Algorithmica (2005) 293–307
DOI: 10.1007/s00453-004-1147-5

Algorithmica

Rooted Maximum Agreement Supertrees[1]

Jesper Jansson,[2] Joseph H.-K. Ng,[2] Kunihiko Sadakane,[3] and Wing-Kin Sung[2]

Abstract. Given a set $\mathcal{T}$ of rooted, unordered trees, where each $T_i \in \mathcal{T}$ is distinctly leaf-labeled by a set $\Lambda(T_i)$ and where the sets $\Lambda(T_i)$ may overlap, the maximum agreement supertree problem (MASP) is to construct a distinctly leaf-labeled tree Q with leaf set $\Lambda(Q) \subseteq \bigcup_{T_i \in \mathcal{T}} \Lambda(T_i)$ such that $|\Lambda(Q)|$ is maximized and for each $T_i \in \mathcal{T}$, the topological restriction of T_i to $\Lambda(Q)$ is isomorphic to the topological restriction of Q to $\Lambda(T_i)$. Let $n = \left|\bigcup_{T_i \in \mathcal{T}} \Lambda(T_i)\right|$, $k = |\mathcal{T}|$, and $D = \max_{T_i \in \mathcal{T}}\{\deg(T_i)\}$. We first show that MASP with $k = 2$ can be solved in $O(\sqrt{D}n \log(2n/D))$ time, which is $O(n \log n)$ when $D = O(1)$ and $O(n^{1.5})$ when D is unrestricted. We then present an algorithm for MASP with $D = 2$ whose running time is polynomial if $k = O(1)$. On the other hand, we prove that MASP is NP-hard for any fixed $k \geq 3$ when D is unrestricted, and also NP-hard for any fixed $D \geq 2$ when k is unrestricted even if each input tree is required to contain at most three leaves. Finally, we describe a polynomial-time $(n/\log n)$-approximation algorithm for MASP.

Key Words. Phylogenetic tree, Maximum agreement supertree, Rooted triplet, Algorithm, Computational complexity.

1. Introduction.

An important objective in phylogenetics is to develop good methods for merging a collection of phylogenetic trees on overlapping sets of taxa into a single supertree so that no (or as little as possible) branching information is lost. Ideally, the resulting supertree can then be used to deduce evolutionary relationships between taxa which do not occur together in any one of the input trees. Supertree methods are useful because most individual studies investigate relatively few taxa [26] and because sample bias leads to certain taxa being studied much more frequently than others [5]. Also, supertree methods can combine trees constructed for different types of data or under different models of evolution. Furthermore, although computationally expensive methods for constructing reliable phylogenetic trees are infeasible for large sets of taxa, they can be applied to obtain highly accurate trees for smaller, overlapping subsets of the taxa which may then be merged using computationally less intense, supertree-based techniques (see, e.g., [8], [18], and [24]).

Since the set of trees which is to be combined may in practice contain contradictory branching structures (for example, if the trees have been constructed from data originating from different genes or if the experimental data contains errors), a supertree

[1] An extended abstract of this article has appeared in *Proceedings of Latin American Theoretical Informatics* (*LATIN* 2004), pages 499–508, volume 2976 of Lecture Notes in Computer Science, Springer-Verlag, Berlin, 2004.

[2] School of Computing, National University of Singapore, 3 Science Drive 2, Singapore 117543. {jansson,nghonkeo,ksung}@comp.nus.edu.sg.

[3] Department of Computer Science and Communication Engineering, Kyushu University, Hakozaki, Higashi-ku, Fukuoka-shi, 812-8581 Japan. sada@csce.kyushu-u.ac.jp.

Received December 8, 2003; revised December 1, 2004. Communicated by M.-Y. Kao.
Online publication March 14, 2005.

method needs to specify how to resolve conflicts. In this paper we consider *maximum agreement supertrees*. The intuitive idea is to identify and remove a smallest possible subset of the taxa so that the remaining taxa can be combined without conflicts. In this way, one would get an indication of which ancestral relationships can be regarded as resolved and which taxa need to be subjected to further experiments. We formalize the above as a computational problem called *the maximum agreement supertree problem* (MASP).

