ and if $e \in E_3 \setminus E_2$, then $e \in E(T)$.

1. $(i^+, i) = (k^+, k^-)$ and $(j^+, j^-) \neq (k, l^-)$: We perform an HSPR move on $T'$ to receive the tree $T''$ with

   $$E(T'') = E(T) \setminus \{(k, i), (i, i^-), (j^+, j^-)\} \cup \{(k, i^-), (j^+, i), (i, j^-)\}.$$

   This is a valid move because:

   - $(k, i) \in E(T)$, because $(k, i) = (k, k^-)$ and $(k, k^-) \in E_1$.
   - $(i, i^-) \in E(T)$, because $(i, i^-) \in E_3$ and $(i, i^-) \notin E_2$.
   - $(j^+, j^-) \in E(T)$, because $(j^+, j^-) \in E_3$, and $(j^+, j^-) \notin E_2$
   - $(j^+, j^-)$ covers rank $i$ by the definition of the moves on $p$.

We can then receive a tree $R^*$ by an HSPR move on $T''$ with

$$E(R^*) = E(T'') \setminus \{(k^+, k), (k, i^-), (l^+, l^-)\} \cup \{(k^+, i^-), (l^+, k), (k, l^-)\}.$$

This is a valid move because:

- $(k^+, k) \in E(T'')$, because $(k^+, k) \in E_1$ and $(k^+, k)$ has not been removed between $T$ and $T''$.
- $(k, i^-) \in E(T'')$, as it has been added between $T$ and $T''$.
- $(l^+, l^-) \in E(T'')$, because $(l^+, l^-) \in E_1$, and $(l^+, l^-)$ has not been removed between $T$ and $T''$.
- $(l^+, l^-)$ covers rank $k$ by the definition of the moves on $p$.

With $i^+ = k^+$ and $i = k^-$ it follows $E(R^*) = E(R)$, and hence $R^* \simeq R$.

2. $(i^+, i) = (k^+, k^-)$ and $(j^+, j^-) = (k, l^-)$:

   Let $T''$ be resulting from $T$ by an HSPR move at rank $i$ that changes $E(T)$ as follows:

$$E(T'') = E(T) \setminus \{(k, i), (i, i^-), (l^+, l^-)\} \cup \{(k, i^-), (l^+, i), (i, l^-)\}.$$

   This is a valid move because:

   - $(k, i) \in E(T)$, because $(k, i) = (k, k^-)$ and $(k, k^-) \in E_1$.
   - $(i, i^-) \in E(T)$, because $(i, i^-) = (k^-, i^-) \in E_3$ and $(k^-, i^-) \notin E_2$.
   - $(l^+, l^-) \in E(T)$, because $(l^+, l^-) \in E_1$.
   - $(j^+, j^-)$ covers rank $i$ by the definition of the moves on $p$.

   We can then receive a tree $R^*$ by an HSPR move on $T''$ with

$$E(R^*) = E(T'') \setminus \{(k^+, k), (k, i^-), (l^+, i)\} \cup \{(k^+, i^-), (l^+, k), (k, i)\}.$$

   This is a valid move because:

   - $(k^+, k) \in E(T'')$, because $(k^+, k) \in E_1$ and $(k^+, k)$ has not been removed between $T$ and $T''$.
   - $(k, i^-) \in E(T'')$, as it has been added between $T$ and $T''$.
   - $(l^+, i) \in E(T'')$, as it has been added between $T$ and $T''$.
   - $(l^+, i)$ covers rank $k$ by the definition of the moves on $p$.

   With $i^+ = k^+$, $i = k^-$, $j^+ = k$, and $j^- = l^-$ it follows $E(R^*) = E(R)$ and hence $R^* \simeq R$.

3. $(i^+, i) = (k, l^-)$ and $(j^+, j^-) \neq (k^+, k^-)$:

   We can perform an HSPR move at rank $i$ on $T$ that gives us the following tree $T''$:

$$E(T'') = E(T) \setminus \{(l^+, i), (i, i^-), (j^+, j^-)\} \cup \{(l^+, i^-), (j^+, i), (i, j^-)\}.$$

   This is a valid move because:

   - $(l^+, i) \in E(T)$, because $(l^+, i) = (l^+, l^-) \in E_1$.
   - $(i, i^-) \in E(T)$, because $(i, i^-) \in E_3$ and $(i, i^-) \notin E_2$.
   - $(j^+, j^-) \in E(T)$, because $(j^+, j^-) \in E_3$ and $(j^+, j^-) \notin E_2$.
   - $(j^+, j^-)$ covers rank $i$ by the definition of the moves on $p$.

   We can then perform an HSPR move at rank $k$ on $T''$ to get a tree $R^*$ with

$$E(R^*) = E(T'') \setminus \{(k^+, k), (k, k^-), (l^+, i^-)\} \cup \{(k^+, k^-), (l^+, k), (k, i^-)\}.$$

   This is a valid move because:

   - $(k^+, k) \in E(T'')$, because $(k^+, k) \in E_1$ and $(k^+, k)$ has not been removed between $T$ and $T''$.
   - $(k, k^-) \in E(T'')$, because $(k^+, k) \in E_1$ and $(k^+, k)$ has not been removed between $T$ and $T''$.
   - $(l^+, i^-) \in E(T'')$, as it has been added between $T$ and $T''$.
   - $(j^+, j^-)$ covers rank $k$ by the definition of the moves on $p$.

   By $i^+ = k$ and $i = l^-$ it then follows $E(R^*) = E(R)$ and hence $R^* \simeq R$.

4. $(i^+, i) = (k, l^-)$ and $(j^+, j^-) = (k^+, k^-)$:

   We can perform an HSPR move at rank $i$ on $T$ to get a tree $T''$ with

$$E(T'') = E(T) \setminus \{(l^+, i), (i, i^-), (k, j^-)\} \cup \{(l^+, i^-), (k, i), (i, j^-)\}.$$

   This is a valid move because:

   - $(l^+, i) \in E(T)$, because $(l^+, i) = (l^+, l^-) \in E_1$.
   - $(i, i^-) \in E(T)$, because $(i, i^-) \in E_3$ and $(i, i^-) \notin E_2$.
   - $(k, j^-) \in E(T)$, because $(k, j^-) = (k, k^-) \in E_1$.
   - $(k, j^-)$ covers rank $i$ by the definition of the moves on $p$.

   An HSPR move at rank $i$ can transform $T''$ to a tree $R^*$ with

$$E(R^*) = E(T'') \setminus \{(k^+, k), (k, i), (l^+, i^-)\} \cup \{(k^+, i), (l^+, k), (k, i^-)\}.$$

   This is a valid move because:

- $(k^+, k) \in E(T'')$, because $(k^+, k) \in E_1$ and $(k^+, k)$ has not been removed between $T$ and $T''$.
- $(k, i) \in E(T'')$, as it has been added between $T$ and $T''$.
- $(l^+, i^-) \in E(T'')$, as it has been added between $T$ and $T''$.
- $(l^+, i^-)$ covers rank $k$ by the definition of the moves on $p$.

By $i^+ = k$, $i = l^-$, $j^+ = k^+$, and $j^- = k^-$ it follows $E(R^*) = E(R)$ and therefore $R^* \simeq R$.

5. $(j^+, j^-) = (k^+, k^-)$ and $(i^+, i) \neq (k, l^-)$:

   Performing an HSPR move at rank $i$ on $T$ can give us a tree $T''$ with

$$E(T'') = E(T) \setminus \{(i^+, i), (i, i^-), (k, j^-)\} \cup \{(i^+, i^-), (k, i), (i, j^-)\}.$$

   This is a valid move because:

   - $(i^+, i) \in E(T)$, because $(i^+, i) \in E_3$ and $(i^+, i) \notin E_2$.
   - $(i, i^-) \in E(T)$, because $(i, i^-) \in E_3$ and $(i, i^-) \notin E_2$.
   - $(k, j^-) \in E(T)$, because $(k, j^-) = (k, k^-) \in E_1$.
   - $(k, j^-)$ covers rank $i$ by the definition of the moves on $p$.

   An HSPR move at rank $k$ can therefore convert $T''$ into a tree $R^*$ with

$$E(R^*) = E(T'') \setminus \{(k^+, k), (k, i), (l^+, l^-)\} \cup \{(k^+, i), (l^+, k), (k, l^-)\}.$$

   This is a valid move because:

   - $(k^+, k) \in E(T'')$, because $(k^+, k) \in E_1$ and $(k^+, k)$ has not been removed between $T$ and $T''$.
   - $(k, i) \in E(T'')$, as it has been added between $T$ and $T''$.
   - $(l^+, l^-) \in E(T'')$, because $(l^+, l^-) \in E_1$ and $(l^+, l^-)$ has not been removed from $T$ to get $T''$.
   - $(l^+, l^-)$ covers rank $k$ by the definition of the moves on $p$.

   By $j^+ = k^+$ and $j^- = k^-$ it follows $E(R^*) = E(R)$ and hence $R^* \simeq R$.

6. $(j^+, j^-) = (k, l^-)$ and $(i^+, i) \neq (k^+, k^-)$:

   We can perform an HSPR move at rank $i$ on $T$ to get a tree $T''$ with edge set

$$E(T'') = E(T) \setminus \{(i^+, i), (i, i^-), (l^+, l^-)\} \cup \{(i^+, i^-), (l^+, i), (i, l^-)\}.$$

   This is a valid move because:

- $(i^+, i) \in E(T)$, because $(i^+, i) \in E_3$ and $(i^+, i) \notin E_2$.
- $(i, i^-) \in E(T)$, because $(i, i^-) \in E_3$ and $(i, i^-) \notin E_2$.
- $(l^+, l^-) \in E(T)$, because $(l^+, l^-) \in E_1$.
- $(l^+, l^-)$ covers rank $i$ because $i \prec k \prec l^+$ and $j^- = l^- \prec i$.

Then an HSPR move at rank $k$ is possible on $T''$ and transforms this tree into $R^*$ with

$$E(R^*) = E(R) \setminus \{(k^+, k), (k, k^-), (l^+, i)\} \cup \{(k^+, k^-), (l^+, k), (k, i)\}.$$

This is a valid move because:

- $(k^+, k) \in E(T'')$, because $(k^+, k) \in E_1$ and $(k^+, k)$ has not been removed between $T$ and $T''$.
- $(k, k^-) \in E(T'')$, as $(k, k^-) \in E_1$ and $(k, k^-)$ has not been removed between $T$ and $T''$.
- $(l^+, i) \in E(T'')$, because it has been added by the move between $T$ and $T''$.
- $(l^+, i)$ covers rank $k$ by the definition of the moves on $p$.

By $j^+ = k$ and $j^- = l^-$, it follows $E(R*) = E(R)$, hence $R^* \simeq R$.

□

**Theorem 8.** *There is always a shortest* HSPR *path between any two trees $T$ and $R$ on which the rank of* HSPR *moves increases monotonically.*

*Proof.* Let $p$ be a shortest path between trees $T$ and $R$. By Lemma 2 we can take any pair of consecutive HSPR moves at ranks $k$ and $i$ and if $k > i$, we can replace them by two HSPR moves so that first a rank $i$ and then a rank $k$ HSPR move is performed. By replacing all such pairs of HSPR moves iteratively, we receive a path on which the ranks of HSPR moves increase monotonically. □

To show that all shortest paths preserve a shared cherry at rank one (Theorem 9), we need the following two lemmas, which describe HSPR moves at a fixed rank $k$ along a shortest path. These lemmas are interesting on their own as they are informative about the local geometry of the HSPR treespace. Recall that $(T)_i$ is the node of rank $i$ in $T$ whereas $T|_i$ is the subtree induced by that node.

