algorithms

MDPI

Article

A Linear-Time Algorithm for the Isometric Reconciliation of Unrooted Trees

Broňa Brejová * and Rastislav Královič *

Department of Computer Science, Faculty of Mathematics, Physics, and Informatics, Comenius University, Mlynská dolina, 842 48 Bratislava, Slovakia

* Correspondence: brejova@dcs.fmph.uniba.sk (B.B.); kralovic@dcs.fmph.uniba.sk (R.K.)

Received: 15 July 2020; Accepted: 4 September 2020; Published: 8 September 2020

Abstract: In the reconciliation problem, we are given two phylogenetic trees. A species tree represents the evolutionary history of a group of species, and a gene tree represents the history of a family of related genes within these species. A reconciliation maps nodes of the gene tree to the corresponding points of the species tree, and thus helps to interpret the gene family history. In this paper, we study the case when both trees are unrooted and their edge lengths are known exactly. The goal is to root them and to find a reconciliation that agrees with the edge lengths. We show a linear-time algorithm for finding the set of all possible root locations, which is a significant improvement compared to the previous $O(N^3 \log N)$ algorithm.

Keywords: reconciliation; phylogeny; gene tree

1. Introduction

Reconciliation of gene trees and species trees is a computational problem arising in evolutionary biology that has been studied since 1979 [1]. Here, we concentrate on a variant proposed by Ma et al. [2,3], called isometric reconciliation. We provide a linear-time algorithm for reconciling two unrooted trees, which is a significant improvement over the previous $O(N^3 \log N)$ algorithm [4].

A species tree represents an evolutionary history of a group of species. Present-day species are typically placed in the leaves, and internal nodes correspond to speciation events. Speciation is a process in which individuals of a single species separate into two groups that further evolve independently.

A gene tree represents an evolutionary history of a group of related genes from several species. In the simplest scenario, each individual has a single copy of a particular gene, and these genes are propagated along the species tree, yielding the gene tree identical to the species tree. However, during evolution, a gene can be lost (deleted), and it can also become duplicated, whereupon a particular species and its descendants have multiple copies of the gene. After a series of such duplication and deletion events, the gene tree and the species tree will differ. In a gene tree, the leaves represent copies of the gene in the present-day species and the internal nodes represent speciations and duplications. Lost gene copies are not represented in gene trees.

The goal of reconciliation is to map nodes of a gene tree to the corresponding points in a species tree that represent the same points in the evolutionary history. This mapping helps to distinguish whether a node of a gene tree corresponds to a speciation or a duplication and to estimate the number of deletions (see an example in Figure 1).

Note that gene trees and species trees representing evolutionary histories are rooted, with the root representing the last common ancestor of all the leaves. However, trees are in practice constructed from DNA or protein sequences, and many commonly used methods are unable to determine the root location, so instead provide only an unrooted version of the tree [5].

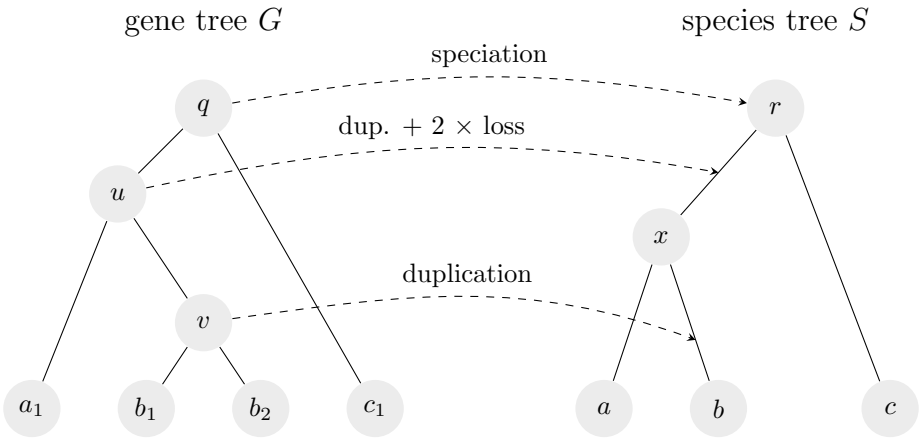

Figure 1. An example of a reconciliation of a gene tree and a species tree. Leaves of the gene tree are labeled by the species of origin; for example, b_1 and b_2 are genes from species b. In the history implied by the reconciliation, there was originally a single copy of the gene, which underwent speciation corresponding to nodes q and r and was passed without change to species c. On the other lineage, the gene was first duplicated (node u), and then underwent speciation in node x, followed by one loss in each of the new lineages. As a result, species a had only one copy of the gene. Finally, the gene was again duplicated on the edge leading to b (node v). Note that if u was mapped to node x by the reconciliation, there would be only a single duplication (node v) and no losses in the corresponding history.

In the isometric reconciliation, both trees have known edge lengths and the reconciliation has to obey these lengths (see an example in Figure 2). The problem was introduced by Ma et al. [2,3], who provided a polynomial-time algorithm for reconciling an unrooted gene tree with a rooted species tree. Brejová et al. [4] have provided a corrected and improved version of this algorithm, which runs in $O(N \log N)$ time, where N is the total number of nodes in the two trees combined. The authors also provided algorithms for several variants of the problem, including an $O(N^5 \log N)$-time algorithm to compute all reconciliations of two unrooted trees. In this paper, we improve this running time to $O(N)$ for gene trees with node degrees bounded by $O(1)$ (typically, unrooted phylogenetic trees have internal nodes of degree three). To achieve this time, we only output pairs of valid root locations in the two trees instead of full reconciliations. Given any pair of such root locations, the full reconciliation can be computed in $O(N \log N)$ time by a simple algorithm given by Brejová et al. [4]. If the earlier algorithm [4] is modified to output only root locations, it still runs in $O(N^3 \log N)$ time; thus the algorithm presented here is a significant improvement.

We also characterize the set of possible root locations, proving that it always forms a connected subgraph of the species tree and that the distances between different root locations in the species tree and their counterparts in the gene tree are preserved.

Finally, note that the problem of gene tree and species tree reconciliation was also studied in other settings [6]. The most traditional is the parsimony setting that uses unweighted trees and seeks a reconciliation minimizing the number of inferred duplications and deletions. This problem can be solved in linear time for two rooted trees [7,8]. For unrooted gene trees, it is possible to find all optimal root locations in linear time [9]. Rooting an unrooted species tree using a set of unrooted gene trees was also considered [10]. Edge lengths were used in the context of probabilistic models of gene duplication and deletion [11–13] and in models allowing horizontal transfer of genes between species [14,15].

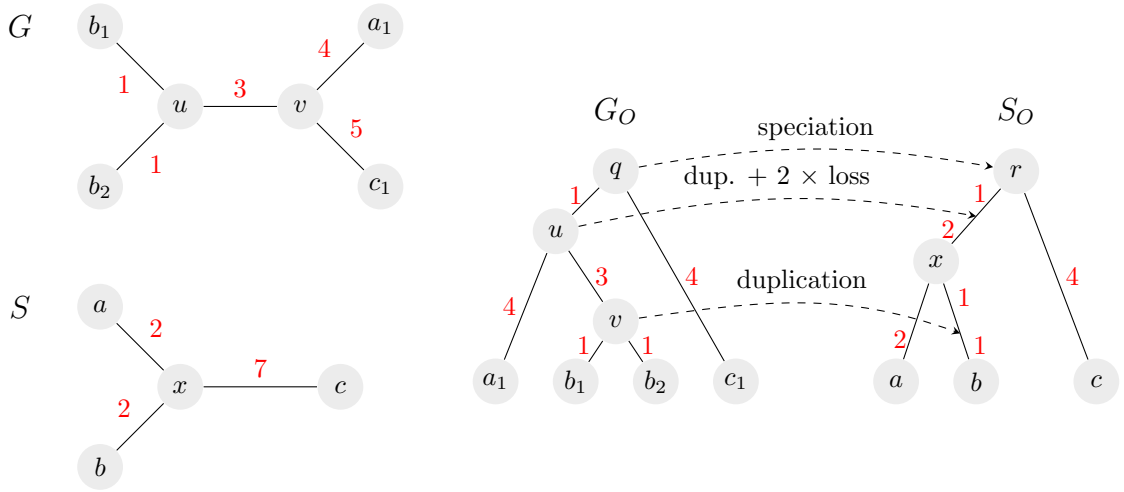

Figure 2. An example of an isometric reconciliation of two unrooted trees. On the left, the input gene tree G and species tree S are shown, both unrooted. On the right is one possible solution, consisting of rooted trees G_O and S_O and a mapping between them. The species tree is rooted on edge $\{x, c\}$ at distance 4 from c. Once the root of S is fixed, the rest of the solution is given uniquely. All possible roots in S are located on the edge $\{x, c\}$ in distances from 0 to 5 from c. Note that in our algorithm, we preprocess tree S to have two leaves b_1 and b_2 instead of leaf b, so that the leaves of G and S are in one-to-one correspondence (see Sections 2.1 and 2.8).