Further motivation for studying maximum agreement supertrees comes from the relation to a well-studied problem known as *the maximum agreement subtree problem* (MAST) in which the input is a set of leaf-labeled trees and the goal is to compute a tree contained in all of the input trees with as many labeled leaves as possible. Our results in this paper complement those previously known for MAST. The computational complexity of MAST has been closely investigated (see Section 1.2), motivated by the practical usefulness of maximum agreement subtrees. For example, maximum agreement subtrees can be used not only to identify small problematic subsets of taxa during phylogenetic reconstruction, but also to measure the similarity of a given set of trees [10], [13], [23] or to estimate a classification's stability to small changes in the data [13]. Moreover, MAST-based algorithms have been used to prepare and improve bilingual context-using dictionaries for automated language translation systems [9], [25].

1.1. *Problem Definitions*. Let T be a tree whose leaves are labeled by a set S. We say that T is *distinctly leaf-labeled by* S if no two leaves in T have the same label. Below, each leaf in such a tree is identified with its corresponding label in S. Given a rooted, unordered, leaf-labeled tree T and a set S', *the topological restriction of* T *to* S' (denoted by $T \mid S'$) is the tree obtained by deleting from T all nodes which are not on any path from the root to a leaf in S' along with their incident edges, and then contracting every edge between a node having just one child and its child (see Figure 1.1). For any tree T, denote its set of leaves by $\Lambda(T)$.

Let $\mathcal{T} = \{T_1, T_2, \ldots, T_k\}$ be a set of rooted, unordered trees, where each T_i is distinctly leaf-labeled and where the sets $\Lambda(T_i)$ may overlap. We write $S = \bigcup_{T_i \in \mathcal{T}} \Lambda(T_i)$ and call S *the leaf set of* $\mathcal{T}$. A *total agreement supertree of* $\mathcal{T}$ is a distinctly leaf-labeled tree Q such that $\Lambda(Q) = S$ and $Q \mid \Lambda(T_i)$ is isomorphic to T_i for every $T_i \in \mathcal{T}$. Note that two or more trees in $\mathcal{T}$ may contain conflicting branching information, in which case a total agreement supertree of $\mathcal{T}$ does not exist. *The total agreement supertree*

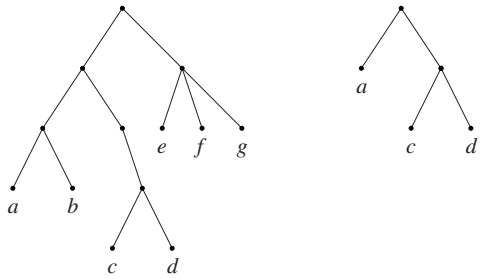

Fig. 1.1. Let T be the tree on the left. Then $T \mid \{a, c, d, h\}$ is the tree shown on the right.

problem (TASP) is: Given a set $\mathcal{T}$ of distinctly leaf-labeled, rooted, unordered trees, output a total agreement supertree of $\mathcal{T}$ if one exists, otherwise output null.

For any $S' \subseteq S$, we let $\mathcal{T} \mid S'$ denote the set $\{T_1 \mid S', T_2 \mid S', \ldots, T_k \mid S'\}$. If there exists a total agreement supertree Q of $\mathcal{T} \mid S'$ then we say that S' is *consistent with* $\mathcal{T}$ and we call Q an *agreement supertree of* $\mathcal{T}$. (Thus, Q is an agreement supertree of $\mathcal{T}$ if and only if $\Lambda(Q) \subseteq S$ and $Q \mid \Lambda(T_i)$ is isomorphic to $T_i \mid \Lambda(Q)$ for every $T_i \in \mathcal{T}$.) A *maximum agreement supertree of* $\mathcal{T}$ is an agreement supertree of $\mathcal{T}$ with as many leaves as possible. *The maximum agreement supertree problem* (MASP) is: Given a set $\mathcal{T}$ of distinctly leaf-labeled, rooted, unordered trees, output a maximum agreement supertree of $\mathcal{T}$.

