Ranked Subtree Prune and Regraft

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

[1]School of Mathematics and Statistics, University of Canterbury, New Zealand
[2]Faculty of Computer Science, Dalhousie University, Canada

✉lena@lenacoll.de
May 15, 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. A variety of computational methods for reconstructing phylogenetic (time) trees from (typically molecular) sequence data rely on algorithms that sample the treespace. These include, for example, Bayesian inference methods that employ Markov Chain Monte Carlo (MCMC) algorithms where in every step a tree modification is proposed and the new tree is accepted if it fulfils certain criteria. The most popular tree proposals used in practice implement 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. Extensive studies of the classic SPR tree rearrangement enabled researchers to analyse phylogenetic inference methods that do not use times. However, little is known about versions of SPR for time trees, which limits comparable studies for time tree inference methods.

In this paper we consider a modification of the classical Subtree Prune and Regraft (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

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).

to known tree rearrangement based treespaces. Surprisingly, we show the counter-intuitive 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".

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 [23], analysing transmission patterns of viruses [28, 31], 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 [13], BEAST2 [7], MrBayes [22], and RevBayes [21], and maximum likelihood methods like IQ-TREE [25], PhyML [17], or RAxML [30]. All these methods rely on tree proposals: at any given step, a new tree similar to the current tree is proposed and is accepted for the next step if it fulfils certain conditions. 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 package, including maximum likelihood methods [17, 30, 27], Bayesian inference methods [13, 21, 26] and also recent parsimony-based methods that are able to infer large-scale phylogenies [36]. 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 [17, 25]. 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 [19]. 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 is $\mathcal{NP}$-hard [18, 5]. Despite this $\mathcal{NP}$-hardness result, fixed-parameter tractable algorithms for computing SPR distances exist, which can be used for analysing topological peaks of posterior distributions [33] or for computing supertrees [32, 34]. Understanding properties of the SPR treespace, which is 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 [34].

Biologically, SPR operations are motivated by processes such as hybridisation, horizontal gene transfer, or recombination, all of which result in reticulation events. The SPR distance is a lower bound to the number of reticulation events needed to reconcile two gene trees [34], which is why SPR is suitable for analysing families of gene trees. Since reticulation events require the lineages that reconcile to be present at the same time, time trees are typically used to model these processes. The classical SPR and NNI rearrangements do not take times of internal nodes into account, which in particular implies that an SPR operation can move a subtree to a different time in the tree. However, the biological processes mentioned restrict the model to force SPR moves to preserve times, that is, SPR moves can only shift a subtree horizontally.

A version of the SPR move adapted to time trees was first introduced by Höhna, Defoin-Platel, and Drummond [19], who analysed 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 [20], 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 [24]. Even though SPR moves are widely used as tree proposals and have been adapted to work for time trees, research on SPR treespaces has so far been limited to classic SPR that does not take times of evolutionary events into account.

A lot of this research has focussed on analysing the geometry of the classic SPR treespace on trees without times, leading to numerous results for SPR on rooted and unrooted trees (without timing information). 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 [18]. 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, 34], 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 [11, 4], 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 [29, 2]. The latter is an important property that has been used to determine the curvature of the treespace by Whidden and Matsen [35] in

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

order to analyse mixing properties of Markov Chain Monte Carlo methods, which are used for Bayesian tree inference. It is important to bear in mind that all of these results have been derived for trees that do not contain timing information of evolutionary events, while SPR on time trees has not yet been considered. This leaves a major gap for analysing tree inference methods using SPR for time trees.

In this paper we fill this gap by introducing SPR moves for ranked trees, where internal nodes are ordered according to times of corresponding evolutionary events. These moves require the height (i.e. rank) at which a subtree is cut and reattached to be the same, similar to the Fixed Node Prune and Regraft move introduced in Höhna, Defoin-Platel, and Drummond [19]. 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. 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 [15] for ranked trees (RNNI) that allows distances to be computed in polynomial time [9], unlike 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 has proven to be important for the polynomial time algorithm in RNNI [9] and also for MAFs in SPR [34]. The discussion of the cluster property is followed by some observations on the shape of shortest paths and how RSPR and HSPR shortest paths are related (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 Technical Introduction

A ranked tree is a rooted binary phylogenetic tree with n leaves uniquely labelled by elements of the set $\{l_1, l_2, \ldots, l_n\}$ where all internal nodes are assigned a unique *rank* in $\{1, \ldots, n-1\}$ such that no node has rank greater than its parent. Furthermore, all leaves of a ranked tree are assigned rank 0. An example of a ranked tree with annotated ranks is given on the left of Figure 1. We refer to the node of rank k in a ranked tree T as $(T)_k$.

A *cluster* C of a ranked 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 connected to two leaves, and we refer to a cherry by its cluster containing two leaves, e.g. $\{c_1, c_2\}$. If the internal node of such a cherry has rank k, we say that the

cherry $\{c_1, c_2\}$ has rank k. A *subtree* T' of a ranked 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' is *induced* by the node v and call the parent of v *parent of the subtree* T'. Given a set of leaves S and a ranked 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. For a cluster C of T, $T|_C$ contains exactly the leaves of C. We can uniquely represent a ranked tree T by a list of its clusters sorted according to increasing rank. This representation is called *cluster representation*. The leftmost ranked tree in Figure 1 for example has cluster representation $[\{l_1, l_2\}, \{l_3, l_4\}, \{l_1, l_2, l_5\}, \{l_1, l_2, l_3, l_4, l_5\}]$.

Throughout this paper we will say *tree* instead of ranked tree and assume that a tree has n leaves, unless stated otherwise.

We say that an edge *covers* a rank k if the bottom node of the edge has rank less than k and the top node has rank greater than k.