In the rest of this section, we introduce necessary notation and formally define the isometric reconciliation problem. In Section 2, we describe our algorithm and supporting proofs, including the characterization of the solution set. We provide further discussion of our results and several open problems in Section 3.

1.1. Notation

A gene tree will be in the rest of the text denoted G, species tree S. Nodes of G are usually denoted as u and v, nodes of S as x and y, the root of G as q, the root of S as r and leaves in both trees as a and b.

A point in a tree is one of its nodes or a point inside an edge where this edge can be subdivided by a new node. By $d_T(u, v)$, we denote the distance between points u and v in tree T. For a tree T with nodes u and v and real value α, such that $0 \leq \alpha \leq d_T(u, v)$, let $T(u, v, \alpha)$ denote the point located at distance α from node u on the path connecting points u and v. By $\mathsf{lca}_T(u, v)$, we denote the lowest common ancestor of u and v in a rooted tree T.

Let $\{u, v\}$ be an edge in an unrooted tree T. By $\mathsf{sub}(T, u, v)$ we will denote a rooted subtree of T obtained by removing edge $\{u, v\}$ from T, taking the resulting connected component containing u and rooting it in u. A tree with m edges thus has $2m$ rooted subtrees. In rooted trees, we will consider subtrees in the conventional sense, consisting of a node and all its descendants.

For a tree T, let $L(T)$ be the set of its leaves and $V(T)$ the set of its nodes. For a tree T and a set of leaves A, let T_A denote the subtree of T induced by A and all paths connecting pairs of leaves from A. In particular, we will often consider the subtree $G_{L(T)}$ for some rooted subtree T of S; this is the part of G that naturally corresponds to subtree T of S.

1.2. Definition of Reconciliation

Consider a rooted or unrooted tree G (the input gene tree), rooted or unrooted tree S (the input species tree) and input mapping $\mu : L(G) \to L(S)$, which gives for each gene its species of origin. Both trees have edges weighted by non-negative weights. An isometric reconciliation of the triple (G, S, μ) is a triple (G_O, S_O, Φ), where G_O and S_O are weighted rooted trees and Φ is a mapping from $V(G_O)$ to $V(S_O)$ such that the following holds:

- The output gene tree G_O is either G, if G is rooted, or a rooted version of G obtained by subdividing some edge and rooting the tree in this newly created node.
- The output species tree S_O is obtained from S by rooting it if S is unrooted and optionally subdividing some edges by additional nodes and also optionally adding a new path leading from above to the original root of S.
- Mapping Φ agrees with input mapping μ on all leaves.
- Mapping Φ preserves the ancestor/descendant relationships and the distances between ancestors and descendants. For each u and v such that u is a parent of v in G_O, we have that $\Phi(u)$ is an ancestor of $\Phi(v)$ and $d_{S_O}(\Phi(u), \Phi(v)) = d_{G_O}(u, v)$.
- Nodes are not added to S_O and G_O unnecessarily. For each node x which was added to S_O except for the root, there is some node u from G_O such that $\Phi(u) = x$. The edges incident to x have strictly positive weights.

Note that when subdividing a weighted edge $\{u, v\}$, we divide the original edge length between the new edges so that the distance of u and v remains the same. The new nodes added to the output species tree S_O represent duplication events; the path above the root of S corresponds to ancient duplications happening before the first speciation represented in the species tree. Figure 3 shows an example of an isometric reconciliation where the root of G_O maps above the original root of S_O. A more detailed discussion of the isometric reconciliation model can be found in the article by Brejová et al. [4].

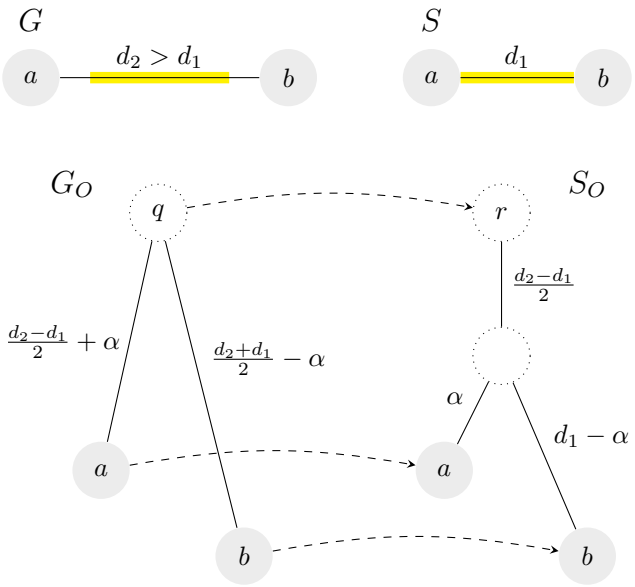

Figure 3. An example of two unrooted trees and their isometric reconciliation, in which the root of G_O is mapped to a point above the original root of S. Both input trees consist of only a single edge; the length of the edge is greater in G. Tree S can be rooted in any point, while the corresponding roots of G form an interval of length d_1 in the middle of the edge. Valid roots are highlighted in yellow.

For algorithmic simplicity, we allow rooting trees only in points located on edges, not in the original nodes. To root a tree in an existing node u, we choose one of the edges incident to u and subdivide it by a new node in the distance 0 from u. This is the only exception where we are allowed to create new edges of zero length.

In this paper, we do not compute the full reconciliation which includes mapping Φ and new nodes added to the input trees. Instead, we output all pairs of valid roots in the two trees, that is, pairs of points in G and S such that when the trees are rooted in these points, they can be reconciled. If desired, the full reconciliation can be easily computed for any particular pair of valid roots in $O(N \log N)$ time.

As we will see in Section 2.7, valid roots in S form a set of intervals of points on individual edges; for edge $\{x, y\}$ in S we can express such an interval as the set of points $S(x, y, \alpha)$ for $\alpha \in [\alpha_1, \alpha_2]$. For some edges, the set of valid roots can be empty, and for some it may contain only a single point. For each non-empty interval, we also have the corresponding interval of valid roots on some edge $\{u, v\}$ in G; this interval thus has the form $G(u, v, \beta)$ for $\beta \in [\beta_1, \beta_2]$.

2. Algorithm

In this section, we describe our linear-time algorithm. A proof-of-concept implementation of this algorithm is available at https://github.com/fmfi-compbio/unrooted-reconciliation.

The previous work shows that for rooted species tree S and unrooted gene tree G, there is at most one root location in G so that the trees can be reconciled [2,4]. The key idea of our algorithm is to take individual rooted subtrees of an unrooted tree S and to compute for each the unique location of the root in the corresponding part of G (Section 2.4).

In the second phase of the algorithm (Section 2.5), we consider each edge of S as a possible root location and combine information from the two rooted subtrees at its endpoints to obtain an interval of positions along this edge where the root can be placed.

To keep the running time linear, our computation in the first two phases of the algorithm does not check all requirements of an isometric reconciliation; the final check in the third phase either validates or rejects all preliminary solutions (Section 2.6).

Finally, in the fourth phase, we compute for each root location in S its exact corresponding root location in G (Section 2.7).

2.1. Assumptions and Preprocessing

In the core of our algorithm, we make the following assumptions.

1. The input trees G and S have the same set of leaves, and the input mapping μ is implicitly represented by leaf equality.
2. All internal nodes of tree S have degree three; S may have zero-length edges.
3. The degrees of all internal nodes of tree G are at least three and are bounded from above by some constant c; all edges of G have strictly positive lengths.

Of these assumptions, the first one may seem unrealistic, because reconciliation is typically performed on gene families where individual species have multiple copies, and thus the input mapping μ is a many-to-one mapping. However, we will show in Section 2.8 that this condition can be satisfied for any input trees by simple preprocessing.

The remaining conditions are satisfied for typical fully resolved phylogenetic trees with strictly positive edge lengths, but we also show how to handle a more general class of trees in preprocessing. The only serious obstacles are high-degree nodes either in the original G or resulting from contracting zero-length edges during preprocessing. Our algorithm works for high-degree nodes in G, but the running time becomes $O(N^3)$.

In the rest of the paper, we assume that any trees G and S satisfy the requirements outlined in this section. The algorithm works for unrooted input trees, but some claims will consider rooted species trees which occur in the algorithm as subproblems.

2.2. Tree Primitives

In our algorithm, we use the following algorithmic building blocks, whose efficient implementation is explained in Section 2.8. We work with points in the trees represented as $T(u, v, \alpha)$ for some nodes u and v in tree T and distance α such that $0 \leq \alpha \leq d_T(u, v)$. The following statements assume that $O(N)$-time preprocessing is done on each input tree $T \in \{G, S\}$.

- Given two points u and v in tree T, we can compute their distance $d_T(u, v)$ in $O(1)$ time.