An *agreement subtree of* $\mathcal{T}$ is a distinctly leaf-labeled tree U such that $\Lambda(U) \subseteq S$ and U is isomorphic to $T_i \mid \Lambda(U)$ for every $T_i \in \mathcal{T}$. A *maximum agreement subtree of* $\mathcal{T}$ is an agreement subtree of $\mathcal{T}$ with the maximum possible number of leaves. *The maximum agreement subtree problem* (MAST)[4] is to find a maximum agreement subtree of $\mathcal{T}$.

Throughout this paper, we let n denote the cardinality of the leaf set and k the number of input trees, i.e., $n = \left|\bigcup_{T_i \in \mathcal{T}} \Lambda(T_i)\right|$ and $k = |\mathcal{T}|$ in the problem definitions above. Furthermore, we let $D = \max_{T_i \in \mathcal{T}}\{\deg(T_i)\}$, where $\deg(T_i)$ is the degree[5] of T_i. We assume that none of the trees in $\mathcal{T}$ has a node with degree 1, so that each tree contains $O(n)$ nodes. (Given an instance with one or more degree 1 nodes, we can replace T_i by $T_i \mid S$ for all $T_i \in \mathcal{T}$ in total time which is linear in the input size.) Note that if we are given a subset S' of S which is consistent with $\mathcal{T}$, then we can efficiently construct a total agreement supertree of $\mathcal{T} \mid S'$ using the algorithm for TASP by Henzinger et al. [18] (see Section 1.2 below and also Lemma 5.1 in Section 5). Hence, in the remainder of this paper, we focus on the subproblem of MASP of computing a maximum cardinality subset S' of S such that S' is consistent with $\mathcal{T}$.

A distinctly leaf-labeled, binary, rooted, unordered tree with three leaves is called a *rooted triplet*. The unique rooted triplet on leaf set $\{x, y, z\}$ in which the lowest common ancestor of x and y is a proper descendant of the lowest common ancestor of x and z (or, equivalently, where the lowest common ancestor of x and y is a proper descendant of the lowest common ancestor of y and z) is denoted by $(\{x, y\}, z)$. For example, $(\{c, d\}, a)$ denotes the rooted triplet on the right in Figure 1.1.

1.2. *Previous Results.* Comprehensive surveys of existing methods for constructing supertrees can be found in [5], [24], and [26]. Below, we mention some known results related to MASP.

Aho et al. [1] presented an algorithm which can be used to solve TASP in $O(kn)$ time when all trees in $\mathcal{T}$ are rooted triplets. Several years later, Henzinger et al. [18] showed how to modify the algorithm to solve TASP for any $\mathcal{T}$ in $\min\{O(Nn^{0.5}), O(N + n^2 \log n)\}$ time,[6] where $N = \sum_{T_i \in \mathcal{T}} |T_i|$ is the total number of nodes in $\mathcal{T}$. In contrast, the analog of TASP for *unrooted* trees is NP-hard, even if all of the input trees are

[4] MAST is also referred to in the literature as *the maximum homeomorphic subtree problem* (MHT).

[5] The *degree of a node* u in a rooted tree is the number of children of u. The *degree of a rooted tree* T is the maximum degree of all nodes in T.

[6] By replacing the deterministic algorithm for dynamic graph connectivity used by Henzinger et al., the running time is improved to $\min\{O(N \log^2 n), O(N + n^2 \log n)\}$. See Lemma 5.1 in Section 5.

quartets (distinctly leaf-labeled, unrooted trees each having four leaves and no nodes with precisely two neighbors) [27]. A polynomial-time algorithm for computing an unrooted total agreement supertree if one exists (in fact, the best such tree according to one of four optimization criteria) when all k input trees are binary and $k = O(1)$ was given by Bryant in [7].