**Lemma 3.** *Let $T$ and $R$ be trees containing subtrees $T|_1, T|_2, \ldots, T|_d$ with $T|_j \not\simeq T|_m$ for all $j \neq m$, so that $(T)_{i_j}$ and $(R)_{i_j}$ induce the subtree $T|_j$ for $j = 1, \ldots, d$ in $T$ and $R$, respectively. Let $(i_j^+, i_j)$ be the edges between $(T)_{i_j}$ and its parent in $T$ for all $j = 1, \ldots, d$. If*

*(i)* $i_j \prec \min_{k=1,\dots,d}(i_k^+)$ *for all* $j = 1, \dots, d$ *and*

*(ii)*

$$E(R) = E(T) \cup \{(i_1^+, i_d)\} \bigcup_{j=1,\dots,d-1} \{(i_{j+1}^+, i_j)\} \setminus \Big( \bigcup_{j=1,\dots,d} \{(i_j^+, i_j)\} \Big),$$

*Then* $d_{\mathrm{HSPR}}(T, R) \leq d$.

Informally, the difference between $T$ and $R$ is the positioning of the subtrees $T|_1, T|_2, \dots, T|_d$, which we can interpret as a permutation of these subtrees. By changing every edge $(i_j^+, i_j)$ of $T$ to $(i_{j+1}^+, i_j)$ in $R$, every subtree $T|_j$ in $R$ is attached where $T|_{j+1}$ is in $T$ for all $j = 1, \dots, d-1$, and with $(i_1^+, i_1)$ being replaced by $(i_1^+, i_d)$, $T|_d$ in $R$ is attached where $T|_1$ is in $T$. We could describe this permutation of the subtrees by $(T|_1, T|_2, \dots, T|_d)$, using the cycle notation for permutations.

*Proof.* We assume without loss of generality that $i_1^+ = \min_{j=1,\dots,d} i_j^+$. If otherwise $i_m^+ = \min_{j=1,\dots,d} i_j^+$ for $m \neq 1$, we rename the subtrees $T|_{((j+d-m+1) \bmod d)+1}$ to $T|_j$ for all $j = 1, \dots, d$. By the assumptions of the lemma, all edges $(i_j, i_j^+)$ with $j \neq i$ in $T$ cover rank $k := i_1^+$. Let $i_1^{++} = \mathrm{parent}_T(k)$ and let $i_0$ be the sibling of $i_1$ in $T$, i.e. $\mathrm{parent}_T(i_0) = \mathrm{parent}_T(i_1) = k$.

We now create a path $p = [T_0 \simeq T, T_1, \dots, T_d \simeq R]$ of length $d$. We describe the moves on $p$ iteratively for every pair $T_{j-1}, T_j$ by using the edge set notation. For every move we make, we prove that it is a valid move by showing that the conditions of Theorem 2 are fulfilled. Note that we can infer from $T|_j \not\simeq T|_m$ that $i_j \neq i_m$ for all $j \neq m$.

We define the first move on $p$ by the following change of the edge set of $T$:

$$\begin{aligned} E(T_1) = E(T) \setminus &\{(i_1^{++}, k), (k, i_0), (i_2^+, i_2)\} \\ &\cup \{(i_1^{++}, i_0), (i_2^+, k), (k, i_2)\}. \end{aligned}$$

This is a valid move, because:

- $(i_1^{++}, k) \in E(T)$ by definition of $i_1^{++}$ and $k$.
- $(k, i_0) \in E(T)$ by definition of $k$ and $i_0$.
- $(i_2^+, i_2) \in E(T)$ by the assumption of the lemma.
- $(i_2^+, i_2)$ covers rank $k$ by the assumptions of the lemma.

The next moves on $p$ between $T_{j-1}$ and $T_j$ for $j = 2, \dots, d-1$ are described by the following changes in edge sets:

$$\begin{aligned} E(T_j) = E(T_{j-1}) \setminus &\{(i_j^+, k), (k, i_{j-1}), (i_{j+1}^+, i_{j+1})\} \\ &\cup \{(i_j^+, i_{j-1}), (i_{j+1}^+, k), (k, i_{j+1})\}. \end{aligned}$$

These are valid moves, because:

- $(i_j^+, k) \in E(T_{j-1})$, because this edge is added to $E(T_{j-1})$ by the previous move between $T_{j-2}$ and $T_{j-1}$.
- $(k, i_{j-1}) \in E(T_{j-1})$, because this edge is added in the move from $T_{j-3}$ to $T_{j-2}$, and it is not changed when moving from $T_{j-2}$ to $T_{j-1}$.
- $(i_{j+1}^+, i_{j+1}) \in E(T_{j-1})$, because $(i_{j+1}^+, i_{j+1}) \in E(T)$ by the assumptions of the lemma, and with $i_j \neq i_m$ for all $j \neq m$ and $i_{j+1}^+ \neq k$ for all $j \neq 1$, this edge is not changed on any previous move on $p$.
- $(i_{j+1}^+, i_{j+1})$ covers rank $k$ by the assumption of the lemma.

The last step, transforming $T_{d-1}$ to $T_d$, we apply an HSPR move to $T_{d-1}$ that changes the edge set as follows:

$$\begin{aligned} E(T_d) = E(T_{d-1}) \setminus \{&(i_d^+, k), (k, i_{d-1}), (i_1^{++}, i_0)\} \\ \cup \{&(i_d^+, i_{d-1}), (i_1^{++}, k), (k, i_0)\}. \end{aligned}$$

This is a valid move, because:

- $(i_d^+, k) \in E(T_{d-1})$, because this edge is added to $E(T_{d-1})$ by the previous move between $T_{d-2}$ and $T_{d-1}$.
- $(k, i_{d-1}) \in E(T_{d-1})$, because this edge is added in the move from $T_{d-3}$ to $T_{d-2}$, and it is not changed when moving from $T_{d-2}$ to $T_{d-1}$.
- $(i_1^{++}, i_0) \in E(T_{d-1})$, because it is added to $E(T_1)$ in the first move on $p$ and with $i_j \neq i_m$ for all $j \neq m$ and $i_1^{++} \neq k$, we can infer that it is not removed from the edge set of any tree on any previous move on $p$.
- $i_0 \prec k \prec i_1^{++}$ by definition of $k$.

We can summarise all changes to the edge sets along $p$ using multisets, which can be

simplified using $k = i_1^+$:

$$
\begin{aligned}
E(T_d) = E(T) &\cup \{(i_1^{++}, i_0), (i_2^+, k), (k, i_2)\} \\
&\bigcup_{j=2,\dots,d-1} \{(i_j^+, i_{j-1}), (i_{j+1}^+, k), (k, i_{j+1})\} \\
&\cup \{(i_d^+, i_{d-1}), (i_1^{++}, k), (k, i_0)\} \\
&\setminus \{(i_1^{++}, k), (k, i_0), (i_2^+, i_2)\} \\
&\setminus \Big( \bigcup_{j=2,\dots,d-1} \{(i_j^+, k), (k, i_{j-1}), (i_{j+1}^+, i_{j+1})\} \Big) \\
&\setminus \{(i_d^+, k), (k, i_{d-1}), (i_1^{++}, i_0)\} \\
= E(T) &\cup \{(i_1^+, i_d)\} \cup \bigcup_{j=2,\dots,d} \{(i_j^+, i_{j-1})\} \\
&\setminus \bigcup_{j=1,\dots,d} \{(i_j^+, i_j)\}.
\end{aligned}
$$

It follows $E(T_d) = E(R)$ and therefore, $T_d \simeq R$. Hence, $p$ is a path from $T$ to $R$ of length $d$, proving $d_{\mathrm{HSPR}}(T, R) \leq d$. □

Another interesting property of shortest paths is that of Lemma 4: No subtree can move twice by HSPR moves at the same rank. It is however possible for one subtree to move twice at different ranks on a shortest path. For example in Figure 7, the subtree only consisting of the leaf $a_5$ is moved by the first HSPR move at rank one and by the last HSPR move at rank two.

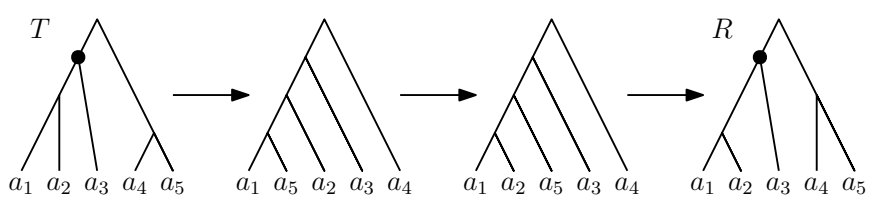


Figure 7: Shortest HSPR path between trees $T$ and $R$. The subtree consisting of only the leaf $a_5$ is moved by the first and the last HSPR move. The path displayed here does not preserve the shared cluster $\{a_1, a_2, a_3\}$ at rank three. The nodes inducing this cluster are highlighted in $T$ and $R$. On this path only subtrees consisting of single leaves move.

**Lemma 4.** *Let $T$ and $R$ be trees connected by a shortest path $p$ containing only* HSPR *moves at rank $k$, moving subtrees $T|_{i_0}, T|_{i_1}, \dots, T|_{i_{d-1}}$ in this order, i.e. $p$ has length $d$.*

*Then every $T|_{i_j}$ is subtree of both $T$ and $R$ and $T|_{i_j} \not\simeq T|_{i_l}$ for all $j \neq l \in \{0, 1, \dots, d-1\}$.*

*Proof.* Let $p$ be a shortest path only containing HSPR moves at rank $k$ as described in the lemma. Every subtree $T|_{i_j}$ with $j \in \{0, 1, \ldots, d-1\}$ is induced by a node $i_j$ with rank less than $k$ in $T$. Therefore, no edges inside any of these subtrees can change on $p$, which implies that $T|_{i_0}, \ldots, T|_{i_{d-1}}$ are subtrees in all trees on $p$, including $T$ and $R$.

To show $T|_{i_j} \not\simeq T|_{i_l}$ for all $j \neq l$, we assume to the contrary that a subtree $T' \in \{T|_{i_0}, \ldots, T|_{i_{d-1}}\}$ moves twice on a $p$. To simplify notation, we assume without loss of generality that it is $T' \simeq T|_{i_0} \simeq T|_{i_{d-1}}$ and that no other subtree moves twice on $p$. Note that it could be $d-1=1$, in which case $T'$ moves twice in two consecutive moves.

Let $i_j^+ = \text{parent}_T(i_j)$ in $T$. Since the first move on $p$ moves the subtree $T|_{i_0}$ and all moves on $p$ are HSPR moves at rank $k$, it is $i_0 = k$. Furthermore, the subtree $T|_{i_j}$ needs to have node $k$ as its parent before it gets pruned from a tree on $p$, as all moves are HSPR moves at rank $k$. This implied that $T|_{i_j}$ gets reattached on the edge $(i_{j+1}^+, i_{j+1})$ for every $j = 0, \ldots, d-2$ to ensure that the next move can prune the subtree $T|_{i_{j+1}}$ at rank $k$. Let $i_0$ be the sibling of $i_1$ in $T$, i.e. $\text{parent}_T(i_0) = \text{parent}_T(i_1) = k$ and let $i_1^{++} = \text{parent}_T(i_1^+)$.

Assuming $p = [T_0 \simeq T, T_1, \ldots, T_d \simeq R]$, the HSPR moves along $p$ can formally be described by the following changes in edge sets (see Theorem 3):

$$\begin{aligned} E(T_1) = E(T) \setminus &\{(i_1^{++}, k), (k, i_0), (i_2^+, i_2)\} \\ &\cup \{(i_1^{++}, i_0), (i_2^+, k), (k, i_2)\} \end{aligned}$$

and for $j = 2, \ldots, d-2$:

$$\begin{aligned} E(T_j) = E(T_{j-1}) \setminus &\{(i_j^+, k), (k, i_{j-1}), (i_{j+1}^+, i_{j+1})\} \\ &\cup \{(i_j^+, i_{j-1}), (i_{j+1}^+, k), (k, i_{j+1})\} \end{aligned}$$

By our assumptions on $p$, $T|_{i_j} \not\simeq T|_{i_m}$ for all $j \neq m$ in $\{1, \ldots, d-1\}$, we know that $(i_{j+1}^+, i_{j+1}) \in E(T_{j-1})$ for all $j = 1, \ldots, d-1$. We can use similar arguments to those used in the proof of Lemma 3 to show that the changes of edge sets presented here describe valid HSPR moves up to $T_{d-2}$.