A *horizontal* SPR *move* (HSPR move) on an edge e of a ranked tree T 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) cut the edge e to obtain two subtrees T_e and T_ρ, where T_ρ has the same root as T,

(ii) suppress the resulting node of degree 2 in T_ρ (top node of edge e), whose rank we denote by r,

(iii) reattach the root of T_e to a newly introduced node v at rank r on an edge that covers rank r in T_ρ.

Since the changes done to the tree T all happen at the node with rank r, we call this an HSPR *move at rank* r. A *rank move* on a tree T swaps the ranks of two nodes $(T)_k$, $(T)_{k+1}$ with consecutive ranks that are not connected by an edge. An example of an HSPR move and a rank move can be found in Figure 1.

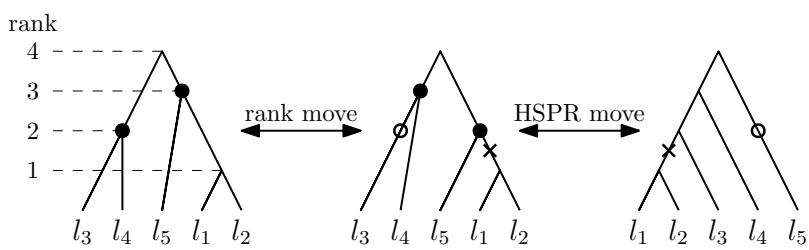

Figure 1: Rank move on the left, swapping the ranks of the highlighted nodes in the leftmost tree. On the right an HSPR move at rank 2 is illustrated, moving the subtree induced by $\{l_1, l_2\}$ by cutting the edge highlighted by a cross and re-attaching it at the edge highlighted by a circle.

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 ranked trees 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 ranked trees that are connected by an edge if one can be transformed into the other by an HSPR move.

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

A *path* between two ranked 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* $d(T, R)$ between trees T and R.

3 Basic Properties of HSPR and RSPR

The first question we want to answer is whether our newly defined treespaces are connected (Theorem 1), 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 the tree rearrangements are used for tree proposals in (MCMC) random walks, which should be able reach any tree in treespace from any starting tree. For developing and interpreting such random walks it is furthermore important to know the number of trees that result from one tree rearrangement operations as well as the maximum distance between any two trees. We establish these two properties, neighbourhood sizes (Theorem 2) and maximum distances (diameter) in RSPR and HSPR (Section 3.1) in this section. At the end of this section, we investigate the cluster property for RSPR and HSPR, explain its importance and why it is surprising that neither of the two spaces has this property (Section 3.2).

Theorem 1. *The treespaces* HSPR *and* RSPR *are connected.*

Proof. We show that any pair of ranked trees T and R are connected by a path of only HSPR moves. Since all HSPR moves are RSPR moves, it then follows that both spaces are connected.

We construct an HSPR path from T to R as follows. Let $\{c_1, c_2\}$ be the cherry induced by the node of rank one in R and let $\{a_1, a_2\}$ be the cherry induced by the node of rank one in T. Consider the following path that transforms T into a tree T_2 that has $\{c_1, c_2\}$ as cherry at rank one (see Figure 2). First, perform an HSPR move at rank one that cuts the subtree containing only leaf a_1 from T and reattaches it on the edge adjacent to c_1. In the resulting tree T_1, $\{c_1, a_1\}$ is the cherry induced by the node of rank one. In a second step, we perform an HSPR move to prune the subtree containing only c_1 from T_1 and reattach

it on the edge adjacent to c_2, resulting in a tree T_2 with cherry $\{c_1, c_2\}$ of rank one. Note that if $\{c_1, c_2\}$ and $\{a_1, a_2\}$ intersect, the trees T_1 or T_2 might be equal to T, and fewer than two steps are required to build the cherry $\{c_1, c_2\}$.

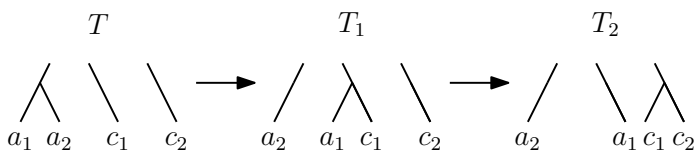

Figure 2: Path between T and T_2 as described in the proof of Theorem 1. The nodes with rank greater than one and potential further leaves are not shown in this illustration.

Now treating the cherry $\{c_1, c_2\}$ as a single leaf c in T_2 and R, we can repeat this procedure iteratively for all nodes in R (from bottom to top) to create a path from T to R. This procedure can be applied to any two ranked trees, so any two trees are connected by a sequence of HSPR moves, which proves the theorem. □

The algorithm used to prove Theorem 1 computes 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].

Because tree inference algorithms often involve sampling tree neighbourhoods, it is important to know the number of *neighbours* of a tree under a tree rearrangement, i.e. the number of trees with distance one from a given tree, as this is the number of possible tree proposals when starting at a fixed tree. In rooted and unrooted SPR the number of neighbours of a tree is quadratic in the number of leaves n [29, 2]. We now show that the number of neighbours in RSPR and HSPR is quadratic in the number of leaves as well, with the number of neighbours in RSPR naturally depending on the shape of the tree. We derive both of these numbers explicitly.

Theorem 2. *The number of neighbours of a ranked tree T with k rank intervals, i.e. pairs of nodes with consecutive ranks i and $i+1$ for some $i \in \{1, \ldots, n-2\}$ that are not connected by an edge, is $(n-1)(n-2)+k$ in* RSPR *and $(n-1)(n-2)$ in* HSPR.