- Given three points u, v and w in a tree with strictly positive edge lengths, we can determine if v is located on the path connecting u and w in $O(1)$ time; we denote this predicate $\mathsf{between}_G(u, v, w)$.
- Given two points $p_1 = T(u_1, v_1, \alpha_1)$, and $p_2 = T(u_2, v_2, \alpha_2)$ in tree T where u_1, u_2, v_1, v_2 are nodes of T, we can represent point $p = T(p_1, p_2, \beta)$ in the form $p = T(w_1, w_2, \gamma)$ where $w_1, w_2 \in \{u_1, v_1, u_2, v_2\}$, and this representation can be computed in $O(1)$ time.
- Given tree T with m edges, we can list all its $2m$ rooted subtrees so that if node w is a child of u in a subtree $\mathsf{sub}(T, u, v)$; then subtree $\mathsf{sub}(T, w, u)$ occurs in the list before subtree $\mathsf{sub}(T, u, v)$. We call this ordering bottom-up order of the rooted subtrees, and it can be computed in $O(N)$ time. If we process rooted subtrees in the bottom-up order, all subtrees of each rooted subtree T' are processed before T' itself.

2.3. Potential Rooting of a Gene Tree

In the first phase of our algorithm, we process all rooted subtrees of the species tree S and compute potential rootings of the corresponding parts of G, as defined in the next definition.

Definition 1. *Let S be a rooted species tree with root r and G an unrooted gene tree with the same set of leaves. A potential rooting of G with respect to S is a pair (q, h) where q is a point in G and h is a real value such that for any leaf $a \in L(S) = L(G)$ we have*

$$d_G(q, a) = h + d_S(r, a).$$

The intuition behind this definition is that in the reconciliation of S with G, point q is the new root of G, and it maps to distance h above node r. In an isometric reconciliation, we always have $h \geq 0$, but here we allow negative values of h to keep the algorithm simple; negative values of h are ruled out in a later phase of the algorithm. In the rest of this section, we prove a series of claims characterizing potential rootings. Although Definition 1 and the following claims are formulated for rooted tree S, we will eventually apply them to rooted subtrees T of an unrooted tree S, and to keep the set of leaves the same, we will restrict G to $G_{L(T)}$.

Claim 1. *Let S be a rooted species tree with root r and G an unrooted gene tree. Then there is at most one potential rooting of G with respect to S. More precisely, if (q_1, h_1) and (q_2, h_2) are potential rootings, then $h_1 = h_2$ and $d_G(q_1, q_2) = 0$.*

Proof. Let (q_1, h_1), (q_2, h_2) be two potential rootings. If $d_G(q_1, q_2) = 0$, then both h_1 and h_2 must be equal to $d_G(q_1, a) - d_S(r, a)$, where r is the root of S, and a is an arbitrary leaf from $L(S)$.

Therefore, let us assume that $d_G(q_1, q_2) > 0$. Consider the forest obtained by removing the path connecting q_1 and q_2 from G. For $i \in \{1, 2\}$, let us denote by a_i an arbitrary leaf of G contained in the component of this forest containing q_i. Due to the choice of leaf a_1, the path from a_1 to q_2 in G goes through q_1, and thus we obtain the following equations:

$$h_2 + d_S(r, a_1) = d_G(q_2, a_1) = d_G(q_1, a_1) + d_G(q_1, q_2) = h_1 + d_S(r, a_1) + d_G(q_1, q_2),$$

$$h_2 - h_1 = d_G(q_1, q_2).$$

Symmetrically, for a_2 we have

$$h_1 + d_S(x, a_2) = d_G(q_1, a_2) = d_G(q_2, a_2) + d_G(q_1, q_2) = h_2 + d_S(x, a_2) + d_G(q_1, q_2),$$

$$h_2 - h_1 = -d_G(q_1, q_2),$$

which is a contradiction, implying that indeed $d_G(q_1, q_2) = 0$. □

Claim 2. *Let S be a rooted species tree with root r and G an unrooted gene tree. Let S' be a subtree of S with root r'. Let (q,h) be a potential rooting of G with respect to S. Then $G_{L(S')}$ has a potential rooting (q',h') with respect to S', where q' is the point of $G_{L(S')}$ closest to q (distance measured by d_G). In particular, if q is inside $G_{L(S')}$, then $q'=q$. Value h' is defined by the following equation:*

$$h' = h - d_G(q,q') + d_S(r,r'). \tag{1}$$

Proof. Let q' be the point of $G_{L(S')}$ closest to q, and let h' be the value defined by Equation (1). Then for any leaf a of S' the path from a to q must lead through q', and thus we have $d_G(q,a) = d_G(q',a) + d_G(q,q')$. Similarly, $d_S(r,a) = d_S(r',a) + d_S(r,r')$. By combining these two equations, the equation $d_G(q,a) = h + d_S(r,a)$ implied by the potential rooting (q,h) and Equation (1), we obtain

$$\begin{aligned} d_G(q',a) &= d_G(q,a) - d_G(q,q') \\ &= h + d_S(r,a) - d_G(q,q') \\ &= h' + d_G(q,q') - d_S(r,r') + d_S(r,a) - d_G(q,q') \\ &= h' - d_S(r,r') + d_S(r,a) \\ &= h' - d_S(r,r') + d_S(r',a) + d_S(r,r') \\ &= h' + d_S(r',a) \end{aligned}$$

Thus we have proved that (q',h') is the potential rooting, as required. □

The following two claims help us to compute potential rootings.

Claim 3. *Let both S and G contain only a single node x. The potential rooting of G with respect to S is $(x,0)$.*

Claim 4. *Let S be a rooted species tree with root r, which is an internal node with two children r_1 and r_2, and let G be an unrooted gene tree. Let S_1 and S_2 be subtrees of S rooted at r_1 and r_2. Let us assume that $G_{L(S_1)}$ and $G_{L(S_2)}$ have potential rootings (q_1,h_1), (q_2,h_2) (both rootings exist). Then the pair (q,h) is a potential rooting of G if and only if the following three conditions are satisfied.*

1. $h = (d_G(q_1,q_2) - d_S(r_1,r_2) + h_1 + h_2)/2$.
2. $q = G(q_1,q_2,\beta)$, *where* $\beta = h + d_S(r_1,r) - h_1$ *and* $0 \le \beta \le d_G(q_1,q_2)$.
3. *Let π be the path in G between q_1 and q_2. For each $i \in \{1,2\}$ either $q_i = q$ or π is completely outside $G_{L(S_i)}$ except for its endpoint q_i.*

Proof. Let $G_1 = G_{L(S_1)}$ and $G_2 = G_{L(S_2)}$. Tree G consists of G_1 and G_2 and possibly the path π connecting these two trees in G.

First, let us prove that if (q,h) is a potential rooting, it satisfies the three conditions. Let us start with Condition 3 for some $i \in \{1,2\}$. If $q = q_i$, the condition is automatically satisfied. Otherwise, we can use Claim 2 to observe that q_i must be the point from $G_{L(S_i)}$ closest to q, implying that the path from q to q_i is outside $G_{L(S_i)}$ except for q_i. This discussion also implies that q is on the path from q_1 to q_2, as needed in Condition 2.

Next, we use Equation (1) from Claim 2, which gives us the value of h expressed as

$$h = h_i + d_G(q,q_i) - d_S(r,r_i) \tag{2}$$

for each of the subtrees S_i. We obtain the desired value of β from Condition 2 directly from Equation (2) for S_1, noting that $\beta = d_G(q_1,q)$. Finally, using Equation (2) for S_2 and noting that $d_G(q,q_2) = d_G(q_1,q_2) - \beta$, we obtain Condition 1.

To prove that the three conditions imply potential rooting, we first prove that Conditions 1 and 2 imply $d_G(q,q_i) = h + d_S(r,r_i) - h_i$ for each $i \in \{1,2\}$. This equation clearly holds for $i = 1$ from

Condition 2 as $\beta = d_G(q_1, q)$. Therefore, let us consider $d_G(q, q_2)$. From definition of q in Condition 2, we see that $d_G(q, q_2) = d_G(q_1, q_2) - \beta$. Then we use definitions of β and h from Conditions 1 and 2, the fact that $d_S(r_1, r_2) = d_S(r, r_1) + d_S(r, r_2)$ and simple manipulations to get

$$\begin{aligned} d_G(q, q_2) &= d_G(q_1, q_2) - h - d_S(r_1, r) + h_1 && (3)\\ &= d_G(q_1, q_2) - \frac{d_G(q_1, q_2) - d_S(r_1, r_2) + h_1 + h_2}{2} - d_S(r_1, r) + h_1 && (4)\\ &= \frac{d_G(q_1, q_2) + d_S(r_1, r_2) + h_1 - h_2}{2} - d_S(r_1, r) - d_S(r, r_2) + d_S(r, r_2) && (5)\\ &= \frac{d_G(q_1, q_2) - d_S(r_1, r_2) + h_1 + h_2}{2} + d_S(r, r_2) - h_2 && (6)\\ &= h + d_S(r, r_2) - h_2. && (7) \end{aligned}$$

Now consider any leaf $a \in L(S)$. We need to prove that $d_G(q, a) = h + d_S(r, a)$. Let i be the index such that a belongs to $L(S_i)$. Thanks to Condition 3, we have that $d_G(q, a) = d_G(q, q_i) + d_G(q_i, a)$. Since (q_i, h_i) is a potential rooting of G_i, we have that $d_G(q_i, a) = h_i + d_S(r_i, a)$. Finally, since $a \in L(s_i)$, we have $d_S(r, a) = d_S(r, r_i) + d_S(r_i, a)$. These three observations yield the desired equality

$$d_G(q, a) = h_i + d_S(r_i, a) + = h_i + d_S(r_i, a) = h + d_S(r, a).$$

□

2.4. Computation of Potential Rootings

In the first phase of the algorithm, we will compute potential rootings of all rooted subtrees of S. Claim 3 gives the potential rooting of a subtree with one leaf, and Claim 4 allows us to compute the potential rooting of a bigger subtree from rootings of its two subtrees.