The computational complexity of MAST has been studied extensively (e.g., [3], [6], [9]–[13], [16], [17], [21]–[23], [28]). Today, the fastest known algorithm for MAST for two trees, invented by Kao et al. [23], runs in $O\big(\sqrt{D}n\log(2n/D)\big)$ time, which is $O(n\log n)$ when $D = O(1)$ and $O(n^{1.5})$ when D is unrestricted.

Amir and Keselman [3] considered the case of $k \geq 3$ input trees. They proved that MAST is NP-hard for three trees with unrestricted degrees, but solvable in polynomial time for three or more trees if the degree of at least one of the trees is bounded by a constant. For the latter case, Farach et al. [10] gave an algorithm with improved efficiency running in $O(kn^3 + n^d)$ time, where d is an upper bound on at least one of the input trees' degrees; Bryant [6] proposed a conceptually different algorithm with the same running time.

Hein et al. [17] proved the following inapproximability result: MAST for three trees with unrestricted degrees cannot be approximated within a factor of $2^{\log^{\delta} n}$ in polynomial time for any constant $\delta < 1$, unless NP $\subseteq$ DTIME$[2^{\text{polylog}\,n}]$. Gąsieniec et al. [16] proved that MAST cannot be approximated within a factor of n^{ε} for any constant ε where $0 \leq \varepsilon < \frac{1}{9}$ in polynomial time unless P = NP, even for instances containing only trees of height 2, and showed that if the number of trees is bounded by a constant and all the input trees' heights are bounded by a constant then MAST can be approximated within a constant factor in $O(n\log n)$ time.

A problem related to MASP and MAST is *the maximum refinement subtree problem* (MRST). Its goal is to construct a tree W with $\Lambda(W) \subseteq S$ which maximizes $|\Lambda(W)|$ such that for each $T_i \in \mathcal{T}$, $T_i \mid \Lambda(W)$ can be obtained from W by applying a series of edge contractions. MRST is NP-hard for $k = 2$ if D is unrestricted [17] but solvable in polynomial time if $k = O(1)$ and $D = O(1)$ [14].

Another related problem is *the maximum compatible subset of rooted triplets problem* (MCSR) in which the input is a set $\mathcal{T}$ of rooted triplets and the objective is to find a $\mathcal{T}' \subseteq \mathcal{T}$ of maximum cardinality such that there exists a total agreement supertree of $\mathcal{T}'$. MCSR is NP-hard [6], [20]; two polynomial-time approximation algorithms for MCSR were given in [16].

1.3. *Our Results and Organization of the Paper.* In Section 2 we make use of known positive and negative results for MAST to obtain an efficient algorithm for MASP restricted to $k = 2$ and an NP-hardness proof for MASP restricted to any fixed $k \geq 3$, respectively. The algorithm for $k = 2$ runs in $O\big(\sqrt{D}\,n\log(2n/D)\big)$ time, which is $O(n\log n)$ when $D = O(1)$ and $O(n^{1.5})$ when D is unrestricted. Then, in Section 3 we present a more complex MAST-based algorithm for solving MASP with $D = 2$. It runs in $O(k(2n^2)^{3k^2})$ time, which is polynomial when $k = O(1)$. In Section 4 we prove that MASP is NP-hard even if all of the input trees are required to be rooted triplets (i.e., $D = 2$ and k is unrestricted). Finally, in Section 5, we describe a simple polynomial-time approximation algorithm for MASP which is guaranteed to find an approximate solution of size at least $(\log n)/n$ times the number of leaves in an optimal solution.

2. Preliminaries. We first investigate the close relationship between MASP and MAST.

LEMMA 2.1. *For any set $\mathcal{T} = \{T_1, T_2, \ldots, T_k\}$ of distinctly leaf-labeled, rooted, unordered trees such that $\Lambda(T_1) = \Lambda(T_2) = \cdots = \Lambda(T_k)$, an optimal solution to MASP for $\mathcal{T}$ is an optimal solution to MAST for $\mathcal{T}$ and vice versa.*