For the last move on $p$, moving the subtree $T|_{i_d} \simeq T|_{i_0} \simeq T'$ by an HSPR move at rank $k$, the subtree $T|_{i_{d-1}}$ needs to become sibling of $T|_{i_d}$ by an HSPR move at rank $k$ between $T_{d-2}$ and $T_{d-1}$. The edge connecting $i_1$, which is the root of $T|_{i_d} \simeq T|_{i_0}$, and its parent is $(i_2^+, i_1)$, which is created by the move between $T_1$ and $T_2$ and with $i_2^+ \neq k$ and $i_j \neq i_k$ for all $j \neq k$ in $\{1, \ldots, d-1\}$, this edge is not changed again on $p$ until reaching $T_{d-2}$. The move between $T_{d-2}$ and $T_{d-1}$ can therefore be described by the following change in the edge set of $T_{d-2}$:

$$\begin{aligned} E(T_{d-1}) = E(T_{d-2}) \setminus &\{(i_{d-1}^+, k), (k, i_{d-2}), (i_2^+, i_1)\} \\ &\cup \{(i_{d-1}^+, i_{d-2}), (i_2^+, k), (k, i_1)\} \end{aligned}$$

Let $(i_{d+1}^+, i_{d+1})$ be the edge on which $T' \simeq T_s^d$ gets reattached by the last move on $p$. We

can then write this last move as

$$E(T_d \simeq R) = E(T_{d-1}) \setminus \{(i_2^+, k), (k, i_{d-1}), (i_{d+1}^+, i_{d+1})\} \\ \cup \{(i_2^+, i_{d-1}), (i_{d+1}^+, k), (k, i_{d+1})\}.$$

It is again not hard to see with Theorem 2 that this describes a valid HSPR move.

Using multisets, we can describe the difference between $E(R)$ and $E(T)$ as follows:

$$\begin{aligned} E(R) = E(T) &\cup \{(i_1^{++}, i_0), (i_2^+, k), (k, i_2)\} \\ &\bigcup_{j=2,\dots,d-2} \{(i_j^+, i_{j-1}), (i_{j+1}^+, k), (k, i_{j+1})\} \\ &\cup \{(i_{d-1}^+, i_{d-2}), (i_2^+, k), (k, i_1)\} \\ &\cup \{(i_2^+, i_{d-1}), (i_{d+1}^+, k), (k, i_{d+1})\} \\ &\setminus \{(i_1^{++}, k), (k, i_0), (i_2^+, i_2)\} \\ &\setminus \Big( \bigcup_{j=2,\dots,d-2} \{(i_j^+, k), (k, i_{j-1}), (i_{j+1}^+, i_{j+1})\} \Big) \\ &\setminus \{(i_{d-1}^+, k), (k, i_{d-2}), (i_2^+, i_1)\} \\ &\setminus \{(i_2^+, k), (k, i_{d-1}), (i_{d+1}^+, i_{d+1})\} \\ = E(T) &\cup \{(i_1^{++}, i_0), (i_2^+, i_{d-1}), (i_{d+1}^+, k), (k, i_{d+1})\} \\ &\bigcup_{j=3,\dots,d-1} \{(i_j^+, i_{j-1})\} \\ &\setminus \{(i_1^{++}, k), (k, i_0), (i_{d+1}^+, i_{d+1})\} \\ &\setminus \Big( \bigcup_{j=2,\dots,d-1} \{(i_j^+, i_j)\} \Big). \end{aligned}$$

Based on this, we can define an alternative path from $T$ to $R$. Therefore, we define a neighbour $\hat{T}$ of $T$ by

$$E(\hat{T}) = E(T) \setminus \{(i_1^{++}, k), (k, i_0), (i_{d+1}^+, i_{d+1})\} \\ \cup \{(i_1^{++}, i_0), (i_{d+1}^+, k), (k, i_{d+1})\}.$$

This describes a valid HSPR move on $T$, because:

- $(i_1^{++}, k) \in E(T)$ by the definition of $i_1^{++}$ and $k$.
- $(k, i_0) \in E(T)$ by the definition of $k$ and $i_0$.
- $(i_{d+1}^+, i_{d+1}^-) \in E(T)$, because $(i_{d+1}^+, i_{d+1}^-) \in E(T_{d-1})$ by the definition of moves on $p$ and this edge is not deleted from the edge set on any move on $p$ before $T_{d-1}$ is reached.

- $(i^+_{d+1}, i^-_{d+1})$ covers rank $k$ by the assumption that all HSPR move are HSPR moves at rank $k$.

Knowing the difference between $E(T)$ and $E(R)$, we can summarise the difference between $E(R)$ and $E(\hat{T})$ to:

$$E(R) = E(\hat{T}) \cup \{(i^+_2, i_{d-1})\} \bigcup_{j=3,\dots,d-1} \{(i^+_j, i_{j-1})\} \setminus \Big( \bigcup_{j=2,\dots,d-1} \{(i^+_j, i_j)\} \Big).$$

$\hat{T}$ and $R$ fulfil the assumptions of Lemma 3, so we get $d_{\mathrm{HSPR}}(\hat{T}, R) = d-2$. Therefore, the path from $T$ to $R$ via $\hat{T}$ has length $1 + d - 2 = d - 1$ and is shorter than $p$, contradicting our assumption that $p$ is a shortest path. Hence, no subtree is moved twice on a shortest with only HSPR moves at fixed rank $k$. □

We are now ready to show that a shared cherry at rank one is preserved on every shortest path.

**Theorem 9.** *Let $T$ and $R$ be trees and $x, y \in \{l_1, \dots, l_n\}$ so that both $T$ and $R$ have $\{x, y\}$ as cherry at rank one. Then every tree on every shortest path between $T$ and $R$ has $\{x, y\}$ as cherry at rank one.*

*Proof.* For this proof we assume to the contrary that $T$ and $R$ are connected by a shortest path $p$ containing a tree whose cherry at rank one is not $\{x, y\}$. We furthermore assume that $T$ and $R$ have minimum distance among all tree pairs connected by such a shortest path. It then follows that $p$ contains HSPR moves at rank one only: Otherwise, we could change the order of ranks of HSPR moves along the path to have all rank one HSPR moves first (Lemma 2). Since the cherry at rank one cannot change by HSPR moves at rank greater than one, $T$ and $R$ would then not be a minimum distance counterexample. Therefore, all HSPR moves on $p$ are at rank one and hence move subtrees that consist of one leaf only.

By the assumption on $T$ and $R$, either $x$ or $y$ is pruned by the first move on $p$. We assume that $x$ is moved first, otherwise we swap notations for $x$ and $y$. Since the last move on $p$ reconstructs the cherry $\{x, y\}$ at rank one, and the subtree containing only the leaf $x$ can only move once at rank one (Lemma 4), the last leaf that moves on $p$ is $y$. Let $a_0 = x, a_1, a_2, \dots, a_{d-1}, a_d = y$ be the sequence of leaves moved on $p$. By Lemma 4, all these leaves are distinct i.e. $a_i \neq a_j$ for all $i \neq j$. Note that $d - 1 \geq 1$, as after the first move on $p$ the parent of $y$ has rank greater than one, but the last move on $p$ moves $y$ by an HSPR move at rank one, for which the parent of $y$ needs to have rank one. Moving $a_{d-1} \neq x$ to become sibling of $y$ in the second to last move is hence necessary to get to $R$, so $d - 1 \geq 1$. The path $p$ as described above is depicted in Figure 8.

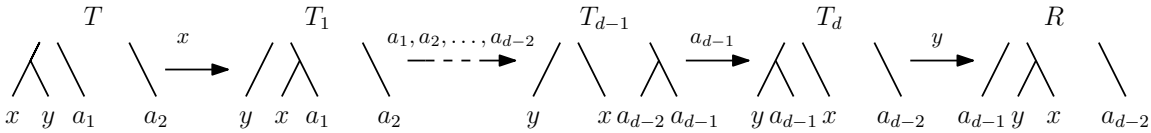


Figure 8: Trees $T$ and $R$ with cherry $\{x, y\}$ at rank one and all moves on a path $p$ from $T$ to $R$ that does not preserve the cherry as described in the proof of Theorem 9. The labels of the arrows indicate the leaves that are pruned and reattached by the corresponding HSPR moves.

Let $a_i^+ = \text{parent}_T(a_i)$ for all $i = 1, \ldots, d$ and $p_{xy} = \text{parent}_T(1)$. Note that this implies $a_i^+ \neq 1$ for all $i$. Assuming $p = [T_0 \simeq T, T_1, \ldots, T_d, T_{d+1} \simeq R]$, we can then describe the moves on $p$ similar to how it has been done in the proof of Lemma 4:

$$\begin{aligned} E(T_1) = E(T_0) \setminus \{&(p_{xy}, 1), (1, y), (a_1^+, a_1)\} \\ \cup \{&(p_{xy}, y), (a_1^+, 1), (1, a_1)\}. \end{aligned}$$

This describes a valid move, since:

- $(p_{xy}, 1) \in E(T_0)$ by the definition of $p_{xy}$.
- $(1, y) \in E(T_0)$, because $y$ is in the cherry of rank one.
- $(a_1^+, a_1) \in E(T_0)$ by the definition of $a_1^+$.
- $(a_1^+, a_1)$ covers rank 1, because $a_1$ is a leaf and $a_1^+ \neq 1$.

$$\begin{aligned} E(T_j) = E(T_{j-1}) \setminus \{&(a_{j-1}^+, 1), (1, a_{j-2}), (a_j^+, a_j)\} \\ \cup \{&(a_{j-1}^+, a_{j-2}), (a_j^+, 1), (1, a_j)\} \end{aligned}$$

$$\text{for } j = 2, \ldots, d-1.$$

This describes a valid move, because:

- $(a_{j-1}^+, 1) \in E(T_{j-1})$, because it has been added to this set by the previous move between $T_{j-2}$ and $T_{j-1}$.
- $(1, a_{j-2}) \in E(T_{j-1})$, because it has been added to $E(T_{j-2})$ by the move between $T_{j-3}$ and $T_{j-2}$ and with $a_i \neq a_j$ for all $i \neq j$, this edge has not been removed from $E(T_{j-2})$ to get $E(T_{j-1})$.
- $(a_j^+, a_j) \in E(T_{j-1})$, because $(a_j^+, a_j) \in E(T)$ and with $a_i \neq a_j$ for all $i \neq j$, this edge is not removed from the edge set in any previous move on $p$.
- $(a_j^+, a_j)$ covers rank 1, because $a_j$ is a leaf and $a_j^+ \neq 1$.

$$E(T_d) = E(T_{d-1}) \setminus \{(a_{d-1}^+, 1), (1, a_{d-2}), (p_{xy}, y)\} \\ \cup \{(a_{d-1}^+, a_{d-2}), (p_{xy}, 1), (1, y)\}$$

This describes a valid move, because:

- $(a_{d-1}^+, 1) \in E(T_{d-2})$, because it has been added to this set by the previous move between $T_{d-3}$ and $T_{d-2}$.
- $(1, a_{d-2}) \in E(T_{d-2})$, because it has been added to $E(T_{d-3})$ by the move between $T_{d-2}$ and $T_{d-2}$.
- $(p_{xy}, y) \in E(T_{d-2})$, because it has been added to $E(T_1)$ by the first move on $p$ and with $y \neq a_i, 1$ for all $i$, this edge is not removed from the edge set in any previous move on $p$.
- $(p_{xy}, y)$ covers rank 1, because $(p_{xy}, 1)$ and $(1, y)$ are edges in $T$.

$$E(R) = E(T_d) \setminus \{(p_{xy}, 1), (1, a_{d-1}), (a_1^+, x)\} \\ \cup \{(p_{xy}, a_{d-1}), (a_1^+, 1), (1, x)\}$$

This describes a valid move, because:

- $(p_{xy}, 1) \in E(T_d)$, because it has been added to this set by the previous move between $T_{d-1}$ and $T_d$.
- $(1, a_{d-1}) \in E(T_d)$, because it has been added to $E(T_{d-2})$ by the move between $T_{d-2}$ and $T_{d-1}$ and with $a_i \neq a_j$ for all $i \neq j$, this edge has not been removed from $E(T_{d-1})$ to get $E(T_d)$.
- $(a_1^+, x) \in E(T_{j-1})$, because it has been added to $E(T_2)$ by the second move on $p$ ($x = a_0$) and with $x \neq a_i, 1$ for all $i$, this edge is not removed from the edge set in any previous move on $p$.
- $(a_1^+, x)$ covers rank 1, because $x$ is a leaf and $1 \prec a_i^+$ for all $i$.