Proof. The number of neighbouring trees resulting from a rank move on a ranked 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 r for $r \in \{1, \ldots, n-1\}$. As the node of rank r has two children, there are two different subtrees that can be moved by such an HSPR move. There are $n-1-r$ edges that cover rank r, excluding the one on which the node of rank r is placed in T, so there are $n-1-r$ potential reattachment edges for the subtree that has been cut. This gives us $2(n-1-r)$ RSPR neighbours of T resulting from an HSPR move at rank r. And because HSPR moves can happen at any rank between 1

and $n-1$, the number of RSPR moves possible on T is:

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

Since all rank moves and HSPR moves result in different trees, the tree 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, $\max_{T,R} d(T,R)$, is called the *diameter* of the treespace. When measuring the similarity of trees using a distance metric, knowing the maximum possible distance between trees is essential to be able to interpret distances. The diameter of both unrooted and rooted SPR space is $n-\Theta(\sqrt{n})$ [11, 4]. We will see in this section that the diameters of HSPR space and RSPR space are linear in the number of leaves n by establishing lower and upper bounds.

Corollary 1. *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 1 to compute a path between any two trees T and R and use its length as upper bound on the diameter. This algorithm uses a bottom-up approach that, starting at tree T, constructs in iteration $i = 1, \ldots, n-2$ the cluster induced by the node of rank i in R. After at most $n-2$ iterations, we receive the destination tree R after at most $2(n-2)$ HSPR moves. The resulting path is valid in both HSPR and RSPR space because it consists only of HSPR moves, which are allowed in both treespaces, and its maximum length is hence an upper bound of the diameter of both spaces. □

It is important to note that the algorithm described in the proof of Theorem 1 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 ranked trees in HSPR.

Lemma 1. *Let T and R be two trees and let x be the number of leaves whose parents have different ranks in T and R. Then $d_{\text{HSPR}}(T,R) \geq \frac{x}{2}$.*

Proof. HSPR moves can change the rank of at most three parents of nodes. The most general case of HSPR moves is illustrated in Figure 3 where the subtree induced by C, which can be a cluster or a single leaf, is moved. Using the notation of that figure, the only nodes whose parents change the displayed HSPR move are the root of A_1, the root of B_2, and the node of rank i:

$$\text{rank}(\text{parent}(A_1)) = i \to j$$
$$\text{rank}(\text{parent}(B_2)) = k \to i$$
$$\text{rank}(\text{parent}(i)) = j \to k$$

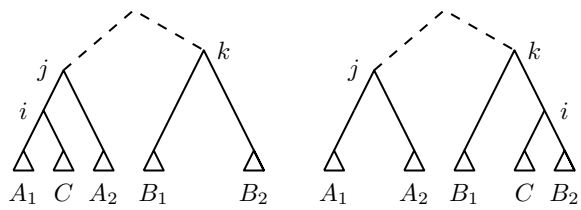

Figure 3: HSPR move pruning the subtree induced by cluster C, moving it onto the edge above the subtree induced by B_2. A_1, A_2, B_1, B_2, and C can be clusters or single leaves. Dashed edges represent parts of the tree that potentially contain further nodes.

As i is always an internal node, an HSPR move can change at most two parents of leaves (here these leaves would be A_1 and B_2). On a path between trees T and R, the parents of all leaves of T need to be updated to match those in R. Since between T and R, x parents of leaves need to change rank and in every move the rank of at most two parents of leaves changes, at least $\frac{x}{2}$ HSPR moves are needed to transform T into R. □

The lower bound given in Lemma 1 can be tight. For example, the two leftmost trees in Figure 4 are connected by one HSPR move and the parents of $x = 2$ leaves (a_2 and a_5) have different ranks in the two trees.

Theorem 3. *There are trees T and R with distance $d_{\text{HSPR}}(T, R) \geq \frac{n}{2}$.*

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, $i \in \{5, \ldots, n\}$, the rank of the parent also is different in T than in R: $\text{rank}(\text{parent}_T(l_i)) = i - 1 \neq i - 2 = \text{rank}(\text{parent}_R(l_i))$. Therefore, the parents of all n leaves have different ranks in T and R, and by Lemma 1 it follows $d_{\text{HSPR}}(T, R) \geq \frac{n}{2}$. □

Therefore, the diameter of the HSPR space is linear in n. Note that we will see later that by Corollary 5, Theorem 3 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 has the same cluster C, we say that this path p *preserves* the cluster C. We moreover say that a treespace has the *cluster property*, if for two trees containing the same cluster C, every tree on every shortest path between them contains C as cluster. 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. The classic unranked rooted version of SPR space has this property, which is essential for the proof that computing SPR distances is $\mathcal{NP}$-hard [5]. We will see in the following that neither RSPR nor HSPR space has the cluster property. This observation is important as it suggests that the proving technique for $\mathcal{NP}$-hardness for unranked SPR, which uses maximum agreement forests, cannot be used for ranked trees.

Theorem 4. HSPR *space and* RSPR *space do not have the cluster property.*

Proof. We prove this theorem by showing that the two trees T and R depicted in Figure 4 have no shortest path between them that preserves their shared cluster $\{l_4, l_5\}$ in HSPR or RSPR. On any path between T and R preserving the cluster $\{l_4, l_5\}$, the rank of the node inducing this cluster needs to decrease from 3 in T to 1 in R while not changing the cluster. Note that originating from T, no HSPR move can decrease the rank of this cluster without breaking it, so 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 cluster property.

In RSPR, at least two rank moves are necessary to move the node inducing $\{l_4, l_5\}$ to rank 1 without breaking it. Since the tree after these two rank moves is not R, at least one additional HSPR move is then needed to receive R, which means that there is no path in RSPR that preserves the cluster $\{l_4, l_5\}$ and is (strictly) shorter than 3.