PROOF. Write $S = \Lambda(T_1) = \Lambda(T_2) = \cdots = \Lambda(T_k)$, let Q be any agreement supertree of $\mathcal{T}$, and let $S' = \Lambda(Q)$. Then, by definition, $Q \mid \Lambda(T_i \mid S') = T_i \mid S'$ for every $T_i \in \mathcal{T}$. Now, $\Lambda(T_i \mid S') = S \cap S' = S'$, so $T_i \mid S' = Q \mid S' = Q$ for every $T_i \in \mathcal{T}$, which means that Q is an agreement subtree of $\mathcal{T}$. Conversely, let U be an agreement subtree of $\mathcal{T}$ whose leaves are distinctly labeled by some set S'. For every $T_i \in \mathcal{T}$, we have $T_i \mid S' = U$. Then $U \mid \Lambda(T_i \mid S') = (T_i \mid S') \mid \Lambda(T_i \mid S') = T_i \mid S'$ for every $T_i \in \mathcal{T}$, i.e., U is an agreement supertree of $\mathcal{T}$.

Now, suppose X is an optimal solution to MASP for $\mathcal{T}$ but not to MAST for $\mathcal{T}$. Then any optimal solution Y to MAST for $\mathcal{T}$ is a better solution to MASP for $\mathcal{T}$ than X. Contradiction. The other case is analogous. □

THEOREM 2.2. *MASP with $k = 2$ can be solved in $O\left(\sqrt{D}\, n \log(2n/D)\right)$ time.*

PROOF. Given an instance $\mathcal{T} = \{T_1, T_2\}$ of MASP with $k = 2$, let $L = \Lambda(T_1) \cap \Lambda(T_2)$ and run the algorithm of Kao et al. [23] on the instance $\mathcal{T} \mid L$ to obtain a maximum agreement subtree U of $\mathcal{T} \mid L$. This takes $O\left(\sqrt{D}\, n \log(2n/D)\right)$ time. By Lemma 2.1, U is also a maximum agreement supertree of $\mathcal{T} \mid L$. Next, for every leaf which appears in exactly one of T_1 and T_2, insert it into U according to its position in T_1 or T_2. More precisely, let $X = L \backslash \Lambda(U)$ and first compute $T_1' = T_1 \mid (\Lambda(T_1) \backslash X)$ and $T_2' = T_2 \mid (\Lambda(T_2) \backslash X)$ in $O(n)$ time. For any node $u \in U$, let $T_1'(u)$ and $T_2'(u)$ be the nodes in T_1' and T_2' respectively corresponding to u. Construct a tree Q as follows: initially, set $Q = T_1'$, then for each edge (u, v) of U, where we assume u is the parent of v, replace the edge in Q between $T_1'(v)$ and its parent with the path in T_2' between $T_2'(v)$ and $T_2'(u)$. Q can be constructed using a total of $O(n)$ time.

Q is an agreement supertree of $\mathcal{T}$ because $Q \mid \Lambda(T_1) = T_1' = T_1 \mid (\Lambda(T_1) \backslash X) = T_1 \mid \left((\Lambda(T_1) \backslash \Lambda(T_2)) \cup \Lambda(U)\right) = T_1 \mid \Lambda(Q)$, where the last equality follows from the relation $\Lambda(Q) = \Lambda(U) \cup (\Lambda(T_1) \backslash \Lambda(T_2)) \cup (\Lambda(T_2) \backslash \Lambda(T_1))$. In the same way, $Q \mid \Lambda(T_2) = T_2 \mid \Lambda(Q)$. Furthermore, Q is a *maximum* agreement supertree of $\mathcal{T}$ because otherwise there would exist an agreement supertree Q^* of $\mathcal{T}$ with more leaves than Q, and then $Q^* \mid L$ is an agreement subtree of $\mathcal{T} \mid L$ by the proof of Lemma 2.1 and $Q^* \mid L$ has $|\Lambda(Q^*) \cap L| > |\Lambda(U)|$ leaves, which contradicts that U is a maximum agreement subtree of $\mathcal{T} \mid L$. □

The running time given in Theorem 2.2 is $O(n \log n)$ for two trees whose degrees are bounded by a constant and $O(n^{1.5})$ for two trees with unrestricted degrees.