Let T be a rooted subtree of S. The point q in a potential rooting (q, h) of $G_{L(T)}$ will be represented in the algorithm as $G(a, b, \alpha)$ for some leaves a and b and distance α. We will also keep a set of representative leaves for each subtree with a potential rooting (q, h). The size of this set will be the same as the degree of q in $G_{L(T)}$. For each edge $\{q, u\}$ incident to q in $G_{L(T)}$, the set will contain a leaf a such that $\mathsf{between}_G(q, u, a)$ is true; that is, the path from q to a uses edge $\{q, u\}$. Note that if q is not a node of G, but rather a point inside some edge e, we will assume that G was modified by subdividing e in this point, and thus the degree of q is two, and it has two incident edges.

The algorithm processes each rooted subtree T of S in the bottom-up order. As a special case, if $G_{L(T)}$ consists only of leaf a, the rooting will be represented as $(q, 0)$, where $q = G(a, a, 0)$, and the representative set is $\{a\}$.

Consider now rooted subtree T with two rooted subtrees S_1, S_2 as in Claim 4; we will also use other notation from the claim. The rooting for subtree T is computed from already known rootings of S_1 and S_2 as follows.

- Compute h using Condition 1 of Claim 4. This requires computing $d_G(q_1, q_2)$, where q_1 and q_2 are represented as $q_1 = G(a_1, b_1, \alpha_1)$ and $q_2 = G(a_2, b_2, \alpha_2)$ for some leaves a_1, b_1, a_2, b_2. This distance computation is one of our tree primitives (see Sections 2.2 and 2.8).
- Compute β from Condition 2; check that $0 \le \beta \le d_G(q_1, q_2)$. Then represent point $q = G(q_1, q_2, \beta)$ in the form $G(a, b, \alpha)$ where $a, b \in \{a_1, b_1, a_2, b_2\}$. This computation is also one of our primitives.
- To check Condition 3, we use the sets of representative leaves $A_1 = \{a_1, \ldots, a_g\}$ and $A_2 = \{b_1, \ldots, b_h\}$ for q_1 and q_2, respectively. We also build the set A for q (see Figure 4).

  If $q_1 = q_2$, Condition 3 is satisfied, and $q = q_1 = q_2$. To build the set A, we start with the union of A_1 and A_2. This clearly covers all outgoing edges from q belonging to $G_{L(T)}$, but some of them may be covered twice, if they are shared between $G_{L(S_1)}$ and $G_{L(S_2)}$. Thus, if for some $a_i \in A_1$ and $b_j \in A_2$ we have that $\mathsf{between}_G(a_i, q, b_j)$ is false, either a_i or b_j is excluded from A.

Now let us assume that $q \neq q_i$ for some $i \in \{1,2\}$; let j be the other index from $\{1,2\}$. We need to check that the edge (q_i, u) leaving from q_i towards q and q_j does not belong to $G_{L(S_i)}$. This is achieved by checking that for each leaf $a \in A_i$ the predicate $\mathsf{between}_G(a, q_i, q_j)$ is true (see Figure 5). If neither q_1 nor q_2 equal q, this check is repeated for both values of i.

To build the set A for q, we consider two cases: if q is one of the endpoints of the path between q_1 and q_2, say $q = q_1$, then we start with set A_1 and add to it one leaf from A_2 as a representative of the edge leaving from q_1 towards q_2.

Finally, if q is inside the path connecting q_1 and q_2, its degree in $G_{L(T)}$ will be two and the set A will consist of one arbitrarily chosen leaf from each A_1 and A_2.

A straightforward implementation of this step takes $O(|A_1| \cdot |A_2|)$ if $q_1 = q_2$ and $O(|A_1| + |A_2|)$ if $q_1 \neq q_2$. If the degree of each node of G is $O(1)$, the running time of this step is also $O(1)$.

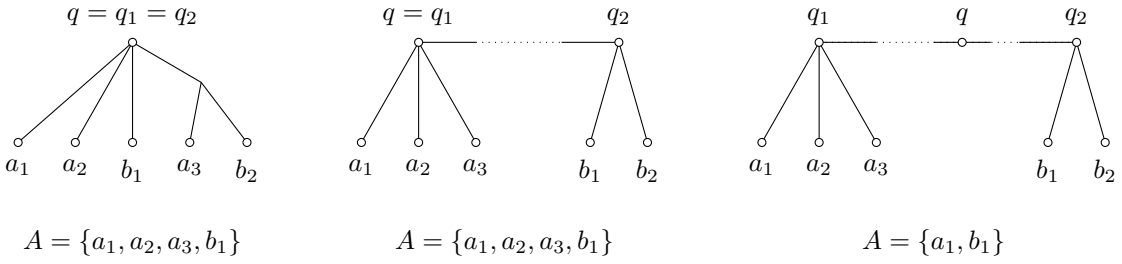

Figure 4. Construction of representative set A for q from representative sets $A_1 = \{a_1, \ldots, a_g\}$ and $A_2 = \{b_1, \ldots, b_h\}$. The algorithm in the text considers three different cases depicted from left to right. Construction of set A involves some arbitrary choices; for example, in the first case, a_3 can be replaced in A by b_2.

Figure 5. Illustration of the algorithm for checking Condition 3 when $q \neq q_i$. We need to check that the path connecting q_i and q_j is outside $G_{L(S_i)}$ except for q_i. For each leaf a from the representative set A_i of q_i, we check that $\mathsf{between}_G(a, q_i, q_j)$ is true. Left: $\mathsf{between}_G(a, q_i, q_j)$ is true; right: $\mathsf{between}_G(a, q_i, q_j)$ is false, $u \in G_{L(S_i)}$.

If the degrees of all nodes of G are $O(1)$, the overall running time for computing potential rootings of all $2m$ subtrees is $O(m)$.

2.5. Potential Roots of an Unrooted Species Tree

Once we have potential rootings computed for all rooted subtrees of an unrooted tree S, we can consider potential roots of S, defined as follows.

Definition 2. *Let G and S be unrooted. A point r in S is called a potential root of S if there exists some potential rooting (q, u) of G with respect to S rooted at r.*

For each edge $\{x_1, x_2\}$ of S, consider rooting tree S at the point $r = S(x_1, x_2, \alpha)$ for $0 \leq \alpha \leq d_S(x_1, x_2)$. This yields a rooted tree S'_α on which we can apply Claim 4, using precomputed potential rootings (q_1, h_1) and (q_2, h_2) for the two subtrees S_1 and S_2 of S rooted at x_1 and x_2. Using this claim, we can compute for which values of α we obtain a potential rooting of G with respect to S'_α.

In particular, Condition 1 of Claim 4 specifies the value of h which does not depend on the unknown distance $\alpha = d_S(x, r)$. Condition 2 specifies the position q of the root in G, but for this position to be valid, value of $\beta = h - h_1 + \alpha$ must be from the specified interval. This translates to interval $[h_1 - h, h_1 - h + d_G(q_1, q_2)]$ for α. Possible values of α also must be within edge $\{x_1, x_2\}$, and thus we take an intersection of two intervals, yielding the following interval of possible values:

$$\alpha \in [\max(0, h_1 - h), \min(d_S(x_1, x_2), h_1 - h + d_G(q_1, q_2))].$$

Note that this interval might be empty, signifying that no potential rooting exists, or it might contain only a single value of α.

Finally, to satisfy Condition 3 of Claim 4 for $i \in \{1, 2\}$, we consider two cases. If the path π connecting q_1 and q_2 lies outside of $G_{L(G_i)}$ except for the endpoints, we can place point q anywhere on this path without violating this condition. However, if some portion of this path is inside $G_{L(G_i)}$ for some $i \in \{1, 2\}$, we have to place q to q_i, restricting possible values of β to a single value (0 or $d_G(q_1, q_2)$, depending on i), and thus α is also restricted to a single value or potentially to an empty set. Condition 3 has to be satisfied for both values of $i \in \{1, 2\}$.

To algorithmically check whether π is outside of $G_{L(G_i)}$, we again use the set A_i of representative leaves for q_i. For each leaf $a \in A_i$, we evaluate the predicate $\mathsf{between}_G(a, q_i, q_j)$, where j is the index from $\{1, 2\}$ different from i. If the predicate is true for all representative leaves, the path is outside.