There is however a path from T and R with only two HSPR moves, where first the leaf l_1 is moved to become a sibling of l_5 and then l_5 is moved to become a sibling of l_4 (see Figure 4). Since this path is shorter than any path in RSPR preserving the cluster $\{l_4, l_5\}$, the RSPR space does not have the cluster property.

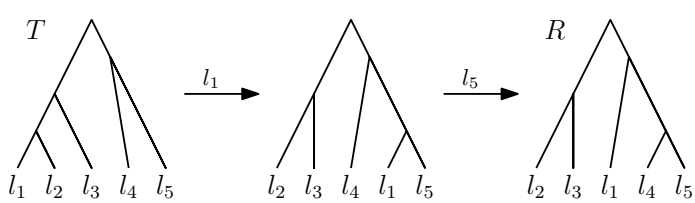

Figure 4: Trees T and R that give the counterexample to the cluster property in RSPR and HSPR in Theorem 4. The labels of the arrows indicate the nodes of the subtrees that are pruned in the corresponding HSPR move.

□

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 distances [5, 18] as well as NNI distances [10]. A recent modification of NNI tree rearrangements for ranked trees has however shown to be computationally tractable [9]. Whether the same is true for ranked SPR distances is a natural question, and we approach it by investigating the structure if shortest paths in RSPR and HSPR treespace.

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 5). 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 6). Then, by further generalising this result, we show in Corollary 3 that for two trees with identical clusters up to a certain rank, all shortest paths between them preserve these clusters.

To prove that there is a shortest path in HSPR on which the 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. Let $p = [T, T', R]$ be a path as described in the lemma, let T_s^1 be the subtree moved between T and T', and let T_s^2 be the subtree moved between T' and R. T_s^2 is subtree of T' and R, as the HSPR move between these trees moves exactly this subtree. It is also a subtree of T, because the HSPR move between T and T' is at rank $k > i$ and does hence not modify T_s^2, which is rooted at the node of rank i.

We now define T'' as the tree resulting from cutting the subtree T_s^2 from T and reattaching it at the position it has in R, i.e. T and T'' are connected by an HSPR move at rank i. It remains to show that T'' and R are connected by an HSPR move at rank k. An illustration of the idea of the following part of the proof can be found in Figure 5.

Let us consider the trees that result from deleting the subtree T_s^2 from T, T', and R and suppressing the resulting degree two node at rank i and subtracting one from all ranks greater than i in the resulting trees. Since T' and R only differ by the position of T_s^2, removing T_s^2 from T' and R results in the same tree $\bar{R}$. The tree $\bar{T}$ that we receive by suppressing T_s^2 from T is connected to $\bar{R}$ by an HSPR move, which corresponds to the

HSPR move between T and T'. Adding T_s^2 back to $\bar{T}$ at the position that it has in R gives us a tree T''. T'' is hence connected to R by an HSPR move at rank k, which is the same move as the one between $\bar{R}$ and $\bar{T}$. Furthermore, suppressing T_s^2 from T'' and T gives us the same tree $\bar{T}$, so T'' and T are connected by an HSPR move at rank i that moves T_s^2.

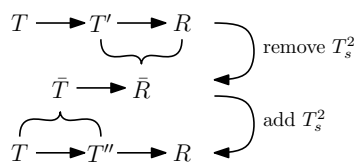

Figure 5: Illustration of the subtree removal and addition of T_s^2 as described in the proof of Lemma 2. Deleting T_s^2 from T' and R results in $\bar{R}$, while deleting T_s^2 from T and T'' results in $\bar{T}$.

We can hence build a path from T via T'' to R that has first an HSPR move at rank i and then an HSPR move at rank k. □

Theorem 5. *For any two trees T and R there is always a shortest* HSPR *path 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, we need the following two lemmas, which describe how performing multiple HSPR moves at a fixed rank r along a shortest path effects the distance between two trees. These lemmas are interesting on their own as they are informative about the local geometry of the HSPR treespace.

Lemma 3. *Let T and R be trees that are identical except for the positions of d subtrees $T_s^1, T_s^2, \ldots, T_s^d$, as described in the following. For $r = \min_{i=1,\ldots,d} \text{rank}(\text{parent}(T_s^i))$, we assume that the edges above the roots of all except one of the subtrees $T_s^1, T_s^2, \ldots, T_s^d$ cover the rank r. We further assume that positions of these subtrees change between T and R in the following way: Each subtree T_s^i in R is at the position that T_s^{i+1} has in T for all $i = 1, \ldots, d-1$ and T_s^d in R is at the position that T_s^1 has in T.*

Then there is a path from T to R that has d HSPR *moves at rank r, i.e. $d(T, R) \leq d$.*

Proof. Let T_s^i be the subtree with parent of minimum rank r among all subtrees $T_s^1, \ldots, T_s^d$. By the assumptions of the lemma, all edges above the roots of T_s^j (for all $j = 1, \ldots, d$ and $j \neq i$) in T cover rank r.

We can now create a path p of length d from T to R. Starting at T, an HSPR move at rank r moves the subtree T_s^i so that T_s^i and T_s^{i+1} become siblings. This move is possible, because the edge above the root of T_s^{i+1} covers the rank r. Note that such an HSPR move at rank r can reattach a subtree onto the edge above any of the subtrees T_s^j for $j \neq i$.

After the initial move, we continue by iterating through $j = i+1, \ldots, d-1$ and moving each T_s^j to become sibling of T_s^{j+1} in every iteration j. After these iterations we perform an HSPR move that cuts T_s^d to become sibling of T_s^1 and continue as before by moving T_s^j with $j = 1, \ldots, i-2$ until we reach a tree where T_d^{i-2} is sibling of T_d^{i-1}. In the last move we move T_d^{i-1} to the position that T_d^i had in T. Note that the resulting tree is R, as it is the tree that equals T except for the permutation of subtrees $(T_s^1, T_s^2, \ldots, T_s^d)$.