Since some edges that are added along $p$ are later deleted, we can describe the changes

between $E(T)$ and $E(R)$ by using multisets:

$$\begin{aligned}
E(R) := E(T) \cup &\{(p_{xy}, y), (a_1^+, 1), (1, a_1)\} \\
&\bigcup_{j=2,\dots,d-1} \{(a_{j-1}^+, a_{j-2}), (a_j^+, 1), (1, a_j)\} \\
&\cup \{(a_{d-1}^+, a_{d-2}), (p_{xy}, 1), (1, y)\} \\
&\cup \{(p_{xy}^+, a_{d-1}), (a_1^+, 1), (1, x)\} \\
&\setminus \{(p_{xy}, 1), (1, y), (a_1^+, a_1)\} \\
&\setminus \Big( \bigcup_{j=2,\dots,d-1} \{(a_{j-1}^+, 1), (1, a_{j-2}), (a_j^+, a_j)\} \Big) \\
&\setminus \{(a_{d-1}^+, 1), (1, a_{d-2}), (p_{xy}, y)\} \\
&\setminus \{(p_{xy}, 1), (1, a_{d-1}), (a_1^+, x)\},
\end{aligned}$$

which we can summarise to:

$$\begin{aligned}
E(R) =& E(T) \cup \Big( \{(a_1^+, 1), (p_{xy}, a_{d-1})\} \bigcup_{j=1,\dots,d-2} \{(a_{j+1}^+, a_j)\} \Big) \\
&\setminus \Big( (p_{xy}, 1) \bigcup_{j=1,\dots,d-1} \{(a_j^+, a_j)\} \Big).
\end{aligned}$$

$T$ and $R$ fulfil the requirements of the trees of Lemma 3, where $i_j^+ = a_j^+$ and $i_j = a_j$ for all $j = 1, \dots, d-1$, $i_d^+ = p_{xy}$, and $i_d = 1$. Applying this lemma gives us $d_{\mathrm{HSPR}}(T, R) \leq d$.

Therefore, $p$ is not shortest paths, contradicting our assumption that there is a shortest path from $T$ to $R$ that does not preserve the shared cherry. □

Theorem 9 implies that the distance of two trees with identical cherry at rank one does not change when one of the leaves in this cherry is deleted (Corollary 3). We will see in Theorem 11 that we can generally not assume that the distance stays the same or decreases when deleting a leaf from two trees.

**Corollary 3.** *Let $T$ and $R$ be trees on $n$ leaves sharing their cherry $\{c_1, c_2\}$ at rank one and let $T'$ and $R'$ result from deleting $c_2$ from $T$ and $R$, respectively, suppressing the internal node of rank one and subtracting one from the rank of all remaining internal nodes so that $T$ and $R$ are trees on $n-1$ leaves. Then $d_{\mathrm{HSPR}}(T', R') = d_{\mathrm{HSPR}}(T, R)$.*

*Proof.* By Theorem 9 all shortest HSPR paths from $T$ to $R$ preserve the cherry $\{c_1, c_2\}$. Therefore, deleting $c_2$ from every tree on a shortest $T$-$R$-path, suppressing the resulting degree-two nodes, and subtracting one from the ranks of all remaining nodes, gives a path

between $T'$ and $R'$, i.e. $d_{\text{HSPR}}(T', R') \leq d_{\text{HSPR}}(T, R)$. If there was a path between $T'$ and $R'$ that was shorter than $d_{\text{HSPR}}(T, R)$, then adding a leaf $c_2$ as sibling of $c_1$ with parent of rank one and adding one to the ranks of all other internal nodes in every tree on $p$ results in a path between $T$ and $R$ of length less than $d_{\text{HSPR}}(T, R)$, which is a contradiction. Hence, $d_{\text{HSPR}}(T, R) = d_{\text{HSPR}}(T', R')$. □

Another observation that follows from Theorem 9 is that if the lowest part of two trees, i.e. all clusters up to a certain rank, is identical, then no shortest path in HSPR changes this part of the tree.

**Corollary 4.** *Let $T$ and $R$ be trees so that for some $i \in \{2, \ldots, n\}$ the cluster induced by $(T)_j$ is identical to the cluster induced by $(R)_j$ for all $j < i$. Then every node $(T')_j$ in every tree $T'$ on every shortest path between $T$ and $R$ induces the same cluster as $(T)_j$ and $(R)_j$ for all $j < i$.*

*Proof.* To prove this corollary, we assume to the contrary that there is a node with rank less than $i$ that induces the same cluster in $T$ and $R$, but this cluster is not present on a shortest path $p$ between $T$ and $R$. Let $j$ be the rank of such a node so that there is no other node with this property in $T$ and $R$ with rank less than $j$. Then all clusters induced by nodes of rank less than $j$ are present in all trees on $p$. We can iteratively apply Corollary 3 to the trees $T$ and $R$ where in the first iteration deleting a cherry leaf and updating the rest of the tree as described in the lemma results in trees $T_1$ and $R_1$ with $d_{\text{HSPR}}(T, R) = d_{\text{HSPR}}(T_1, R_1)$. This can be repeated until trees $T_{j-1}$, $R_{j-1}$ are received with $d_{\text{HSPR}}(T_{j-1}, R_{j-1}) = d_{\text{HSPR}}(T, R)$, where $T_{j-1}$ and $R_{j-1}$ are trees on $n-(j-1)$ leaves. As all clusters induced by nodes of rank less than $j$ are present in all trees on $p$, the leaves that are deleted from $T$ and $R$ to receive $T_{j-1}$ and $R_{j-1}$, respectively, can be deleted in the same order in all trees on $p$ in the same way, and we receive a path $p'$ from $T_{j-1}$ to $R_{j-1}$ with $|p'| = |p|$. Since the cluster induced by the nodes of rank $j$ is the same in $T$ and $R$, $T_{j-1}$ and $R_{j-1}$ have the same cherry at rank one. Because $p$ does not preserve the cluster at rank $j$, there must be a tree on $p'$ not containing the cherry of rank one in $T_{j-1}$ and $R_{j-1}$, which implies by Corollary 3 that $p'$ is not a shortest path, i.e. $d_{\text{HSPR}}(T_{j-1}, R_{j-1}) < |p'| = |p|$. This however is a contradiction to $d_{\text{HSPR}}(T_{j-1}, R_{j-1}) = d_{\text{HSPR}}(T, R) = |p|$. Therefore, there cannot be a shortest path $p$ between $T$ and $R$ that does not preserve the cluster of rank $j < i$ that is present in $T$ and $R$, which concludes the proof of this corollary. □

It is in general not true that a cluster $C$ that is induced by nodes of the same rank $r$ in two trees $T$ and $R$ is present in all trees on all shortest paths. A counterexample to this can be found in Figure 7 where the shared cluster $\{a_1, a_2, a_3\}$ at rank three is not present in every tree on any shortest path from $T$ to $R$, which we found out through exhaustive search using our implementation [8].

### 4.2 RSPR **Shortest Paths**

The RSPR space can be interpreted as an extension of HSPR in which rank moves are added to provide shortcuts between some trees. Here, we investigate how the addition of rank moves changes shortest paths. Our analysis of these paths will provide insights into the relationship of the complexity of computing distances in HSPR and RSPR. We show in Theorem 10 that we can change the order of moves on paths in RSPR, while not changing their length, so that all rank moves are grouped at the beginning followed by a sequence of only HSPR moves. This indicates that shortest paths in HSPR and RSPR can be very similar, sometimes identical (e.g. in the case of caterpillar trees, Corollary 6), and suggests that the complexity of computing shortest paths (and distances) might be the same in both treespaces. We first need the following lemma:

**Lemma 5.** *Let $T$ and $R$ trees and $p = [T, T', R]$ a path with an* HSPR *move between $T$ and $T'$ and a rank move between $T'$ and $R$. Then there is a path $p' = [T, T'', R]$ consisting of either two* HSPR *moves or a rank move followed by an* HSPR *move.*

Whether the path $p'$ in Lemma 5 contains two HSPR moves or a rank move followed by an HSPR move depends on the specific moves on $p$, as we will see in the proof of this lemma. For this proof we use the cluster representation of trees and describe HSPR moves by the corresponding changes in cluster representation as described in Theorem 3.

*Proof.* Let $k$ and $k+1$ for some $k \in \{1, \ldots, n-3\}$ be the ranks of the nodes of the rank move between $T'$ and $R$. Let $T' = [C_1, \ldots, C_{k-1}, A \cup B, C \cup D, C_{k+2}, \ldots, C_{n-1}]$ be the cluster representation of $T'$, which is illustrated in the top middle of Figure 9. Since $T'$ and $R$ are connected by a rank move of nodes $k$ and $k+1$, the cluster representation of $R$ is $R = [C_1, \ldots, C_{k-1}, C \cup D, A \cup B, C_{k+2}, \ldots, C_{n-1}]$ (see top right of Figure 9).

In the following, we distinguish different HSPR moves possible between $T$ and $T'$ and show how to replace $T'$ by a tree $T''$ to get a path $p' = [T, T'', R]$ with either two HSPR moves or a rank move followed by an HSPR move.

1. The HSPR move between $T$ and $T'$ is neither a rank $k$ nor a rank $k+1$ HSPR move.

   An HSPR move at rank greater than $k$ does not change the clusters induced by nodes $k$ and $k+1$, so we can in this case first perform a rank move of nodes $k$ and $k+1$ on $T$ to get a tree $T''$. Since all clusters of rank greater than $k+1$ are identical in $T$ and $T''$ we can perform an HSPR move on $T''$ that changes the clusters of this tree in exactly the same way as they change between $T$ and $T'$. This HSPR move on $T''$ then results in $R$, giving us a path $p' = [T, T'', R]$ with the desired properties.

   If the HSPR move between $T$ and $T'$ is an HSPR move at rank $r < k$, it might change the clusters induced by the nodes of rank $k$ and $k+1$. It does however not matter, in which order these two clusters appear in the tree, they would change in the exact same way if they were swapped. Therefore, we can first swap the ranks of the nodes

with rank $k$ and $k+1$ in $T$, resulting in a tree $T''$, and then perform the same HSPR move on $T''$ as the one between $T$ and $T'$. This results in a path $p'' = [T, T'', R]$ with a rank move between $T$ and $T''$ and an HSPR move between $T''$ and $R$.

2. $T$ and $T'$ are connected by an HSPR move at rank $k$. Without loss of generality we can assume that $T|_A$ is moved between $T$ and $T'$, otherwise we swap notations for $A$ and $B$. We now further distinguish whether there is an edge connecting the nodes of rank $k$ and $k+1$ in $T$.