The result of this phase of the algorithm is for every edge $\{x_1, x_2\}$ of S an interval of potential roots of S which yield potential rootings of G. The location of the root q is given in the form $G(a, b, f(\alpha))$ where α is the distance of the root of S from node x_1 and f is a function of the form $f(\alpha) = \alpha + d$, where $d = h - h_1$ as discussed above.

2.6. Checking Distances

In the third phase of the algorithm, we are given a set of potential roots, and we need to verify whether they correspond to actual reconciliations. We do so based on comparing distances between pairs of selected leaves in trees S and G, as explained in the following two claims.

Note that the definition of isometric reconciliation requires that distances between a node of G and its descendants and ancestors are preserved by the mapping. However, for other pairs of nodes, the distance in G can be greater than the distance of the corresponding points in S. For example, in Figure 2 leaves a_1 and b_1 of G are at distance 8, but the corresponding leaves a and b from S have a distance of only 4. Next we show that the distances in G cannot be smaller than distances in S.

Claim 5. *Let (G_O, S_O, Φ) be an isometric reconciliation of a triple (G, S, μ), where trees S and G can be either rooted or unrooted. Then for each two nodes u and v from G, we have $d_G(u, v) \geq d_S(\Phi(u), \Phi(v))$.*

Proof. Note that the definition of isometric reconciliation implies that if u is an ancestor of v in G; then $\Phi(u)$ is an ancestor of $\Phi(v)$ and $d_{G_O}(u, v) = d_{S_O}(\Phi(u), \Phi(v))$. Additionally, note that for any nodes u and v of G we have $d_G(u, v) = d_{G_O}(u, v)$, and similarly, the distances remain preserved between S and S_O.

Let $w = \mathsf{lca}_{G_O}(u, v)$ and $x = \mathsf{lca}_{S_O}(\Phi(u), \Phi(v))$. Note that $\Phi(w)$ must be an ancestor of x, because it is an ancestor of both $\Phi(u)$ and $\Phi(v)$.

$$\begin{aligned} d_G(u, v) &= d_{G_O}(w, u) + d_{G_O}(w, v) \\ &= d_{S_O}(\Phi(w), \Phi(u)) + d_{S_O}(\Phi(w), \Phi(v)) \\ &\geq d_{S_O}(x, \Phi(u)) + d_{S_O}(x, \Phi(v)) \\ &= d_{S_O}(\Phi(u), \Phi(v)) \\ &= d_S(\Phi(u), \Phi(v)) \end{aligned}$$

□

Given a rooted tree T, let ℓ_T be the mapping which maps each node x of T to the smallest leaf in the subtree of T rooted at x under some fixed complete order on the set $L(T)$.

Claim 6. *Let S be a rooted tree and G an unrooted tree. Trees S and G can be reconciled if and only if the following two conditions hold:*

- *There is a potential rooting (q, h) of G (let G' be G rooted in q)*
- *For each two nodes v_1 and v_2 which are in G' children of the same node we have*

$$d_{G'}(\ell_{G'}(v_1), \ell_{G'}(v_2)) \geq d_S(\ell_{G'}(v_1), \ell_{G'}(v_2)). \tag{8}$$

Note that Equation (8) uses the assumption that the leaves are shared between G, G' and S.

Proof. If the reconciliation (G_O, S_O, Φ) exists, we can obtain the potential rooting (q, h) by taking the root of G_O as q and setting $h = d_{S_O}(\Phi(q), r)$ where r is the root of tree S. Inequality (8) is implied by Claim 5.

Let us now assume the existence of a potential rooting (q, h) satisfying the two conditions. We will root tree G in q and construct the mapping Φ as follows. Node q is mapped to the point located in distance h above the root of S; let us call this point r and denote tree S extended with r as S'. Each node v of the rooted G' is then mapped to $\Phi(v) = S'(r, \ell_{G'}(v), d_{G'}(q, v))$. First of all, we need to prove that this mapping exists, in particular that $d_{G'}(q, v) \leq d_{S'}(r, \ell_{G'}(v))$. This is true because the definition of a potential rooting implies that $d_{S'}(r, \ell_{G'}(v)) = d_{G'}(q, \ell_{G'}(v))$, and since the path from q to $\ell_{G'}(v)$ passes through v, we have the desired inequality. Note that the leaves of G' are correctly mapped by Φ to their counterparts in S'.

Next we prove that if v is an ancestor of w in G' and $\Phi(v)$ is an ancestor of $\Phi(w)$ in S', then $d_G(v, w) = d_S(\Phi(v), \Phi(w))$. Based on the ancestor relationships, the path from w to q passes through v and the path from $\Phi(w)$ to r passes through $\Phi(v)$, and therefore, $d_{G'}(w, q) = d_{G'}(w, v) + d_{G'}(v, q)$ and $d_{S'}(\Phi(w), r) = d_{S'}(\Phi(w), \Phi(v)) + d_{S'}(\Phi(v), r)$. Based on the definition of Φ, $d_{G'}(w, q) = d_{S'}(\Phi(w), r)$ and $d_{G'}(v, q) = d_{S'}(\Phi(v), r)$. These equations imply the desired result.

Finally, we need to prove that if v is the parent of w in G' then $\Phi(v)$ is an ancestor of $\Phi(w)$ in S'. Let $a_w = \ell_{G'}(w)$ and $a_v = \ell_{G'}(v)$. If $a_w = a_v$, both $\Phi(v)$ and $\Phi(w)$ are on the path from r to a_w, and by comparing their distances from the root, we see that $\Phi(v)$ must be an ancestor of $\Phi(w)$.

If $a_w \neq a_v$, let w' be the child of v on the path to a_v. Then $\ell_{G'}(w') = a_v$, and w and w' are two children of the same node of G'. Therefore, we can apply Inequality (8) to obtain $d_{G'}(a_v, a_w) \geq d_{S'}(a_v, a_w)$. Note that $v = \mathsf{lca}_{G'}(a_v, a_w)$ and thus $d_{G'}(a_v, a_w) = d_{G'}(a_v, q) + d_{G'}(a_w, q) - 2d_{G'}(q, v)$. Similarly in S', let $x = \mathsf{lca}_{S'}(a_v, a_w)$; then $d_{S'}(a_v, a_w) = d_{S'}(a_v, r) + d_{S'}(a_w, r) - 2d_{S'}(r, x)$. By combining these three facts, we get the inequality $d_{G'}(v, q) \leq d_{S'}(x, r)$. As $\Phi(q) = r$ and r is an ancestor of $\Phi(v)$, we have that $d_{G'}(v, q) = d_{S'}(\Phi(v), r)$. Therefore, we see that the depth of $\Phi(v)$ in S' is smaller or equal to the depth of x. Both x and $\Phi(v)$ are ancestors of a_v, which implies that $\Phi(v)$ must be an ancestor of x (which includes the possibility that $\Phi(v) = x$). This in turn implies that $\Phi(v)$ must be also an ancestor of a_w. Again, since both $\Phi(v)$ and $\Phi(w)$ are ancestors of a_w and $\Phi(w)$ is in a greater depth than $\Phi(v)$, it must be a descendant of $\Phi(v)$. □

Claim 5 shows that given unrooted input trees G and S, we can check any pair of leaves a and b and if $d_G(a, b) < d_S(a, b)$, then no reconciliation of these trees exists. Claim 6 shows that for a fixed potential root in S, it is sufficient to check distances between $O(N)$ pairs of leaves to verify that the potential rooting corresponds to a valid reconciliation.

However, we would like to verify all potential roots of S in $O(N)$ total time. For every rooted subtree $\mathsf{sub}(G, v, u)$ of G, we can find its smallest leaf; this leaf will correspond to $\ell_{G'}(v)$ for all rooted versions G' of G in which this rooted subtree appears as one of the subtrees. These smallest leaves can

be easily computed in $O(N)$ using the bottom-up ordering of rooted subtrees of G. Then, given any node u of G, we take all pairs of its neighbors v_1 and v_2, take the smallest leaves a_1 and a_2 in the subtrees $\mathsf{sub}(G, v_1, u)$ and $\mathsf{sub}(G, v_2, u)$ and check whether $d_G(a_1, a_2) \geq d_S(a_1, a_2)$. Provided that tree G has at least one internal node, these checks cover all pairs of leaves $\ell_{G'}(v_1)$ and $\ell_{G'}(v_2)$ from Claim 6 that can occur for any root location in S. If any of these checks is negative, we reject all potential roots of S (using Claim 5); if all checks are positive, then all potential roots of S are valid (using Claim 6). Note that the case of tree G without an internal node is trivial, as it then has at most two leaves. If the degrees of all nodes in G are $O(1)$, the computation can be done in $O(N)$ time.