Every T_d^j is moved exactly once to reach its final position, which is the one it has in R. Therefore, there are exactly d moves required on this path p from T to R, which implies $d(T, R) \leq d$. □

The difference between trees T and R in Lemma 3 can be interpreted as a permutation of the positions of subtrees $(T_s^1, T_s^2, \ldots, T_s^d)$ (in cycle notation).

Another interesting property of shortest paths is that of the next lemma, Lemma 4: No subtree can move twice 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.

Lemma 4. *Let T and R be two trees connected by a shortest path p where all* HSPR *moves happen at the same rank r and move the subtrees $T_s^0, T_s^1, \ldots, T_s^{d-1}$ in this order, i.e. p has length d.*

Then every T_s^i is subtree of both T and R and T_s^i and T_s^j are distinct subtrees for all $i \neq j \in \{0, 1, \ldots, d-1\}$.

Proof. Since all moves on p are HSPR moves at rank r, every tree T_s^i with $i \in \{0, 1, \ldots, d-1\}$ has a root with rank less than r. This means that no edges inside these subtrees can change on p, which implies that $T_s^0, \ldots, T_s^{d-1}$ are subtrees in all trees on p, including T and R.

To prove that these subtrees are all different, we assume to the contrary that two of them are identical. We furthermore only consider the path between the trees on which the identical subtree moves twice. Let T' be the subtree moved by the first HSPR move and the last move on the path. We update the notation so that this part of the path starts at tree T and ends at R and the subtrees moved are $T_s^0, \ldots, T_s^{d-1}$ with T_s^0 and T_s^{d-1} being identical. We refer to this subtree as $T' = T_s^0 = T_s^{d-1}$. Note that it could be $d - 1 = 1$, in which case T' moves twice in two consecutive moves.

By our assumption that T' is the only subtree on p that is moved twice, every T_s^i with $i \in \{1, \ldots, d-2\}$ moves exactly once, and it moves to the position that T_s^{i+1} had in T, to allow T_s^{i+1} to be the next subtree that is moved at rank r. Let T'' be the subtree that is

sibling of T' in R. The first move on p moves T' to the position of T_s^1, and the last move on p moves T' to its position as neighbour of T'' in R. Note that T_s^{d-2} moves to the position that T' has before the last move, which is the position that T_s^1 had in T. Therefore, T_s^{d-2} ends up having the position in R that T_s^1 had in T.

We can construct an alternative path from T to R in the following way: First, move T' to become sibling of T''. This gives us a tree T_1 that is identical to R except for the positions of $T_s^1, \ldots, T_s^{d-2}$. More precisely, the positions of these subtrees are permuted by $(T_s^1, \ldots, T_s^{d-2})$, as described in Lemma 3. By Lemma 3 we can then find a path from T_1 to R with $d-2$ moves. Therefore, there is a path from T to R via T_1 that has length $d-1$ which contradicts our assumption on p being a shortest path.

Hence, there cannot be a subtree that is moved twice by an HSPR move at the same rank r on a shortest path. □

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

Theorem 6. *Let T and R be trees that have the same cherry $\{x, y\}$ at rank one. Then every tree on every shortest path between T and R has the cherry $\{x, y\}$ at rank one.*

Proof. For contradiction, we assume that trees T and R have the same cherry $\{x, y\}$ at rank one and there is a shortest path between them containing a tree with a different cherry at rank one. 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 needs to reconstruct the cherry $\{x, y\}$ at rank one, and the leaf x can only move once at rank one (Lemma 4), the last leaf that moves on p is y. We can hence assume that $x, a_1, a_2, \ldots, a_{d-1}, y$ is the sequence of distinct leaves (Lemma 4) moved on p. 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, which can only be achieved if another leaf a_{d-1} is moved to become sibling of y. The path p as described above is depicted in Figure 6.

The moves between T and R can be interpreted as a change of positions of the leaves $x, a_1, a_2, \ldots, a_{d-1}, y$ in the following way: Every leaf a_i with $i \in \{1, \ldots, d-2\}$ in R is at the position that a_{i+1} has in T. The position of a_{d-1} in R is that of the cherry $\{x, y\}$ in T and the cherry $\{x, y\}$ is moved to the position that a_1 had in T. The reason for this is that first x moves to the position of a_1 and at the end of p, y moves to become sibling of x

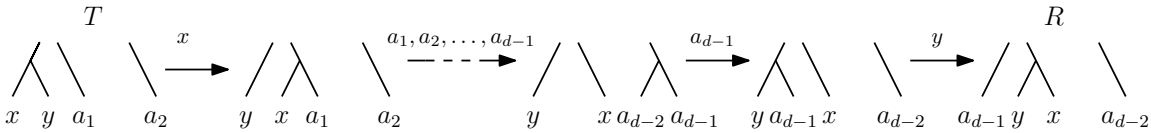

Figure 6: Trees T and R with cherry $\{x, y\}$ at rank one and all moves on a path p from T to R that does not preserve the cherry as described in the proof of Theorem 6. The labels of the arrows indicate the leaves that are pruned and reattached by the corresponding HSPR moves.

at this position. And because a_{d-1} moves to become sibling of y in the second to last step, and y did not move before this step on p, the position of a_{d-1} in R is that of $\{x, y\}$ in T.