   2.1. Let there be an edge connecting the nodes with ranks $k$ and $k+1$ in $T$. Note that by our assumptions on $T'$, $\text{parent}_{T'}(T|_C) = \text{parent}_{T'}(T|_D) = k+1$ and $\text{parent}_{T'}(T|_A) = k$. Therefore, an HSPR move at rank $k$ on $T'$ that moves the subtree $T|_A$ and creates a tree $T$ containing an edge $(k+1, k)$ must move $T|_A$ to become sibling of either $T|_C$ or $T|_D$. We assume without loss of generality that $T|_A$ and $T|_D$ are siblings in $T$, as depicted in Figure 9, otherwise we change notation for $C$ and $D$. Then the cluster representation of $T$ is:

$$T = [C_1, \ldots, C_{k-1}, A \cup D, A \cup C \cup D, C'_{k+2}, \ldots, C'_{n-2}],$$

   where for all $m = k+1, \ldots, n-2$

$$C'_m = \begin{cases} C_m \setminus A & \text{if } A \subset C_m \text{ and } D \not\subset C_m \\ C_m \cup A & \text{if } D \subset C_m \\ C_m & \text{otherwise.} \end{cases}$$

   To get a path $p'$ with the desired properties, we perform an HSPR move at rank $k$ on $T$ that moves $T|_D$ to become sibling of $T|_C$, resulting in a tree $T''$ in which the parent of $T''|_A$ has rank $k+1$. This tree $T''$ has cluster representation:

$$T'' = [C_1, \ldots, C_{k-1}, C \cup D, A \cup C \cup D, C''_{k+2}, \ldots, C''_{n-1}]$$

   where for all $m = k+1, \ldots, n-2$:

$$C''_m = \begin{cases} C'_m \setminus D & \text{if } D \subset C'_m \text{ and } C \not\subset C'_m \\ C'_m \cup D & \text{if } C \subset C'_m \\ C'_m & \text{otherwise.} \end{cases}$$

   Remember that every cluster is the union of two clusters at lower rank and/or leaves of a tree. Since the node $(T)_{k+1}$ induced the cluster $A \cup C \cup D$, it is $A \subset C'_m$ if and only if $(C \cup D) \subset C'_m$ for all $m \geq k+1$. Therefore, all clusters induced by nodes with rank greater than $k+1$ are the same in $T$ and $T''$: $C''_m = C'_m$ for all $m = k+2, \ldots, n-1$. We can now perform an HSPR move at

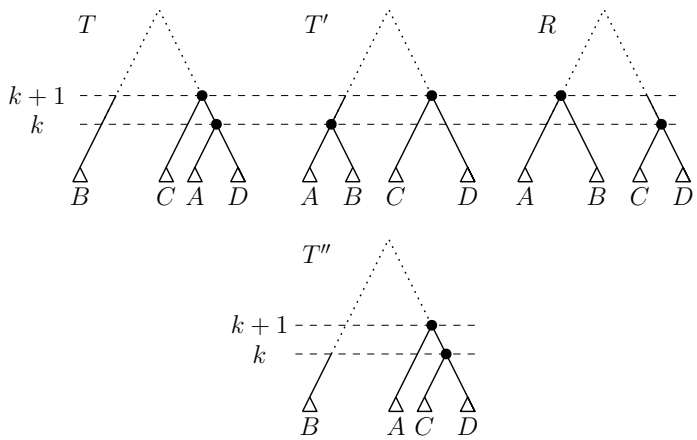


Figure 9: Trees $T$, $T'$, and $R$ on $p$, and alternative path $p'$ from $T$ to $R$ via $T''$ at the bottom if $T$ and $T'$ are connected by an HSPR move at rank $k$ and the nodes of rank $k$ and $k+1$ in $T$ are connected by an edge, as explained in Figure 2.1.. The dotted parts of the trees might contain further nodes and leaves.

rank $k+1$ on $T''$ that moves the subtree $T''|_A$ to become sibling of $T''|_B$, which results in a tree $\hat{R}$ with cluster representation:

$$\hat{R} = [C_1, \ldots, C_{k-1}, C \cup D, A \cup B, \hat{C}_{k+2}, \ldots, \hat{C}_{n-2}]$$

where for all $m = k+1, \ldots, n-2$:

$$\hat{C}_m = \begin{cases} C'_m \setminus A & \text{if } A \subset C'_m \text{ and } B \not\subset C'_m \\ C'_m \cup A & \text{if } B \subset C'_m \\ C'_m & \text{otherwise.} \end{cases}$$

Using the fact that $B \subset C_m$ if and only if $B \subset C'_m$, we can summarise the change of a cluster at rank $m > k+1$ between $T$ and $\hat{R}$ as follows:

$$\hat{C}_m = \begin{cases} C_m \setminus A & \text{if } A \subset C_m \text{ and } B \not\subset C_m \text{ and } D \not\subset C_m \\ C_m \cup A & \text{if } B \subset C_m \text{ and } D \subset C_m \\ C_m & \text{otherwise.} \end{cases}$$

Since the cluster at rank $k$ in $T'$ is $A \cup B$, it is $A \subset C_m$ if and only if $B \subset C_m$ for all $m > k$. Therefore, $\hat{C}_m = C_m$ for all $m > k+1$. It follows that the cluster representation of $\hat{R}$ and $R$ coincides, which gives us $\hat{R} \simeq R$, so $p'$ is a path from $T$ to $R$ with two HSPR moves (see Figure 9).

2.2. If there is no edge between the nodes with rank $k$ and $k+1$ in $T$, then there is a cluster $E$ in $T$ so that $T|_A$ is sibling of $T|_E$ in $T$ with $E \neq A, B, C, D$ (see Figure 10). Then the cluster representation of $T$ is:

$$T = [C_1, \ldots, C_{k-1}, A \cup E, C \cup D, C'_{k+2}, \ldots, C'_{n-1}]$$

where for all $m = k+1, \ldots, n-2$:

$$C'_m = \begin{cases} C_m \setminus A & \text{if } A \subset C_m \text{ and } E \not\subset C_m \\ C_m \cup A & \text{if } E \subset C_m \\ C_m & \text{otherwise.} \end{cases}$$

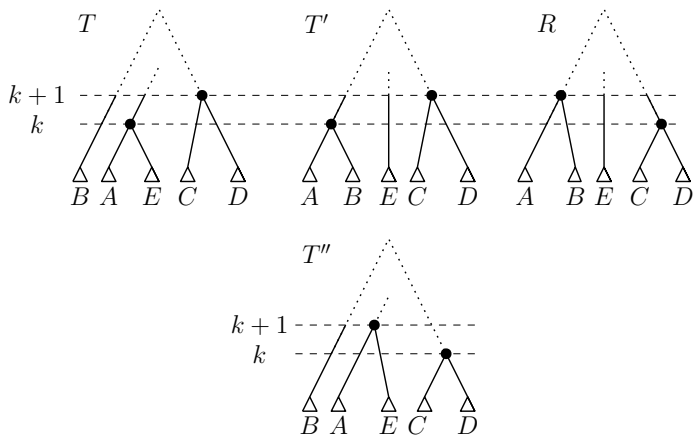


Figure 10: Trees $T$, $T'$, and $R$ on $p$ at the top, and alternative path $p'$ from $T$ to $R$ via $T''$ at the bottom if $T$ and $T'$ are connected by an HSPR move at rank $k$ and the nodes of rank $k$ and $k+1$ in $T$ are not connected by an edge. The dotted parts of the trees might contain further nodes and leaves.

To create an alternative path $p'$, we first perform a rank move swapping the ranks of the nodes $k$ and $k+1$ in $T$ to receive a tree $T''$ with cluster representation:

$$T'' = [C_1, \ldots, C_{k-1}, C \cup D, A \cup E, C'_{k+2}, \ldots, C'_{n-1}].$$

As second move on $p'$ we perform an HSPR move at rank $k+1$, moving the subtree $T''|_A$ to become sibling of $T''|_B$, giving us the following tree $\hat{R}$:

$$\hat{R} = [C_1, \ldots, C_{k-1}, C \cup D, A \cup B, \hat{C}_{k+2}, \ldots, \hat{C}_{n-1}],$$

where clusters $\hat{C}_m$ with $m \geq k+2$ are defined as follows:

$$\hat{C}_m = \begin{cases} C'_m \setminus A & \text{if } A \subset C'_m \text{ and } B \not\subset C'_m \\ C'_m \cup A & \text{if } B \subset C'_m \\ C'_m & \text{otherwise.} \end{cases}$$

To show that $\hat{C}_m = C_m$ for all $m = k+2, \ldots, n-1$, we distinguish whether $A \subset C_m$ or $A \not\subset C_m$. Note that $A \subset C_m$ if and only if $B \subset C_m$ and also $B \subset C_m$ if and only if $B \subset C'_m$.

If $A \subset C_m$ and $E \not\subset C_m$, then $C'_m = C_m \setminus A$ and since $B \subset C'_m$, $\hat{C}_m = C'_m \cup A$, resulting in $\hat{C}_m = C_m$. If on the other hand $A \subset C_m$ and $E \subset C_m$, then $C'_m = C_m \cup A$ and with $B \subset C'_m$, $\hat{C}_m = C'_m \cup A$, resulting in $\hat{C}_m = C_m$.

If $A \not\subset C_m$ and $E \subset C_m$, then $C'_m = C_m \cup A$ and since $B \not\subset C'_m$, $\hat{C}_m = C'_m \setminus A$, resulting in $\hat{C}_m = C_m$. If on the other hand $A \not\subset C_m$ and $E \not\subset C_m$, then $C'_m = C_m$ and since $B \not\subset C'_m$, $\hat{C}_m = C_m$, resulting in $\hat{C}_m = C_m$.

So for every $m = k+2, \ldots, n-1$, it is $\hat{C}_m = C'_m$, which implies $\hat{R} \simeq R$, so $p'$ is a path from $T$ to $R$ with first a rank move and then an HSPR move.

3. $T$ and $T'$ are connected by an HSPR move at rank $k+1$.

   As $T|_C$ and $T|_D$ are the subtrees that are children of node $k+1$ in $T'$, we can assume without loss of generality that $T|_C$ is moved between $T$ and $T'$, otherwise we swap notation for $C$ and $D$.

   We again further distinguish whether the node of rank $k+1$ is parent of the node of rank $k$ in $T$ or not.

   3.1. Let there be an edge connecting nodes $k+1$ and $k$ in $T$. Remember that by our assumptions on $T'$, $\text{parent}_{T'}(T|_A) = \text{parent}_{T'}(T|_B) = k$. Therefore, an HSPR move at rank $k+1$ on $T'$ that moves the subtree $T|_C$ and creates a tree $T$ containing an edge $(k+1, k)$ must move $T|_C$ to become sibling of $T|_{A \cup B}$ (see Figure 11).

   Then $T$ has the following cluster representation:

$$T = [C_1, \ldots, C_{k-1}, A \cup B, A \cup B \cup C, C'_{k+2}, \ldots C'_{n-1}],$$

   where

$$C'_m = \begin{cases} C_m \setminus C & \text{if } C \subset C_m \text{ and } (A \cup B) \not\subset C_m \\ C_m \cup C & \text{if } (A \cup B) \subset C_m \\ C_m & \text{otherwise} \end{cases}$$

   for all $m = k+2, \ldots, n-1$.