2.7. Final Output and Possible Solution Sets

This section has two goals. First, we want to characterize the set of possible solutions of the isometric reconciliation for two unrooted trees, which is of theoretical interest. We also need the results proved here to efficiently transform the output of our algorithm to its final form. We will characterize sets of potential roots, but these results also apply to the final valid roots, as according to the previous section, either all potential roots are valid or none are.

We start with a generalized version of Claim 2, which can be proved in the same way as the original claim.

Claim 7. *Let S be a rooted species tree with root r and G an unrooted gene tree. Let A be a subset of leaves, and consider trees S_A and G_A induced by leaves from A and all paths connecting them. Let r' be the root of S_A, that is, the lowest common ancestor in S of all leaves in A. Let (q, h) be a potential rooting of G with respect to S. Then G_A has a potential rooting (q', h') with respect to S_A, where q' is the point of G_A closest to q (distance measured by d_G). Value h' is defined by the following equation:*

$$h' = h - d_G(q, q') + d_S(r, r'). \tag{9}$$

In particular, if r belongs to S_A then $r' = r$ and if in addition q belongs to G_A, then also $q' = q$ and $h' = h$.

Before stating our core observation characterizing the set of potential solutions of the problem in Claim 9, we will add one more technical claim.

Claim 8. *Consider points r_1 and r_2 in unrooted S and q_1 and q_2 in unrooted G and values h_1 and h_2 such that (q_1, h_1) is a potential rooting of G with respect to S rooted in r_1 and (q_2, h_2) is a potential rooting with respect to S rooted in r_2. Let a be any leaf, let x be the point on the path between r_1 and r_2 closest to leaf a under d_S, and let u be the point on the path between q_1 and q_2 closest to a under d_G. Then*

$$d_G(u, q_1) - d_S(x, r_1) = (d_G(q_1, q_2) - d_S(r_1, r_2) + h_1 - h_2)/2.$$

In particular, if $h_1 = h_2$ and $d_G(q_1, q_2) = d_S(r_1, r_2)$, then $d_G(u, q_1) = d_S(x, r_1)$.

Note that we will later show that conditions $h_1 = h_2$ and $d_G(q_1, q_2) = d_S(r_1, r_2)$ are always satisfied, and thus $d_G(u, q_1) = d_S(x, r_1)$.

Proof. From the definition of a potential rooting, we get $d_G(a, q_1) - d_S(a, r_1) - h_1 = d_G(a, q_2) - d_S(a, r_2) - h_2 = 0$. By splitting individual distances at points x and u, we get $d_G(a, q_1) = d_G(a, u) + d_G(u, q_1)$, $d_G(a, q_2) = d_G(a, u) + d_G(u, q_2) = d_G(a, u) + (d_G(q_1, q_2) - d_G(u, q_1))$ and similarly for tree S. By combining these equations, we get the desired claim. □

Claim 9. *Let G and S be unrooted trees. Let $\mathcal{R}$ be the set of potential roots of S. Let r_1 and r_2 be two points from $\mathcal{R}$, and let r_3 be some point on the path connecting r_1 and r_2 in S. For $i \in \{1, 2, 3\}$, let $S^{(i)}$ be S rooted in r_i, and let (q_i, h_i) be the potential rooting of G with respect to $S^{(i)}$ (if it exists). Then the following four statements hold:*

1. $d_S(r_1, r_2) = d_G(q_1, q_2)$;
2. $h_1 = h_2$;
3. $r_3 \in \mathcal{R}$ *and thus the potential rooting* (q_3, h_3) *exists;*
4. q_3 *is on the path between* q_1 *and* q_2.

Proof. If both r_1 and r_2 belong to the same edge of S, all four statements are implied by the discussion in Section 2.5, where it is proved that the set of potential rootings for a single edge is either empty or it is an interval of points on this edge. These points all share the same value of h and their counterparts form an interval on some path in G of the same length as the interval in S.

Consider now the case when r_1 and r_2 are not on the same edge. Let $x_1, \ldots, x_z$ be the path in S such that r_1 is on the edge $\{x_1, x_2\}$, and r_2 is on the edge $\{x_{z-1}, x_z\}$ (refer to Figure 6). All leaves can be partitioned into three classes: let A_1 be the set of leaves a such that $\mathsf{between}_S(a, x_1, x_z)$ is true, A_2 be the set of leaves such that $\mathsf{between}_S(a, x_z, x_1)$ is true and B be the remaining leaves. Let $A = A_1 \cup A_2$.

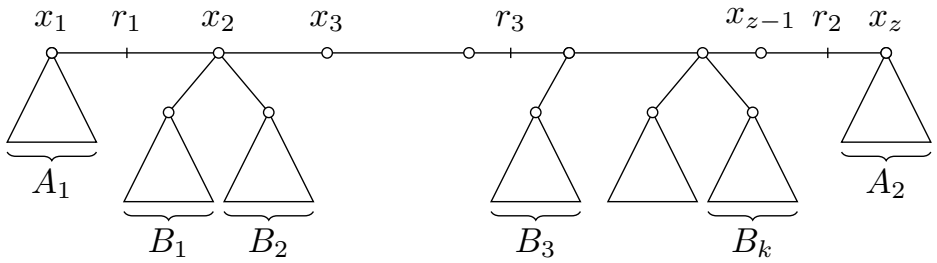

Figure 6. The path containing r_1, r_2 and r_3 in tree S. Leaves are partitioned into sets A_1, A_2 and B, where B is in the figure represented by the union $B_1 \cup B_2 \cup \cdots \cup B_k$.

Consider now trees S_A and G_A. Nodes $x_2, \ldots, x_{z-1}$ all have degree two in S_A, and thus we can replace path $x_1, \ldots, x_z$ by a single edge in S_A. We can apply Claim 7 to $S^{(1)}$ and $S^{(2)}$ to see that points r_1 and r_2 must be potential roots in S_A as well (see Figure 7). Since r_1 and r_2 are on a single edge in S_A, all four statements of this claim hold for r_1, r_2 and r_3 in trees G_A and S_A. Let (q'_i, h'_i) be the potential rooting of G_A with respect to $S_A^{(i)}$ for $i \in \{1, 2, 3\}$. We thus have that (q'_3, h'_3) exists as a potential rooting of G_A with respect to $S^{(3)}$; it is located on the path between q'_1 and q'_2, $d_S(r_1, r_2) = d_G(q'_1, q'_2)$, and $h'_1 = h'_2$. From now on we will denote the common value $h'_1 = h'_2$ as h. We also have that $h'_3 = h$ because r_1, r_2 and r_3 are all on the same edge of S_A.

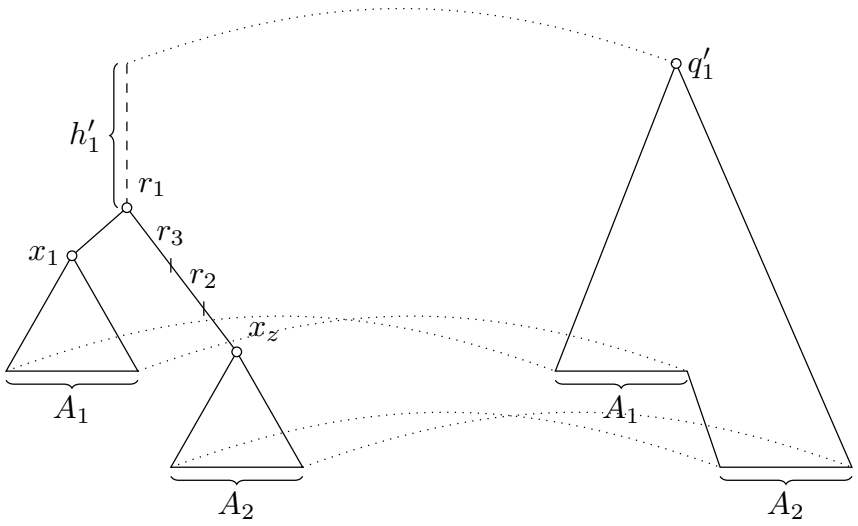

Figure 7. Tree $S_A^{(i)}$ (**left**) and tree G_A with potential rooting (q'_1, h'_1) (**right**).