Using this knowledge of the difference between T and R, we can now construct a path between these trees that has length less than $|p| = d + 1$, which leads to a contradiction of our assumption that a path that does not preserve a shared cherry can be a shortest path. Between T and R, the positions of the subtrees $T_s^0, T_s^1, \ldots, T_s^{d-1}$ on leaf sets $\{x, y\}, \{a_1\}, \ldots, \{a_{d-1}\}$, respectively, are permuted by $(T_s^0, \ldots, T_s^{d-1})$. All of these subtrees are rooted either in leaves or in the node of rank one, which implies that the roots of all these subtrees have rank less than $r = \min_{i=1,\ldots,d} \text{rank}(\text{parent}(T_s^i)) > 1$. We can apply Lemma 3 to see that there is a path between T and R of length d. Moreover, we can infer from the construction of the path in Lemma 3 that the subtrees $T_s^0, \ldots, T_s^{d-1}$ are moved at rank $r > 1$, which implies that the cherry $\{x, y\}$ is preserved on this path.

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

Theorem 6 implies that for 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 2). We will see in Theorem 8 that we can generally not assume that the distance stays the same or decreases when deleting a leaf from two trees.

Corollary 2. *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. Then $d_{\text{HSPR}}(T, R) = d_{\text{HSPR}}(T, R)$.*

Proof. By Theorem 6 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 gives a path between T' and R', i.e. $d(T', R') \leq d(T, R)$. If there was a path between T' and R' that was shorter than $d(T, R)$, then adding a leaf c_2 as sibling of c_1 with parent of rank one in every tree on p results in a path between T and R of length less than $d(T, R)$, which is a contradiction. Hence, $d(T, R) = d(T', R')$. □

Another observation that follows from Theorem 6 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 3. *If all clusters induced by nodes of rank less than some $i \in \{2, \ldots, n\}$ coincide in trees T and R, then every shortest path between T and R preserves these clusters.*

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].

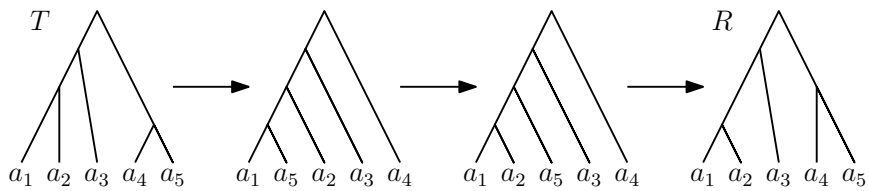

Figure 7: Shortest HSPR path between trees T and R that does not preserve the shared cluster $\{a_1, a_2, a_3\}$ at rank three. There is no shortest path in HSPR between T and R that preserves this cluster.

4.2 RSPR Shortest Paths

The RSPR space can be interpreted as an extension of HSPR in which rank moves are added and provide shortcuts between some trees. Here, we investigate how the addition of rank moves changes shortest paths. We show in Theorem 7 that we can change the order of moves on paths in RSPR, while not changing their length, so that all rank moves are right 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 5), 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.*

Proof. Let k and $k+1$ for some $k \in \{1, \ldots, n-2\}$ be the ranks of the nodes at which the rank move between T' and R is performed. We denote the clusters induced by the children of the node of rank k in T' by A and B and the clusters induced by the children of the node of rank $k+1$ in T' by C and D, as depicted in the tree T' in the top of Figure 8.

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.

   The nodes of rank k and $k+1$ are not affected by this move, so we can swap the order of the moves between T and R: 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. If there is an edge connecting the nodes with ranks k and $k+1$ in T, then $T|_A$ is sibling of either $T|_C$ or $T|_D$ in T. The reason for this is that by our assumptions on T', the subtrees $T'|_C$ and $T'|_D$ have the node of rank $k+1$ as parent (see Figure 8). We can assume that $T|_A$ and $T|_D$ are siblings in T, as depicted in Figure 8, otherwise we change notation for C and D.

   To get a path p' with the desired properties, we can perform an HSPR move on T that moves $T|_C$ to become sibling of $T|_D$, resulting in a tree T'' in which the parent of $T''|_A$ has rank $k+1$. An HSPR move that moves $T''|_A$ to become sibling of $T''|_B$ then results in R (see Figure 8).

   2.2. If there is no edge between the nodes with rank k and $k+1$ in T, then $T|_A$ has sibling $T|_E$ in T with $E \neq A, B, C, D$. We can then first perform a rank move on the nodes of rank k and $k+1$ in T to receive T''. Following this by an HSPR move moving $T''|_A$ to become sibling of $T''|_B$ results in R. A comparison of the path p between T and R and the alternative path p' from T to R via T'' is illustrated in Figure 9.

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

   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. If the node of rank k is child of the node of rank $k+1$ in T, either $T|_C$ or $T|_D$ is sibling of $T|_{A \cup B}$. We can assume without loss of generality that $T|_C$ and $T|_{A \cup B}$ are siblings, otherwise we swap notations for C and D. A path p' can now be constructed by first doing an HSPR move on T moving $T|_A$ to become sibling of $T|_C$ in the resulting tree T'', followed by an HSPR move moving $T''|_C$ to become sibling of $T''|_D$, which gives us R (as depicted in Figure 10).

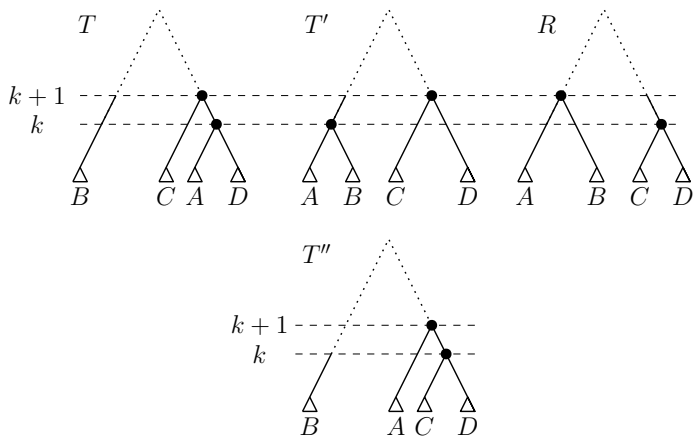

Figure 8: Trees T, T', and R on p, and alternative path p' from T to R via T'' at the bottom if T and T' are connected by an HSPR move at rank k and the nodes of rank k and $k+1$ in T are connected by an edge, as explained in case 2.1.. The dotted parts of the trees might contain further nodes and leaves.