   A path $p'$ can now be constructed by first performing an HSPR move at rank $k$ on $T$ moving $T|_A$ to become sibling of $T|_C$ in the resulting tree $T''$:

$$T'' = [C_1, \ldots, C_{k-1}, A \cup C, A \cup B \cup C, C''_{k+2}, \ldots, C''_{n-1}]$$

   with

$$C''_m = \begin{cases} C'_m \setminus A & \text{if } A \subset C'_m \text{ and } C \not\subset C'_m \\ C'_m \cup A & \text{if } C \subset C'_m \\ C'_m & \text{otherwise} \end{cases}$$

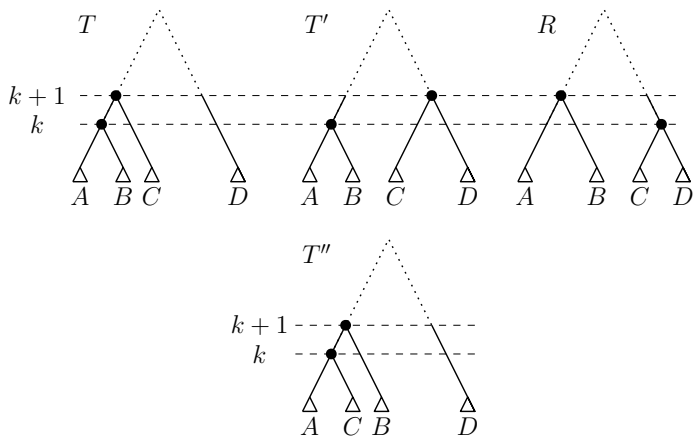


Figure 11: Trees $T$, $T'$, and $R$ on $p$, and alternative path $p'$ from $T$ to $R$ via $T''$ at the bottom if $T$ and $T'$ are connected by an HSPR move at rank $k+1$ and the nodes of rank $k$ and $k+1$ in $T$ are connected by an edge, The dotted parts of the trees might contain further nodes and leaves.

for all $m = k+2, \ldots, n-1$. Because the node of rank $k+1$ in $T$ induced the cluster $A \cup B \cup C$, $A \subset C'_m$ if and only if $C \subset C'_m$ for all $m = k+2, \ldots, n-1$. Therefore, $C''_m = C'_m$ for all $m = k+2, \ldots, n-1$. We then perform an HSPR move at rank $k$ on $T''$ that moves the subtree $T''|_C$ to become sibling of $T''|_D$, resulting in the following tree $\hat{R}$:

$$\hat{R} = [C_1, \ldots, C_{k-1}, C \cup D, A \cup B, \hat{C}_{k+2}, \ldots, \hat{C}_{n-1}]$$

with

$$\hat{C}_m = \begin{cases} C'_m \setminus C & \text{if } C \subset C'_m \text{ and } D \not\subset C'_m \\ C'_m \cup C & \text{if } D \subset C'_m \\ C'_m & \text{otherwise} \end{cases}$$

for all $m = k+2, \ldots, n-1$. To show $\hat{R} \simeq R$, we use that $C \subset C_m$ if and only if $D \subset C_m$, and $A \subset C_m$ if and only if $B \subset C_m$ for all $m = k+2, \ldots, n-1$, because the clusters induced by nodes of rank $k$ and $k+1$ in $T'$ are $A \cup B$ and $C \cup D$, respectively. Furthermore, we distinguish whether $(A \cup B) \subset C_m$, $(C \cup D) \subset C_m$, and $(A \cup B \cup C \cup D) \cap C_m = \emptyset$

If $(A \cup B) \subset C_m$ for some $m = k+2, \ldots, n-1$, then $C'_m = C_m \cup C$. Then (i) $\hat{C}_m = C'_m \setminus C$ if $D \not\subset C'_m$ or (ii) $\hat{C}_m = C'_m \cup C$ if $D \subset C'_m$. Note that if $D \subset C'_m$, it must be $D \subset C_m$, as $D$ is not removed from any clusters between trees $T'$ and $T$. And since $D \subset C_m$ if and only if $C \subset C_m$, we can see that $\hat{C}_m = C_m$ for both cases (i) and (ii).

If $(C \cup D) \subset C_m$ for some $m = k+2, \ldots, n-1$, then (i) $C'_m = C_m \setminus C$ if $(A \cup B) \not\subset C_m$ or (ii) $C'_m = C_m \cup C$ if $(A \cup B) \subset C_m$. Again, it is $D \subset C'_m$ if and only if $D \subset C_m$. Therefore, it must be $\hat{C}_m = C'_m \cup C$ in both cases (i) and (ii). With $C \subset C_m$, it follows $\hat{C}_m = C_m$.

If $(A \cup B \cup C \cup D) \cap C_m = \emptyset$, then $C'm = C_m$ and $\hat{C}_m = C'_m$, hence $\hat{C}_m = C_m$. We can conclude that $\hat{R} \simeq R$, so $p'$ is a path from $T$ to $R$ consisting of two HSPR moves.

3.2. There is no edge between the nodes of rank $k$ and $k+1$ in $T$.

Let $E$ be a cluster in $T$ so that the subtree $T|_E$ is sibling of $T|_C$ in $T$ with $E \neq A, B, C, D$, i.e. $T|_C$ is moved to become sibling of $T|_E$ by the HSPR move between $T'$ and $T$ (see Figure 12). The cluster notation for $T$ is:

$$T = [C_1, \ldots, C_{k-1}, A \cup B, C \cup E, C'_{k+2}, \ldots, C'_{n-1}]$$

with

$$C'_m = \begin{cases} C_m \setminus C & \text{if } C \subset C_m \text{ and } E \not\subset C_m \\ C_m \cup C & \text{if } E \subset C_m \\ C_m & \text{otherwise} \end{cases}$$

for all $m = k+2, \ldots, n-1$.

We construct a path $p'$ by first performing a rank move swapping the nodes $k$ and $k+1$ of $T$, resulting in the tree $T''$:

$$T'' = [C_1, \ldots, C_{k-1}, C \cup E, A \cup B, C'_{k+2}, \ldots, C'_{n-1}]$$

We can then preform an HSPR move on $T''$, moving the subtree $T''|_C$ to become sibling of $T''|_D$, which gives us the tree $\hat{R}$:

$$\hat{R} = [C_1, \ldots, C_{k-1}, C \cup D, A \cup B, \hat{C}_{k+2}, \ldots, \hat{C}_{n-1}]$$

with

$$\hat{C}_m = \begin{cases} C'_m \setminus C & \text{if } C \subset C'_m \text{ and } D \not\subset C'_m \\ C'_m \cup C & \text{if } D \subset C'_m \\ C'_m & \text{otherwise} \end{cases}$$

for all $m = k+2, \ldots, n-1$.

To show that $\hat{C}_m = C_m$ for all $m = k+2, \ldots, n-1$, we distinguish whether $C \subset C_m$ or $C \not\subset C_m$. Note that since $C \cup D$ is the cluster induced by node $k$ in $T$, it is $C \subset C_m$ if and only if $D \subset C_m$ and since $D$ is not removed from any clusters between $T'$ and $T$, also $D \subset C_m$ if and only if $D \subset C'_m$ for all $m = k+2, \ldots, n-1$.

If $C \subset C_m$ and $E \not\subset C_m$, then $C'_m = C_m \setminus C$ and since $D \subset C'_m$, $\hat{C}_m = C'_m \cup C$, resulting in $\hat{C}_m = C_m$. If on the other hand $C \subset C_m$ and $E \subset C_m$, then $C'_m = C_m \cup C$ and with $D \subset C'_m$, $\hat{C}_m = C'_m \cup C$, resulting in $\hat{C}_m = C_m$.
If $C \not\subset C_m$ and $E \subset C_m$, then $C'_m = C_m \cup C$ and since $D \not\subset C'_m$, $\hat{C}_m = C'_m \setminus C$, resulting in $\hat{C}_m = C_m$. If on the other side $C \not\subset C_m$ and $E \not\subset C_m$, then $C'_m = C_m$ and $\hat{C}_m = C'_m$, resulting in $\hat{C}_m = C_m$.
In any case, it is $\hat{C}_m = C_m$ for all $m = k+2, \ldots, n-1$ and therefore $\hat{R} \simeq R$, which implies that $p'$ is a path from $T$ to $R$ that consists of a rank move followed by an HSPR move.

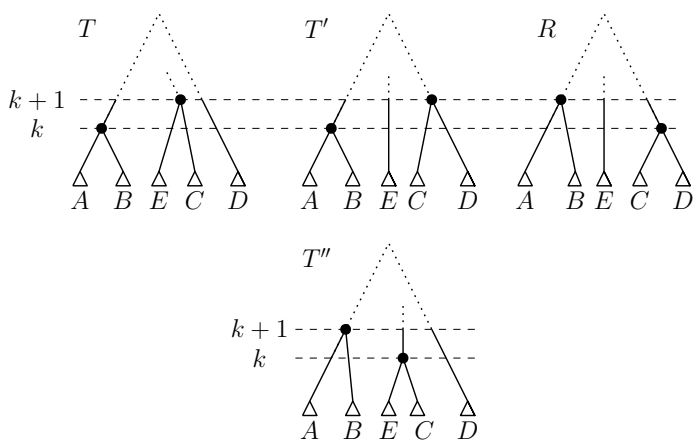


Figure 12: Trees $T$, $T'$, and $R$ on $p$ at the top, and alternative path $p'$ from $T$ to $R$ via $T''$ at the bottom if $T$ and $T'$ are connected by an HSPR move at rank $k$ and the nodes of rank $k$ and $k+1$ in $T$ are not connected by an edge. The dotted parts of the trees might contain further nodes and leaves.

In all cases above, we found an alternative path $p'$ to the path $p$ so that $p'$ consists of either two HSPR moves or a rank move followed by an HSPR move, which concludes the proof of this lemma. □

Using Lemma 5, we can now prove that we can change the order of moves on an RSPR path so that in the beginning we have a sequence of rank moves, followed by a sequence of HSPR moves.

**Theorem 10.** *In* RSPR *there is always a shortest path between two trees that has a sequence of rank moves (if there are any) at the beginning followed by only* HSPR *moves.*

*Proof.* To prove the theorem we assume that there is a shortest path $p = [T_0, T_1, \ldots, T_d]$ between trees $T_0$ and $T_d$ that has at least one rank move preceded by an HSPR move. Let $T_{i-1}$ and $T_i$ for some $1 < i < d-1$ be connected by the first HSPR move on $p$ that has a

rank move following it, i.e. $T_i$ and $T_{i+1}$ are connected by a rank move. By Lemma 5, we can replace $T_i$ by a tree $T_i'$ so that there is a rank move or HSPR move between $T_{i-1}$ and $T_i'$ and an HSPR move between $T_i'$ and $R$. When iteratively applying this procedure to all rank moves that are preceded by an HSPR move on $p$, we receive a path from $T$ to $R$ that has all rank moves at the beginning of the path, followed by a sequence of only HSPR moves. □

Theorem 10 implies that any (not necessarily shortest) path between two trees $T$ and $R$ can be converted into a $T$-$R$-path of the same length that has all rank moves bundled at the beginning of the path, followed by HSPR moves. We can also shuffle moves on an RSPR path in the opposite way so that HSPR moves happen first and rank moves last. Additionally, we can infer the following from Theorem 10.

**Corollary 5.** *Let $T_u$ and $R_u$ be rooted (unranked) trees. Then there are ranked trees $T = (T_u, \mathrm{rank}_T)$ and $R = (R_u, \mathrm{rank}_R)$ so that $d_{\mathrm{RSPR}}(T, R) = d_{\mathrm{HSPR}}(T, R)$.*

*Proof.* Let $\widehat{\mathrm{rank}}_T$ be an arbitrary rank function of $T_u$, i.e. $T' = (T_u, \widehat{\mathrm{rank}}_T)$ is a ranked tree, and let $\mathrm{rank}_R$ be a rank function on $R_u$ so that $R = (R_u, \mathrm{rank}_R)$ is a ranked tree. By Theorem 10, we can find a shortest path $p$ from $T'$ to $R$ in RSPR that has all rank moves bundled in the beginning of the path. Let $T$ be the ranked tree on $p$ after this sequence of rank moves, i.e. the remainder of $p$ between $T$ and $R$ consists of HSPR moves only. Since there is a path from $T'$ to $T$ consisting of rank moves only, the unranked versions of $T'$ and $T$ are identical, i.e. $T = (T_u, \mathrm{rank}_T)$ for some rank function $\mathrm{rank}_T$ on $T_u$. Furthermore, as $p$ is a shortest path in RSPR, the part of $p$ between $T$ and $R$ is a shortest path between those trees in RSPR. And because this part of $p$ consists of HSPR moves only, it must also be a shortest path in HSPR. We conclude that there are ranked trees $T = (T_u, \mathrm{rank}_T)$ and $R = (R_u, \mathrm{rank}_R)$ so that $d_{\mathrm{RSPR}}(T, R) = d_{\mathrm{HSPR}}(T, R)$. □

**Corollary 6.** *If there is a shortest path between trees $T$ and $R$ in* RSPR *that contains a caterpillar tree, then there is a shortest path from $T$ to $R$ that consists of* HSPR *moves only and $d_{\mathrm{HSPR}}(T, R) = d_{\mathrm{RSPR}}(T, R)$.*

*Proof.* Let $T$ and $R$ be trees connected by a shortest path $p$ that contains a caterpillar tree $T_c$. The sequence of trees on $p$ between $T$ and $T_c$ is a shortest path between these trees, and its reversed order a shortest path from $T_c$ to $T$. Applying Theorem 10 to this path from $T_c$ to $T$ gives a shortest path where all rank moves are at the beginning of the path. Since $T_c$ is a caterpillar tree, there cannot be rank moves on $T_c$ which implies that there is a shortest paths from $T$ to $T_c$ that consists of HSPR moves only.