THEOREM 2.3. *For any fixed $k \geq 3$, MASP with unrestricted D is NP-hard.*

PROOF. Let $\mathcal{T}$ be an arbitrary instance of MAST. Any leaf which does not belong to $L = \Lambda(T_1) \cap \Lambda(T_2) \cap \cdots \cap \Lambda(T_k)$ cannot appear in an agreement subtree of $\mathcal{T}$, so a maximum agreement subtree of $\mathcal{T} \mid L$ is also a maximum agreement subtree of $\mathcal{T}$. The former is in turn equal to a maximum agreement supertree of $\mathcal{T} \mid L$ by Lemma 2.1. Hence, an algorithm for solving MASP could be used to solve MAST. The theorem now follows from the NP-hardness of MAST for any fixed $k \geq 3$ when D is unrestricted [3]. □

The proof of Theorem 2.3 also implies that the inapproximability results of [16] and [17] for MAST mentioned in Section 1.2 hold for MASP as well.

3. A Polynomial-Time Algorithm for MAST with $D = 2$ and $k = O(1)$. In this section we show how MASP restricted to $D = 2$ can be reduced to MAST for a set of k distinctly leaf-labeled binary trees having $O((2n)^{k^2})$ leaves. Thus, we can solve MASP with $D = 2$ in polynomial time if $k = O(1)$.

Without loss of generality, assume that every $a \in S$ appears in at least two trees in $\mathcal{T}$. (If a appears in exactly one tree in $\mathcal{T}$, we can obtain a maximum agreement supertree of $\mathcal{T}$ as follows: (1) remove a from $\mathcal{T}$; (2) compute a maximum agreement supertree T' for the modified $\mathcal{T}$; and (3) insert a into T' according to its position in the original $\mathcal{T}$, as described in the proof of Theorem 2.2 above.)

Our transformation consists of two steps. MASP is first transformed to MAST for non-distinctly leaf-labeled trees; then the latter problem is transformed to MAST. Here, by a *homeomorphic subtree* of a (distinctly or non-distinctly) leaf-labeled tree R, we mean a tree which is isomorphic to $R \mid L$ for some subset L of the leaves of R, and by an agreement subtree of a set $\mathcal{R} = \{R_1, R_2, \ldots, R_k\}$ of non-distinctly leaf-labeled trees, we mean a distinctly leaf-labeled tree which is a homeomorphic subtree of every $R_i \in \mathcal{R}$. *MAST for non-distinctly leaf-labeled trees* is defined as: Given a set $\mathcal{R}$ of k rooted, unordered trees which are leaf-labeled by a set S and where each leaf label may occur several times in each tree, find an agreement subtree of $\mathcal{R}$ with as many leaves as possible.

We now describe the first step, i.e., the transformation from MASP to MAST for a set $\mathcal{R} = \{R_1, R_2, \ldots, R_k\}$ of non-distinctly leaf-labeled binary trees. For every $T_i \in \mathcal{T}$, construct R_i using the following routine:

1. Set $R_{i,0} = T_i$ and $L_0 = \Lambda(T_i)$.
2. For $j = 1$ to k do
   (a) Let $L_j = L_{j-1} \cup \Lambda(T_j)$ and let $U = T_j \mid (L_j \backslash L_{j-1})$.
   (b) Initially, set $R_{i,j} = R_{i,j-1}$. Attach $|\Lambda(U)|$ copies of U to every edge of $R_{i,j}$. Next, let r be a new node having the current root of $R_{i,j}$ and the root of another copy of U as its two children, attach $|\Lambda(U)| - 1$ copies of U to the edge between r and the root of $R_{i,j}$, and make r the new root of $R_{i,j}$.
3. Set $R_i = R_{i,k}$.