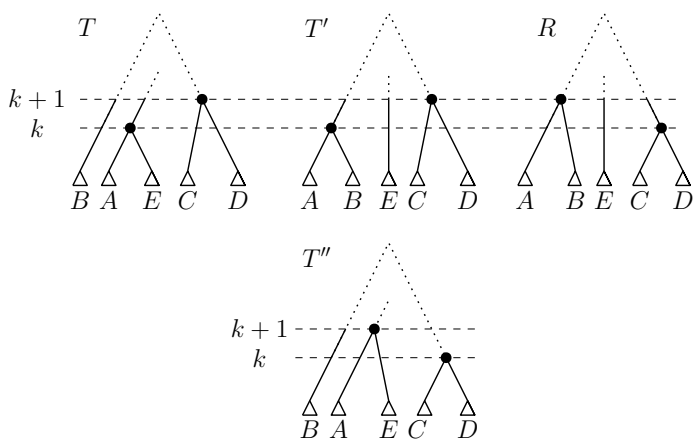

Figure 9: Trees T, T', and R on p, and alternative path p' from T to R via T'' at the bottom if T and T' are connected by an HSPR move at rank k and the nodes of rank k and $k+1$ in T are not connected by an edge, The dotted parts of the trees might contain further nodes and leaves.

3.2. If there is no edge between the nodes of rank k and $k+1$ in T, then there is a subtree $T|_E$ that is sibling of $T|_C$ or $T|_D$ with $E \neq A, B, C, D$. We can assume without loss of generality that $T|_C$ and $T|_E$ are siblings, otherwise we swap notations for C and D. In this case, a path p' can be constructed by first performing a rank move swapping the nodes of rank k (with children $T|_A$ and

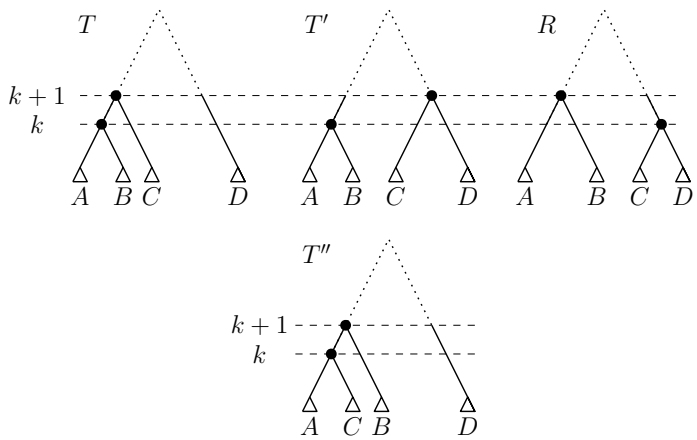

Figure 10: Trees T, T', and R on p, and alternative path p' from T to R via T'' at the bottom if T and T' are connected by an HSPR move at rank $k+1$ and the nodes of rank k and $k+1$ in T are connected by an edge, The dotted parts of the trees might contain further nodes and leaves.

$T|_B$) and $k+1$ (with children $T|_C$ and $T|_E$) of T. We can preform an HSPR move on the resulting tree T'', moving that the subtree $T''|_C$ to become sibling of $T''|_D$, which gives us the tree R as depicted in Figure 11.

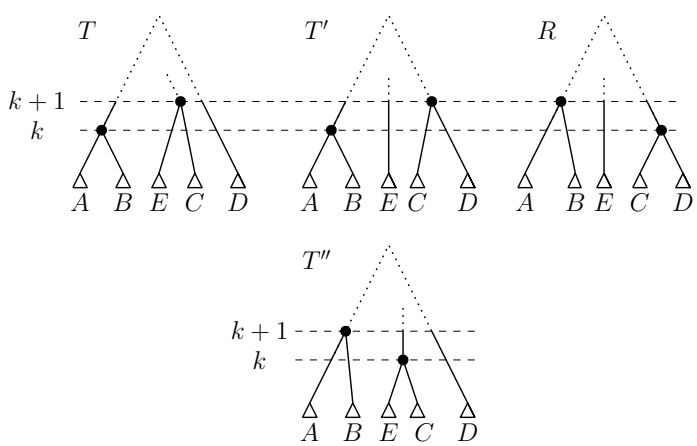

Figure 11: Trees T, T', and R on p, and alternative path p' from T to R via T'' at the bottom if T and T' are connected by an HSPR move at rank k and the nodes of rank k and $k+1$ in T are not connected by an edge, The dotted parts of the trees might contain further nodes and leaves.

In all cases above, we found an alternative path to the path p that consists of either

two HSPR 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 7. *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 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 7 implies that any (not necessarily shortest) path between two ranked 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. 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 7.

Corollary 4. *Two rooted unranked trees T and R can always be ranked so that for the resulting trees T' and R', $d_{\mathrm{RSPR}}(T', R') = d_{\mathrm{HSPR}}(T', R')$.*

Corollary 5. *If there is a shortest path between ranked 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 ranked 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 7 to this path from T_c to T gives a shortest path where all rank moves are at the beginning of the path. But 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.

With 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 put the shortest path from T to T_c with only HSPR moves and the one from T_c and R with only HSPR moves together 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 trees can change the distance between them. 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 [16, 12, 14, 6]. 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. Therefore, 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 8). We will see that this is the case when two trees are similar and the new leaf is added at very different positions 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* $\frac{n-1}{2}$.