By the same argument, the restriction of $p$ from $T_c$ to $R$ can be transformed into a path of the same length that consists of only HSPR moves.

We can concatenate the shortest path from $T$ to $T_c$ with only HSPR moves and the one from $T_c$ and $R$ with only HSPR moves and receive a path from $T$ to $R$ that has the same length as $p$. This path is hence a shortest path and consists of HSPR moves only. □

## 5 Adding leaves

In this section we consider how adding a leaf to two trees can change their distance. Especially when considering data sets of ongoing evolutionary processes, like virus transmissions, it would be ideal to re-use the already inferred tree, and some methods to do this already exist [6, 13, 15, 17]. For analyses where new leaves are added to an already existing tree, it is important to understand how adding a leaf can change a tree. It is also of interest to know how the distance between two trees changes when a new leaf is added. One would naturally assume that the addition of a leaf to two trees increases their distance. For HSPR and RSPR, however, we find that adding one leaf can decrease the distance between two trees linearly in $n$ (Theorem 11). We will see that this is the case when two trees are similar and the new leaf is added at different heights in the tree. Such leaves with varying positions in a tree are often referred to as "rogue taxa" [1], and our results here show that the existence of these can have a big impact on the distance between two trees under RSPR and HSPR.

Before we describe how adding a leaf results in a decrease in distance, we need the following observation.

**Lemma 6.** *The caterpillar trees*

$$T = [\{l_1, l_2\}, \{l_1, l_2, l_3\}, \ldots, \{l_1, l_2, l_3, \ldots, l_{n-1}\}, \{l_1, l_2, \ldots, l_n\}] \text{ and}$$
$$R = [\{l_2, l_3\}, \{l_2, l_3, l_4\}, \ldots, \{l_2, l_3, l_4, \ldots, l_n\}, \{l_1, l_2, \ldots, l_n\}]$$

*have* HSPR *and* RSPR *distance greater than or equal to* $\lceil \frac{n-1}{2} \rceil$.

The trees $T$ and $R$ of Lemma 6 are displayed in the top row of Figure 13. Whether the bound for the distance of $T$ and $R$ in Lemma 6 is sharp remains an open question.

*Proof.* The parents of $n-1$ leaves $l_1, l_3, l_4, \ldots, l_n$, i.e. all leaves except $l_2$, have different ranks in $T$ and $R$. By Lemma 1 it follows $d_{\text{HSPR}}(T, R) \geq \lceil \frac{n-1}{2} \rceil$. And as the HSPR and RSPR distance between caterpillar trees coincides (Corollary 6), $d_{\text{RSPR}}(T, R) \geq \lceil \frac{n-1}{2} \rceil$. □

Note that Lemma 6 implies that the diameter of RSPR is greater than or equal to $\lceil \frac{n-1}{2} \rceil$, which is the same boundary we already found for HSPR in Theorem 6. We are now ready to prove the main theorem of this section.

**Theorem 11.** *Adding a leaf to two trees can decrease their* HSPR *distance by* $\lceil \frac{n-1}{2} \rceil - 1$.

*Proof.* Let $T$ and $R$ be the trees of Lemma 6. We add a leaf $l_{n+1}$ to these trees $T$ and $R$ as follows (see Figure 13): A new root is added to $T$ so that the resulting tree $T'$ has $l_{n+1}$ and the old root of $T$ as children. In $R$ we attach $l_{n+1}$ as sibling of $l_1$ so that their parent

has rank one, and the ranks of all other internal nodes of $R$ are increased by one, giving us a tree $R'$ on $n+1$ leaves. The cluster representations of $T'$ and $R'$ are:

$$T' = [\{l_1, l_2\}, \{l_1, l_2, l_3\}, \ldots, \{l_1, l_2, \ldots, l_n\}, \{l_1, l_2, \ldots, l_n, l_{n+1}\}] \text{ and}$$
$$R' = [\{l_1, l_{n+1}\}, \{l_2, l_3\}, \{l_2, l_3, l_4\}, \ldots, \{l_2, l_3, l_4, \ldots, l_n\}, \{l_1, l_2, \ldots, l_n, l_{n+1}\}].$$

Let $C_i$ and $C'_i$ be the cluster induced by the node $i$ in $T'$ and $R'$, respectively, for all $i = 1, \ldots, n-1$. We can then describe the difference between the cluster representations of $T'$ and $R'$ as follows: $C'_1 = \{l_1, l_{n+1}\}$ and for all $m = 2, \ldots, n$:

$$C'_m = \begin{cases} C_m \setminus \{l_1\} & \text{if } \{l_1\} \subset C_m \text{ and } \{l_{n+1}\} \not\subset C_m \\ C_m \cup \{l_1\} & \text{if } \{l_{n+1}\} \subset C_m \\ C_m & \text{otherwise.} \end{cases}$$

By Theorem 3, $T'$ and $R'$ are connected by an HSPR move.

Therefore, adding one leaf to $T$ and $R$ changes the distance from $d_{\text{HSPR}}(T, R) \geq \lceil \frac{n-1}{2} \rceil$ (Lemma 6) to $d_{\text{HSPR}}(T', R') = 1$, which means that their distance decreases by at least $\lceil \frac{n-1}{2} \rceil - 1$. □

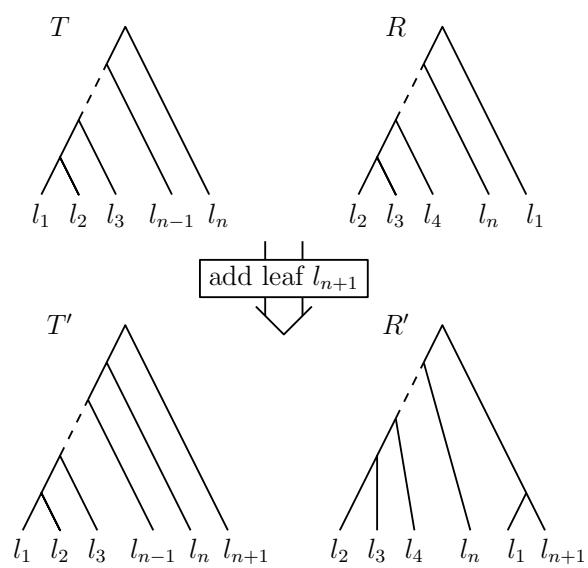


Figure 13: Top: trees $T$ and $R$ with HSPR distance greater than or equal to $\frac{n-1}{2}$ (Lemma 6). Bottom: Adding leaf $l_{n+1}$ to both $T$ and $R$ results in trees $T'$ and $R'$ that are connected by one HSPR move that moves $l_1$.

In Theorem 12 we will see that adding a leaf can only increase the HSPR distance if the unranked versions of the given trees are connected by a rooted (unranked) SPR move.

**Theorem 12.** *Let* $T = (T_u, \text{rank}_R)$ *and* $R = (R_u, \text{rank}_R)$ *be trees with* HSPR *distance* $d_{\text{HSPR}}(T, R) = d > 1$*, and let* $T'$ *and* $R'$ *be trees resulting from adding a leaf* $x$ *to* $T$ *and* $R$*, respectively, so that* $d_{\text{HSPR}}(T', R') = 1$*.*

*If* $T_u$ *and* $R_u$ *are the unranked versions of* $T$ *and* $R$*, respectively, then the rooted (unranked)* SPR *distance of* $T_u$ *and* $R_u$ *is one:* $d_{\text{SPR}}(T_u, R_u) = 1$*.*

*Proof.* Since $T'$ and $R'$ have HSPR distance one, there is a subtree $T|_i$ present in $T'$ and $R'$ that is moved between them. It follows directly from the definitions of SPR and HSPR moves that if $T'$ and $R'$ are connected by an HSPR move, their unranked versions $T'_u$ and $R'_u$ are connected by an SPR move, too, and the subtree moving between $T'_u$ and $R'_u$ is the unranked version $T|_i^u$ of $T|_i$.

To show $d_{\text{SPR}}(T_u, R_u) = 1$, we compare the distance of $T_u$ and $R_u$ with that of $T'_u$ and $R'_u$ and analyse how removing leaf $x$ from $T'_u$ and $R'_u$ to obtain $T_u$ and $R_u$ affects their distance. We know that $T'_u$ and $R'_u$ differ only in the position of subtree $T|_i^u$. We consider following cases, depending on the position of the leaf $x$ relative to $T|_i^u$:

(i) If $T|_i^u$ has $x$ as its only leaf, then $T_u \simeq R_u$, contradicting the assumptions of the theorem.

(ii) If $x$ is not in $T|_i^u$, then $T_u$ and $R_u$ differ only by the position of $T|_i^u$.

(iii) If $x$ is not the only leaf in the leaf set of $T|_i^u$, then $T_u$ and $R_u$ differ only by the position of $T|_i^u - x$.

While case (i) is not possible under our assumptions, case (ii) and (iii) both imply that $T_u$ and $R_u$ are connected by one SPR move. Therefore, we conclude that $d_{\text{SPR}}(T_u, R_u) = 1$. □

# 6 Discussion

Many tree inference methods use SPR moves for tree proposals, and extensive research made the SPR treespace usable for analysing tree inference methods and their output. There have however not been any comparable studies for phylogenetic time trees, even though time trees are inferred from sequence data in many applications and software packages like BEAST2 [7] for time tree inference are extremely popular. One approach to considering SPR moves for time trees is that by Song [32], where SPR moves are allowed to re-attach subtrees at the same height or closer to the root than its previous attachments. With this paper, we introduce two ranked SPR treespaces, RSPR and HSPR, that are motivated by the variations of SPR moves that are actually used as tree proposals in practice and have some properties in common with rooted (unranked) SPR. We show that all of these treespaces are connected, have neighbourhood sizes quadratic in the number of leaves $n$, and have a diameter linear in $n$. These properties have already been proven to be useful for SPR, which provides a good range of different trees for tree proposals [36]. Adding

ranks to SPR provides an even more biologically relevant distance measure, as it ensures that the times of nodes in subtrees cannot change by one HSPR move, which particularly makes sense when using the number of SPR moves as a proxy for the number of reticulation events such as hybridisation, recombination, or horizontal gene transfer. Furthermore, HSPR moves provide a wider range of trees at close distance than for example RNNI moves, which is especially useful for tree proposals. These observations demonstrate that studying ranked SPR treespaces may provide insights to better understanding phylogenetic methods for time trees.

We also find some interesting differences between ranked and unranked SPR spaces. One of those is the absence of the (weak and strong) cluster property in ranked SPR. This suggests that the work on Maximum Agreement Forests (MAFs) cannot be transferred to ranked SPR spaces (see [37] for a discussion on MAF-like problems). Since MAFs are essential in the proofs of $\mathcal{NP}$-hardness for classic SPR, a different strategy is needed to prove the complexity of computing distances in RSPR and HSPR. Note that it is currently not known whether this problem is $\mathcal{NP}$-hard in the ranked SPR treespaces.