See Figures 3.1 and 3.2 for an example. For every $i \in \{1, 2, \ldots, k\}$, any leaf in $\Lambda(T_i)$ appears exactly once in R_i whereas multiple copies of all leaves not in $\Lambda(T_i)$ are inserted

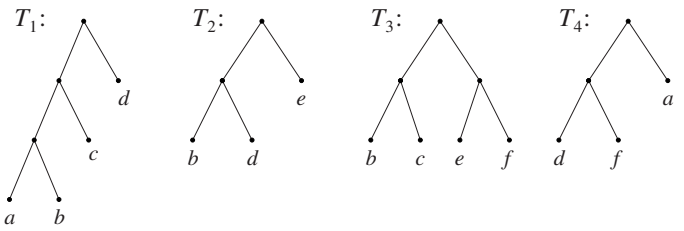

Fig. 3.1. Four input trees T_1, T_2, T_3, and T_4.

at all potential positions at which they might appear in an MASP (see Lemma 3.2). Based on the above construction, T_i is a homeomorphic subtree of R_i. In addition, the set $\mathcal{R}$ has the following properties.

LEMMA 3.1. *For every $R_i \in \mathcal{R}$, the number of leaves in R_i is at most $(2n^2)^k$.*

PROOF. Fix $i \in \{1, 2, \ldots, k\}$, write $s_j = |L_j|$ for all $0 \leq j \leq k$, and let t_j be the number of leaves of $R_{i,j}$. We claim that $t_j \leq (2s_j^2)^{j+1}$ if $j < i$ and that $t_j \leq (2s_j^2)^j$ if $j \geq i$.

When $j = 0$, $t_j = |\Lambda(T_i)| \leq 2s_0^2$. In iteration $j \geq 1$, $R_{i,j-1}$ has $2t_{j-1} - 2$ edges and the number of leaves of U is $s_j - s_{j-1}$, so at most $(2t_{j-1} - 1) \cdot (s_j - s_{j-1})$ copies of U are inserted to obtain $R_{i,j}$, which gives us $t_j \leq t_{j-1} + 2t_{j-1} \cdot (s_j - s_{j-1})^2 \leq 2t_{j-1} \cdot s_j^2$.

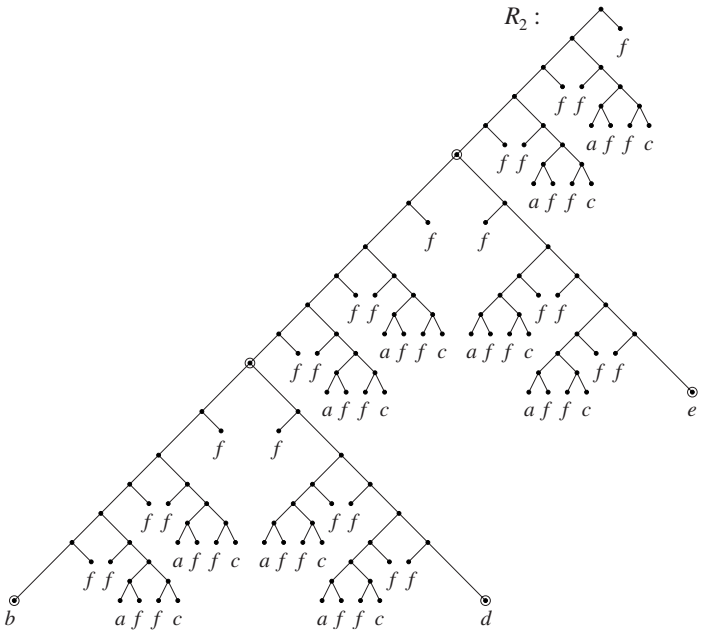

Fig. 3.2. $R_2 (= R_{2,4})$ is constructed from the trees in Figure 3.1. The marked nodes are the ones that originate from T_2.

In iteration i, no leaves are inserted, i.e., $s_i - s_{i-1} = 0$, and we have $t_i = t_{i-1}$. Now, the claim follows by induction on j and using the fact that $s_{j-1} \leq s_j$ for all j, together with $t_i = t_{i-1}$.