The trees T and R of Lemma 6 are displayed in the top row of Figure 12.

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 $d_{\text{HSPR}}(T, R) \geq \frac{n-1}{2}$. And as the HSPR and RSPR distance between caterpillar trees coincides (Corollary 5), $d_{\text{RSPR}}(T, R) \geq \frac{n-1}{2}$. □

Note that Lemma 6 implies that the diameter of RSPR is greater than or equal to $\frac{n-1}{2}$, which is the same boundary we already found for HSPR in Theorem 3. We are now ready to prove the main theorem of this section.

Theorem 8. *Adding a leaf to two ranked trees can decrease their* HSPR *distance by* $\frac{n-1}{2} - 1$.

Proof. We can add a leaf l_{n+1} to the trees T and R of Lemma 6 (depicted in top of Figure 12) so that the resulting trees are connected by one HSPR move: We add a new root to T so that the resulting tree T' has l_{n+1} and T as children. Furthermore, we attach l_{n+1} as sibling of l_1 in R so that their parent has rank one and increase the ranks of all other internal nodes of R by one, giving us a tree R' on $n+1$ leaves. T' and R' are depicted in Figure 12 and it is not hard to see that they are connected by one HSPR move that moves the subtree only consisting of the leaf l_1.

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

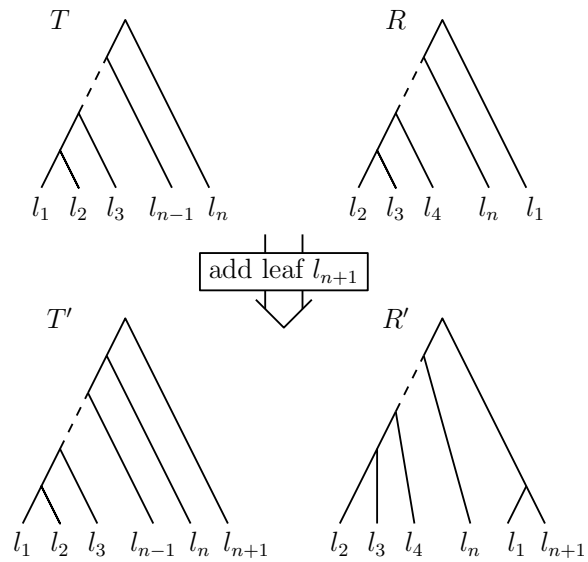

Figure 12: Top: trees T and R with HSPR distance greater than or equal to $\frac{n-1}{2}$ (Lemma 6). Bottom: Adding leaf l_{n+1} to both T and R results in trees T' and R' that are connected by one HSPR move that moves l_1.

By Theorem 9 we see that adding a leaf can only increase the HSPR distance if the two given trees are connected by a rooted (unranked) SPR move. In the following by "unranked version" of a ranked tree T we mean the unranked tree T_u with the same tree topology as T.

Theorem 9. *Let T and R be ranked trees with* HSPR *distance $d_{\text{HSPR}}(T, R) = d > 1$. Furthermore, let T' and R' be ranked trees resulting from adding a leaf x to T and R, respectively, so that $d_{\text{HSPR}}(T', R') = 1$.*

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

Proof. Since T' and R' have HSPR distance one, there is a subtree T_s present in both these trees that is moved by the HSPR move between them. Furthermore, the unranked versions T'_u and R'_u of T' and R', respectively, have distance at most one in (unranked) rSPR as well because either T'_u and R'_u are identical or we can move T_s. This distance cannot be zero in rSPR because, otherwise, changing the ranking of T' and R' would result in trees with distance zero, which cannot be the case if T' and R' are connected by an HSPR move. Hence, $d_{\text{rSPR}}(T'_u, R'_u) = 1$.

Now consider the effect of removing leaf x from T'_u and R'_u to obtain T_u and R_u. We have that T'_u and R'_u differ only in the position of subtree T_s. Then there are three possibilities for T_u and R_u, (i) they differ only in the position of T_s, (ii) they differ only in the position of $T_s - x$, that is, x was part of but not all of T_s, or (iii) $T_u = R_u$, that is, $T_s = x$. Case (i) and (ii) imply that T_u and R_u differ by one rSPR move of T_s or $T_s - x$, respectively. In Case (iii) T_s only consists of x, which means that deleting this leaf from T' and R' results in trees with HSPR distance zero, i.e. $d(T, R) = 0$, contradicting the assumptions of the theorem.

Therefore, we conclude that $d_{\text{rSPR}}(T_u, R_u) = 1$. □

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. With this paper, we fill this gap in research by introducing two ranked SPR treespaces, RSPR and HSPR, that have some properties in common with rooted (unranked) SPR (rSPR). 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 rSPR, which provides a good range of different trees for tree proposals [33]. Adding ranks to rSPR 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 particularly 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 cluster property in ranked SPR. This suggests that the work on Maximum Agreement Forests (MAFs) cannot be transferred to ranked SPR spaces [34]. Since MAFs are essential in the proofs of $\mathcal{NP}$-hardness, 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. That the order of moves on shortest paths in RSPR can be changed so that first rank moves and then HSPR moves are performed suggest 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 distances in these treespaces. Another characteristic of ranked SPR spaces that make 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 establishing 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 [33, 35].

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_paper. 2023.

[9] 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.

[10] 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.

[11] 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.

[12] 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.

[13] 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.

[14] 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.

[15] 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.

[16] 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.

[17] 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.

[18] 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.

[19] 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.

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

[21] 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.

[22] 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.

[23] 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.

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

[25] 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.

[26] 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.

[27] 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.

[28] 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.

[29] 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.

[30] 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.

[31] 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.

[32] 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.

[33] 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.

[34] 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.

[35] 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.

[36] 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.