We did however obtain some results on properties for shortest paths in HSPR and RSPR. The order of moves on shortest paths in RSPR can be changed so that first rank moves and then HSPR moves are performed, which suggests that the complexity of computing distances in the two spaces is the same. Furthermore, we found that there is a shortest path on which the ranks of HSPR moves increases monotonically. Leveraging these results in ranked SPR spaces might lead to developing an algorithm for computing or approximating distances in these treespaces. Another characteristic of ranked SPR spaces that makes them stand out from known tree rearrangement based treespaces is that adding one leaf to two trees can decrease their distance linearly in $n$. This is very interesting behaviour, as it has not been observed before in any of the known tree rearrangement based treespaces, and seems to be related to the presence of rogue leaves. It is worthwhile investigating the influence of this effect on tree inference algorithms, especially when considering algorithms that aim to add new sequence data to existing phylogenies (online algorithms).

This paper provides the definition and first analysis of spaces of ranked trees to enable studying time tree inference methods in the same way as untimed tree inference methods. One important open question is that of the complexity of computing distances in RSPR and HSPR treespace. A first step could be to establish that the complexity of computing distances is the same for these two ranked treespaces. Furthermore, the exact diameter of HSPR and RSPR space is still to be determined. We can use our computations [8] to show that the diameter of HSPR follows the formula $\lfloor \frac{3}{2}(n-2) \rfloor$ for $n \leq 7$, but whether this is true for any $n$ remains an open question. Another very important next step for research on ranked SPR is developing algorithms, ideally fixed-parameter tractable ones, to calculate or approximate distances. This would facilitate leveraging our newly defined treespaces to allow analysing BEAST2 [7] output as it has been done for MrBayes output with unranked SPR [36, 38].

## References


[1] AJ Aberer, D Krompass, and A Stamatakis. "Pruning Rogue Taxa Improves Phylogenetic Accuracy: An Efficient Algorithm and Webservice". *Systematic Biology* 62.1 (2013), pp. 162–166. DOI: 10.1093/sysbio/sys078.

[2] BL Allen and M Steel. "Subtree Transfer Operations and Their Induced Metrics on Evolutionary Trees". *Annals of Combinatorics* 5.1 (2001), pp. 1–15. DOI: 10.1007/s00026-001-8006-8.

[3] JM Alves, S Prado-López, JM Cameselle-Teijeiro, and D Posada. "Rapid evolution and biogeographic spread in a colorectal cancer". *Nature Communications* 10.11 (2019), p. 5139. DOI: 10.1038/s41467-019-12926-8.

[4] R Atkins and C McDiarmid. "Extremal Distances for Subtree Transfer Operations in Binary Trees". *Annals of Combinatorics* 23.1 (2019), pp. 1–26. DOI: 10.1007/s00026-018-0410-4.

[5] M Bordewich and C Semple. "On the Computational Complexity of the Rooted Subtree Prune and Regraft Distance". *Annals of Combinatorics* 8.4 (2005), pp. 409–423. DOI: 10.1007/s00026-004-0229-z.

[6] R Bouckaert, L Collienne, and A Gavryushkin. *Online Bayesian Analysis with BEAST 2*. Pages: 2022.05.03.490538 Section: New Results. 2022. DOI: 10.1101/2022.05.03.490538.

[7] R Bouckaert, J Heled, D Kühnert, T Vaughan, CH Wu, D Xie, MA Suchard, A Rambaut, and AJ Drummond. "BEAST 2: A Software Platform for Bayesian Evolutionary Analysis". *PLOS Computational Biology* 10.4 (2014), e1003537. DOI: 10.1371/journal.pcbi.1003537.

[8] L Collienne. *rankedSPR*. https://github.com/bioDS/rankedSPR. 2023.

[9] L Collienne, K Elmes, M Fischer, D Bryant, and A Gavryushkin. "Discrete coalescent trees". *Journal of Mathematical Biology* 83.5 (2021), p. 60. DOI: 10.1007/s00285-021-01685-0.

[10] L Collienne and A Gavryushkin. "Computing nearest neighbour interchange distances between ranked phylogenetic trees". *Journal of Mathematical Biology* 82.1 (2021), p. 8. DOI: 10.1007/s00285-021-01567-5.

[11] B DasGupta, X He, T Jiang, M Li, J Tromp, and L Zhang. "On computing the nearest neighbor interchange metric". DIMACS Series in Discrete Mathematics and Theoretical Computer Science 55 (2000). DOI: 10.1090/dimacs/055.

[12] Y Ding, S Grünewald, and PJ Humphries. "On agreement forests". *Journal of Combinatorial Theory, Series A* 118.7 (2011), pp. 2059–2065. DOI: 10.1016/j.jcta.2011.04.013.

[13] V Dinh, AE Darling, and FA Matsen IV. "Online Bayesian Phylogenetic Inference: Theoretical Foundations via Sequential Monte Carlo". *Systematic Biology* 67.3 (2018), pp. 503–517. DOI: 10.1093/sysbio/syx087.

[14] AJ Drummond and A Rambaut. "BEAST: Bayesian evolutionary analysis by sampling trees". *BMC Evolutionary Biology* 7.1 (2007), p. 214. DOI: 10.1186/1471-2148-7-214.

[15] M Fourment, BC Claywell, V Dinh, C McCoy, FA Matsen IV, and AE Darling. "Effective Online Bayesian Phylogenetics via Sequential Monte Carlo with Guided Proposals". *Systematic Biology* 67.3 (2018). Ed. by E Suskso, pp. 490–502. DOI: 10.1093/sysbio/syx090.

[16] A Gavryushkin, C Whidden, and FA Matsen. "The combinatorics of discrete timetrees: theory and open problems". *Journal of Mathematical Biology* 76.5 (2018), pp. 1101–1121. DOI: 10.1007/s00285-017-1167-9.

[17] MS Gill, P Lemey, MA Suchard, A Rambaut, and G Baele. "Online Bayesian Phylodynamic Inference in BEAST with Application to Epidemic Reconstruction". *Molecular Biology and Evolution* 37.6 (2020), pp. 1832–1842. DOI: 10.1093/molbev/msaa047.

[18] S Guindon, JF Dufayard, V Lefort, M Anisimova, W Hordijk, and O Gascuel. "New algorithms and methods to estimate maximum-likelihood phylogenies: assessing the performance of PhyML 3.0". *Systematic Biology* 59.3 (2010), pp. 307–321. DOI: 10.1093/sysbio/syq010.

[19] G Hickey, F Dehne, A Rau-Chaplin, and C Blouin. "SPR Distance Computation for Unrooted Trees". *Evolutionary Bioinformatics* 4 (2008), EBO.S419. DOI: 10.4137/EBO.S419.

[20] S Höhna, M Defoin-Platel, and AJ Drummond. "Clock-constrained tree proposal operators in Bayesian phylogenetic inference". *2008 8th IEEE International Conference on BioInformatics and BioEngineering*. Athens, Greece: IEEE, 2008, pp. 1–7. DOI: 10.1109/BIBE.2008.4696663.

[21] S Höhna and AJ Drummond. "Guided tree topology proposals for Bayesian phylogenetic inference". *Syst. Biol.* 61.1 (2012), pp. 1–11.

[22] S Höhna, MJ Landis, TA Heath, B Boussau, N Lartillot, BR Moore, JP Huelsenbeck, and F Ronquist. "RevBayes: Bayesian Phylogenetic Inference Using Graphical Models and an Interactive Model-Specification Language". *Systematic Biology* 65.4 (2016), pp. 726–736. DOI: 10.1093/sysbio/syw021.

[23] JP Huelsenbeck and F Ronquist. "MRBAYES: Bayesian inference of phylogenetic trees". *Bioinformatics* 17.8 (2001), pp. 754–755. DOI: 10.1093/bioinformatics/17.8.754.

[24] LA Hug et al. "A new view of the tree of life". *Nature Microbiology* 1.55 (2016), pp. 1–6. DOI: 10.1038/nmicrobiol.2016.48.

[25] G Jones. "Algorithmic improvements to species delimitation and phylogeny estimation under the multispecies coalescent". *J. Math. Biol.* 74.1-2 (2017), pp. 447–467.

[26] S Linz and C Semple. "A Cluster Reduction for Computing the Subtree Distance Between Phylogenies". *Annals of Combinatorics* 15.3 (2011), pp. 465–484. DOI: 10.1007/s00026-011-0108-3.

[27] LT Nguyen, HA Schmidt, A von Haeseler, and BQ Minh. "IQ-TREE: A Fast and Effective Stochastic Algorithm for Estimating Maximum-Likelihood Phylogenies". *Molecular Biology and Evolution* 32.1 (2015), pp. 268–274. DOI: 10.1093/molbev/msu300.

[28] HA Ogilvie, RR Bouckaert, and AJ Drummond. "StarBEAST2 Brings Faster Species Tree Inference and Accurate Estimates of Substitution Rates". *Mol. Biol. Evol.* 34.8 (2017), pp. 2101–2114.

[29] MN Price, PS Dehal, and AP Arkin. "FastTree 2 – Approximately Maximum-Likelihood Trees for Large Alignments". *PLOS ONE* 5.3 (2010). Publisher: Public Library of Science, e9490. DOI: 10.1371/journal.pone.0009490.

[30] A Rambaut, DL Robertson, OG Pybus, M Peeters, and EC Holmes. "Phylogeny and the origin of HIV-1". *Nature* 410.68326832 (2001), pp. 1047–1048. DOI: 10.1038/35074179.

[31] YS Song. "On the Combinatorics of Rooted Binary Phylogenetic Trees". *Annals of Combinatorics* 7.3 (2003), pp. 365–379. DOI: 10.1007/s00026-003-0192-0.

[32] YS Song. "Properties of Subtree-Prune-and-Regraft Operations on Totally-Ordered Phylogenetic Trees". *Annals of Combinatorics* 10.1 (2006), pp. 147–163. DOI: 10.1007/s00026-006-0279-5.

[33] A Stamatakis. "RAxML version 8: a tool for phylogenetic analysis and post-analysis of large phylogenies". *Bioinformatics* 30.9 (2014), pp. 1312–1313. DOI: 10.1093/bioinformatics/btu033.

[34] Y Turakhia et al. "Pandemic-scale phylogenomics reveals the SARS-CoV-2 recombination landscape". *Nature* 609.79297929 (2022), pp. 994–997. DOI: 10.1038/s41586-022-05189-9.

[35] C Whidden, RG Beiko, and N Zeh. "Fixed-Parameter Algorithms for Maximum Agreement Forests". *SIAM Journal on Computing* 42.4 (2013), pp. 1431–1466. DOI: 10.1137/110845045.

[36] C Whidden and FAI Matsen. "Quantifying MCMC Exploration of Phylogenetic Tree Space". *Systematic Biology* 64.3 (2015), pp. 472–491. DOI: 10.1093/sysbio/syv006.

[37] C Whidden and FA Matsen. "Calculating the Unrooted Subtree Prune-and-Regraft Distance". *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 16.3 (2019), pp. 898–911. DOI: 10.1109/TCBB.2018.2802911.

[38] C Whidden and FA Matsen. "Ricci–Ollivier curvature of the rooted phylogenetic subtree–prune–regraft graph". *Theoretical Computer Science*. Special issue on Analytic Algorithmics and Combinatorics 699 (2017), pp. 1–20. DOI: 10.1016/j.tcs.2017.02.006.

[39] C Ye, B Thornlow, A Hinrichs, A Kramer, C Mirchandani, D Torvi, R Lanfear, R Corbett-Detig, and Y Turakhia. "matOptimize: a parallel tree optimization method enables online phylogenetics for SARS-CoV-2". *Bioinformatics* 38.15 (2022), pp. 3734–3740. DOI: 10.1093/bioinformatics/btac401.