Next we prove that $q_1 = q'_1$ and $q_2 = q'_2$. According to Claim 7, q'_1 is the closest point in G_A to q_1. Consider now the path from q_1 to q'_2, and note that q'_2 is in G_A. The point where this path first enters G_A must be q_1. Symmetrically, q'_2 must be on the path from q_2 to q'_1. This implies that the points q_1, q'_1, q'_2, q_2 are on a single path in G in this order, although some adjacent points may coincide. Let us now assume that $q_1 \neq q'_1$, and thus q_1 is not in G_A. Then there must be a leaf $b \in B$ such that q_1 is on the path from b to q'_1. For $i \in \{1, 2\}$ we have that $h = h'_i = h_i - d_G(q_i, q'_i)$ by Claim 7. Since $r_i \in \mathcal{R}$, we have $d_G(b, q_i) = d_S(b, r_i) + h_i$. Combining these two facts, we get

$$d_G(b, q_i) = d_S(b, r_i) + h + d_G(q_i, q'_i). \tag{10}$$

In addition,

$$\begin{aligned} d_G(b, q_2) &= d_G(b, q_1) + d_G(q_1, q'_1) + d_G(q'_1, q'_2) + d_G(q'_2, q_2) && (11)\\ &= (d_S(b, r_1) + h + d_G(q_1, q'_1)) + d_G(q_1, q'_1) + d_S(r_1, r_2) + d_G(q'_2, q_2). && (12) \end{aligned}$$

By combining the right-hand side of Equation (10) for $i = 2$ and the right-hand side of Equation (12), we get

$$d_S(b, r_2) - (d_S(b, r_1) + d_S(r_1, r_2)) = 2d_G(q_1, q'_1).$$

The left-hand side of this equation is at most 0 due to triangle inequality, and therefore, the right-hand side must be zero, which is a contradiction with our assumption that q_1 is not in G_A. Therefore, we have that $q_1 = q'_1$ and analogously $q_2 = q'_2$. This implies (via Claim 7) that $h_1 = h_2 = h$, completing the proof of Statements 1 and 2 in the general case for the original trees G and S.

Finally, we need to prove Statements 3 and 4 for original trees G and S. We have proved that (q'_3, h'_3) is a potential rooting of G_A with respect to $S_A^{(3)}$ and that $h'_3 = h$. We will prove that (q'_3, h) is also a potential rooting of G with respect to $S^{(3)}$. Consider a leaf a. If $a \in A$, then we already know from G_A that $d_G(q'_3, a) = d_S(r_3, a) + h$.

We still need prove the same equality for $a \in B$. Consider points u and x as in Claim 8; the claim implies that $d_S(x, r_1) = d_G(u, q_1)$. Point x may be located either between r_1 and r_3 or between r_3 and r_2. We will assume the latter; the former is proved analogously by switching the roles of r_1 and r_2. Point u is then located between q_3 and q_2, as thanks to Claim 8, the distances along paths from r_1 to r_2 and from q_1 to q_2 are preserved. Finally, we obtain

$$\begin{aligned} d_S(a, r_3) &= d_S(a, r_1) - d_S(r_1, r_2)\\ &= d_G(a, q_1) - h - d_G(q_1, q'_3)\\ &= d_G(a, q'_3) - h. \end{aligned}$$

□

The previous claim broadly characterizes the full set of solutions; the next one observes a useful property of solutions located on a single edge of S.

Claim 10. *Let G and S be unrooted trees. Let $\{x, y\}$ be an edge in S and let $[\alpha_1, \alpha_2]$ be an interval such that every point $S(x, y, \alpha)$ for $\alpha \in [\alpha_1, \alpha_2]$ is a potential root of S. Then there is an edge $\{u, v\}$ in G and interval $[\beta_1, \beta_2]$ such that when we root S in the potential root $S(x, y, \alpha)$ for $\alpha \in [\alpha_1, \alpha_2]$, its corresponding potential rooting (q, h) has the form $q = G(u, v, \beta)$ for $\beta \in [\beta_1, \beta_2]$.*

Proof. Our discussion in Section 2.5 implies that potential rootings of G corresponding to a single edge of S form an interval on some path, that is, they can be expressed as $G(a, b, \beta)$. It remains to prove that the interval of possible values of β is within one edge of G. This is implied by Claim 8, where as r_1 and r_2, we use the endpoints $S(x, y, \alpha_1)$ and $S(x, y, \alpha_2)$, and as q_1 and q_2 the endpoints in

the corresponding interval in S. Since the path between r_1 and r_2 is within a single edge of S, point x for any leaf will be either r_1 or r_2. This implies that point u will be either q_1 or q_2. As a result, there is no node on the path between q_1 and q_2, where the paths to other leaves branch out. As tree G has all internal nodes of degree at least 3, q_1 and q_2 must be located on a single edge. □

Claims 9 and 10 are important for the final part of our algorithm, in which we transform the solution to the final form. As we have seen in the proof of Claim 10, the potential roots of G for a single edge of S are specified within an interval on some path. Thanks to Claim 10, we know that they are in fact located on a single edge. Our goal is now to algorithmically find this edge $\{u, v\}$ in G and the interval $[\beta_1, \beta_2]$. To keep the running time $O(N)$, we want to avoid doing such computation for each edge of S individually. Instead, we use the connected nature of the set of solutions in both S and G implied by Claim 9. We traverse tree S by a depth-first search, and the first time we encounter an edge with a non-empty set of potential roots, we find the corresponding edge in G; even a naive $O(N)$-time search is sufficient. Each time we encounter another edge $\{x, y\}$ of S with a non-empty solution set, we have already seen some adjacent edge $\{x, z\}$ with a solution. The potential root of G corresponding to rooting S in the shared vertex x is some point p, which we have already found, and thus we need to search for appropriate edge $\{u, v\}$ only among the edges incident to p. Since the node degree in G is bounded by $O(1)$, this can be done in $O(1)$ time. To check if a candidate edge $\{u', v'\}$ is the one containing points of the form $G(a, b, \alpha)$, we need to verify $\mathsf{between}_G(a, u', b)$, $\mathsf{between}_G(a, v', b)$ and to compare $d_G(a, u)$ and $d_G(a, v)$ to the desired range of α.

2.8. Further Algorithmic Details

In this section, we provide further details on the input preprocessing and tree primitives from Sections 2.1 and 2.2.

First, we need to preprocess our trees so that they have the same set of leaves. Definition of reconciliation allows multiple leaves $a_1, \ldots, a_k$ of G to correspond to a single leaf a of S (this corresponds to multiple copies of a gene in a single species). In that case, we add new leaves $a_1, \ldots, a_k$ to S, all connected to node a by edges of zero length. Each leaf a_i of G will naturally correspond to leaf a_i of S. If, on the other hand, some leaf of S does not occur in G, it can be deleted from S. Leaves of G with no corresponding leaf of S are not considered in the reconciliation problem. Thus these changes will satisfy Assumption 1 from Section 2.1.

Assumption 2 can be achieved by replacing nodes of higher degree in S with binary subtrees connected by zero-length edges. A node of degree two in both S and G can be bypassed by an edge connecting its two neighbors. The condition on non-zero lengths in G (Assumption 3) can be achieved by contracting edges of zero length. However, if some leaf a is connected to its neighbor in G by an edge of length 0, contracting this edge would remove the leaf. Therefore, we can instead lengthen the edge to a by some constant, and then also lengthen the edge leading to a in S by the same amount.

It is easy to see that all these transformations leave the set of solutions practically unchanged; any reconciliation in the preprocessed trees can be easily mapped back to the original trees and vice versa. The only exceptions are reconciliations that root the modified trees on edges added or extended in the preprocessing phase. Such rootings have no counterpart in the original trees and can be simply omitted from the final output.

Now let us consider efficient implementation of the tree primitives from Section 2.2 for each tree $T \in \{G, S\}$. In the precomputation phase, we root the tree T in an arbitrary node. On the resulting rooted tree, we precompute data structures allowing $O(1)$ computation of lca_T; this precomputation can be done in $O(N)$ time [16,17]. We also store the distance to the root in each node; let us denote this value for node u as $d[u]$. Then the distance between nodes u and v is $d[u] + d[v] - 2d[\mathsf{lca}_T(u, v)]$, which can be computed in constant time.

However, we also need to be able to compute distances between points of the form $p_1 = T(u_1, v_1, \alpha_1)$ and $p_2 = T(u_2, v_2, \alpha_2)$. Our technique for this computation is to synchronize these

points, that is, to express them both in the form $p_i = T(w_1, w_2, \beta_i)$, where $w_1, w_2 \in \{u_1, v_1, u_2, v_2\}$. Once the points are synchronized on a single path between w_1 and w_2, their distance is simply $|\beta_1 - \beta_2|$.

To synchronize the points, we first consider the path between u_1 and v_1 and node u_2. The path from u_1 to u_2 diverges from the path from u_1 to v_1 at some node x_1. We can compute the distance $d_T(u_1, x_1)$ as $(d_T(u_1, v_1) + d_T(u_1, u_2) - d_T(v_1, u_2))/2$; this is well known from the neighbor joining algorithm [18]. Our next goal is to express p_1 in the form $T(u_2, y_1, \gamma_1)$ for some node $y_1 \in \{u_1, v_1\}$. If $\alpha_1 \leq d_T(u_1, x_1)$, p_1 is located on the shared portion of the two paths considered above and thus $p_1 = T(u_2, u_1, d_T(u_1, u_2) - \alpha_1)$. Otherwise, p_1 is located after the path from u_1 to v_1 passing through x_1, and thus $p_1 = T(u_2, v_1, \gamma_1)$, where $\gamma_1 = d_T(u_2, v_1) - d_T(u_1, v_1) + \alpha_1$. To obtain this expression for γ_1, observe that $d_T(v_1, p_1) = d_T(u_1, v_1) - \alpha_1$ and $\gamma_1 = d_T(u_2, p_1) = d_T(u_2, v_1) - d_T(v_1, p_1)$.

Now we have both points specified in the form $p_i = T(u_2, y_i, \gamma_i)$, where $y_2 = v_2$ and $\gamma_2 = \alpha_2$. We now consider the node x_2 where paths from u_2 to y_1 and from u_2 to y_2 diverge. Again, we can compute $d_T(u_2, x_2)$ as above. If at least one of p_1 and p_2 is closer to u_2 than x_2, then both points lie on a path from u_2 to one of the y_i and can be synchronized easily. Otherwise, they are synchronized on the path from y_1 to y_2 using similar distance transformations as above. Since the synchronization is only a simple case analysis, it can be easily computed in $O(1)$ time, and thus we are able to compute distances between any two points in $O(1)$ time.

The next primitive is to take two points p_1 and p_2 of the form $p_i = T(u_i, v_i, \alpha_i)$ and to express point $p = T(p_1, p_2, \beta)$ in the form $p = T(w_1, w_2, \gamma)$ where $w_1, w_2 \in \{u_1, v_1, u_2, v_2\}$. This is also easily achieved by synchronization. Once we have both p_1 and p_2 expressed on the same path between some w_1 and w_2, we easily determine the necessary distance γ of the new point p from w_1, since we know the distance of p_1 from w_1 and distance of p from p_1. Again, this works in $O(1)$ time.

Distance computations can be also used to evaluate predicate $\mathsf{between}_T(u, v, w)$. In the absence of zero-length edges, this predicate is satisfied if and only if $d_T(u, w) = d_T(u, v) + d_T(v, w)$. In our algorithm, this predicate is used only for tree G, which is assumed to have strictly positive edge lengths.

Finally, we need to provide a bottom-up ordering of all rooted subtrees of a tree T. This is computed by two tree traversals. Note that the tree is for technical reasons rooted in an arbitrary node.

First, we do a post-order traversal of the tree. After exploring node u and its subtrees, we output subtree $\mathsf{sub}(T, u, v)$, where v is the parent of u in the current rooting. If u is a root, we skip this output. This traversal lists for each edge one of the two subtrees corresponding to the removal of this edge—namely, the one rooted in the node, which is lower in the auxiliary rooting. The second traversal proceeds in the pre-order fashion. After reaching node u from its parent v, it outputs $\mathsf{sub}(T, v, u)$, and afterwards it recursively traverses children of u.

3. Conclusions

In this paper, we have shown how to compute valid root locations for isometric reconciliation of two unrooted trees in linear time, assuming that the gene tree does not have nodes with very high degrees, which is typically true for phylogenetic trees. Nonetheless, linear-time algorithm for trees with arbitrary degrees is an interesting open question. Full reconciliation can be easily computed for any selected root location discovered by our algorithm. However, root locations themselves might be of interest; indeed rooting unrooted trees is one of the motivations for performing reconciliation [9,10].

Note that in practical applications, we might have one species tree and multiple gene trees, each for a different gene family. Our algorithm can reconcile each gene tree separately with the species tree, and then we can choose a root location in S that agrees with the highest number of gene trees. This in turn implies root locations in these gene trees.

The earlier algorithm for isometric reconciliation of unrooted trees [4] finds all distinct reconciliations in $O(N^5 \log N)$ time; the high running time is partially caused by the potentially high number of such reconciliations. Perhaps some observations from the present paper can be used to provide a tighter upper bound and thus a more efficient algorithm for completely listing all solutions. In practice, it might be more desirable to find algorithms for efficiently computing various statistics

over the set of all solutions, such as the minimum and maximum numbers of duplication events occurring on a particular edge of a species tree.

Unlike classical parsimony-based reconciliation approaches, isometric reconciliation uses edge length information. However, edge lengths computed by phylogeny reconstruction methods are not exact, and thus an extension of this algorithm taking into account edge length uncertainty would be desirable.

In this paper, we also characterize the set of all solutions, showing that they form connected subgraphs in both trees. This is similar to the plateau property defined for parsimony-based reconciliation of an unrooted gene tree with a rooted species tree [9,19].

Author Contributions: Investigation, B.B. and R.K., writing—original draft preparation, B.B. writing—review and editing, B.B. and R.K. Both authors have read and agreed to the published version of manuscript.

Funding: This research was supported in part by grants from the Slovak Research and Development Agency (APVV-18-0239) and VEGA (1/0463/20, 1/0601/20, 1/0458/18).

Acknowledgments: The authors would like to thank Tomáš Vinař, Dana Pardubská and anonymous reviewers for useful comments.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Goodman, M.; Czelusniak, J.; Moore, G.W.; Romero-Herrera, A.; Matsuda, G. Fitting the gene lineage into its species lineage, a parsimony strategy illustrated by cladograms constructed from globin sequences. *Syst. Biol.* **1979**, *28*, 132–163. [CrossRef]
2. Ma, J.; Ratan, A.; Raney, B.J.; Suh, B.B.; Miller, W.; Haussler, D. The infinite sites model of genome evolution. *Proc. Natl. Acad. Sci. USA* **2008**, *105*, 14254–14261. [CrossRef] [PubMed]
3. Ma, J.; Ratan, A.; Raney, B.J.; Suh, B.B.; Zhang, L.; Miller, W.; Haussler, D. DUPCAR: Reconstructing contiguous ancestral regions with duplications. *J. Comput. Biol.* **2008**, *15*, 1007–1027. [CrossRef] [PubMed]
4. Brejová, B.; Gafurov, A.; Pardubská, D.; Sabo, M.; Vinař, T. Isometric gene tree reconciliation revisited. *Algorithms Mol. Biol.* **2017**, *12*, 17. [CrossRef] [PubMed]
5. Felsenstein, J. *Inferring Phylogenies*; Sinauer Associates: Sunderland, MA, USA, 2004.
6. Doyon, J.P.; Ranwez, V.; Daubin, V.; Berry, V. Models, algorithms and programs for phylogeny reconciliation. *Brief. Bioinform.* **2011**, *12*, 392–400. [CrossRef] [PubMed]
7. Zhang, L. On a Mirkin-Muchnik-Smith conjecture for comparing molecular phylogenies. *J. Comput. Biol.* **1997**, *4*, 177–187. [CrossRef] [PubMed]
8. Eulenstein, O. *A Linear Time Algorithm for Tree Mapping*; GMD-Forschungszentrum Informationstechnik: Sankt Augustin, Germany, 1997.
9. Górecki, P.; Tiuryn, J. Inferring phylogeny from whole genomes. *Bioinformatics* **2007**, *23*, e116–e122. [CrossRef] [PubMed]
10. Emms, D.M.; Kelly, S. STRIDE: Species tree root inference from gene duplication events. *Mol. Biol. Evol.* **2017**, *34*, 3267–3278. [CrossRef] [PubMed]
11. Sennblad, B.; Lagergren, J. Probabilistic orthology analysis. *Syst. Biol.* **2009**, *58*, 411–424. [CrossRef] [PubMed]
12. Górecki, P.; Burleigh, G.J.; Eulenstein, O. Maximum likelihood models and algorithms for gene tree evolution with duplications and losses. *BMC Bioinform.* **2011**, *12*, 1. [CrossRef] [PubMed]
13. Doyon, J.P.; Hamel, S.; Chauve, C. An efficient method for exploring the space of gene tree/species tree reconciliations in a probabilistic framework. *IEEE/ACM Trans. Comput. Biol. Bioinform.* **2012**, *9*, 26–39. [CrossRef] [PubMed]
14. Doyon, J.P.; Scornavacca, C.; Gorbunov, K.Y.; Szöllősi, G.J.; Ranwez, V.; Berry, V. An efficient algorithm for gene/species trees parsimonious reconciliation with losses, duplications and transfers. In *Comparative Genomics (RECOMB-CG 2012)*; Springer: Berlin/Heidelberg, Germany, 2010; Volume 6398, pp. 93–108.
15. Bansal, M.S.; Alm, E.J.; Kellis, M. Efficient algorithms for the reconciliation problem with gene duplication, horizontal transfer and loss. *Bioinformatics* **2012**, *28*, i283–i291. [CrossRef] [PubMed]

16. Harel, D.; Tarjan, R.E. Fast algorithms for finding nearest common ancestors. *SIAM J. Comput.* **1984**, *13*, 338–355. [CrossRef]
17. Bender, M.A.; Farach-Colton, M. The LCA problem revisited. In *LATIN 2000: Theoretical Informatics*; Springer: Berlin/Heidelberg, Germany, 2000; Volume 1776, pp. 88–94.
18. Saitou, N.; Nei, M. The neighbor-joining method: A new method for reconstructing phylogenetic trees. *Mol. Biol. Evol.* **1987**, *4*, 406–425. [PubMed]
19. Górecki, P.; Eulenstein, O.; Tiuryn, J. Unrooted tree reconciliation: A unified approach. *IEEE/ACM Trans. Comput. Biol. Bioinform.* **2013**, *10*, 522–536. [CrossRef] [PubMed]