Finally, $s_k = n$ gives $t_k \leq (2n^2)^k$. □

LEMMA 3.2. *For any tree X which is distinctly leaf-labeled by some $S' \subseteq S$, X is an agreement supertree of $\mathcal{T}$ if and only if X is an agreement subtree of $\mathcal{R}$.*

PROOF. ($\leftarrow$) Suppose X is an agreement subtree of $\mathcal{R}$ with $\Lambda(X) = S'$. For any $R_i \in \mathcal{R}$ and $S^* \subseteq S$, denote by $R_i \| S^*$ the set of all homeomorphic subtrees of R_i which are distinctly leaf-labeled by S^*. Observe that $R_i \| \Lambda(T_i) = \{T_i\}$. For every i, $X \in R_i \| S'$ by the definition of agreement subtree, so $X \mid \Lambda(T_i)$ belongs to $(R_i \| S') \mid \Lambda(T_i) = R_i \| (S' \cap \Lambda(T_i)) = (R_i \| \Lambda(T_i)) \mid S' = \{T_i \mid S'\}$, i.e., $X|\Lambda(T_i) = T_i|\Lambda(X)$. This shows that X is an agreement supertree of $\mathcal{T}$.

($\rightarrow$) Suppose X is an agreement supertree of $\mathcal{T}$. Consider a fixed i. We prove that $X \mid L_j$ is a homeomorphic subtree of $R_{i,j}$ for every $j \in \{0, 1, 2, \ldots, k\}$. When $j = 0$, $X \mid L_0 = X \mid \Lambda(T_i) = T_i \mid \Lambda(X)$ is a homeomorphic subtree of $T_i = R_{i,0}$. Next, assume $X \mid L_{j-1}$ is a homeomorphic subtree of $R_{i,j-1}$. Let $\mathcal{A}$ be the set of maximal subtrees of $X \mid L_j$ induced by the set of nodes which are not on any path between two leaves from $X \mid L_{j-1}$. Each $A \in \mathcal{A}$ is a homeomorphic subtree of U (this is because A is a homeomorphic subtree of $X \mid (L_j \backslash L_{j-1})$ and therefore a homeomorphic subtree of $X \mid \Lambda(T_j) = T_j \mid \Lambda(X)$, so we have $A = (T_j \mid \Lambda(X)) \mid \Lambda(A) = T_j \mid \Lambda(A)$; furthermore, $\Lambda(A) \subseteq (L_j \backslash L_{j-1})$ which means that A is a homeomorphic subtree of $T_j|(L_j \backslash L_{j-1}) = U$). Then, since $|\Lambda(U)|$ copies of U are attached to each edge of $R_{i,j-1}$ and above the root to obtain $R_{i,j}$ and since $\mathcal{A}$ contains at most $|\Lambda(U)|$ elements, it follows that $X \mid L_j$ must be a homeomorphic subtree of $R_{i,j}$. Therefore, $X \mid L_k = X$ is a homeomorphic subtree of $R_{i,k} = R_i$ for every $i = 1, 2, \ldots, k$, i.e., X is an agreement subtree of $\mathcal{R}$. □

Next, we transform MAST for the set $\mathcal{R}$ of non-distinctly leaf-labeled binary trees to MAST for a set $\mathcal{P} = \{P_1, P_2, \ldots, P_k\}$ of binary trees which are distinctly leaf-labeled by a set $\left\{a^1_{b_1,b_2,\ldots,b_k}, a^2_{b_1,b_2,\ldots,b_k} \mid a \in S, 1 \leq i \leq k, 1 \leq b_i \leq \gamma_a[i]\right\}$, where $\gamma_a[i]$ is the number of occurrences of leaf label a in R_i.

To describe the transformation, we need some additional notation. For every $a \in S$, define $a([b_1..d_1], [b_2..d_2], \ldots, [b_k..d_k])$, where b_i and d_i are positive integers with $b_i \leq d_i$ for all $1 \leq i \leq k$, to be a rooted caterpillar[7] with $2\prod_{i=1}^{k}(d_i - b_i + 1)$ leaves labeled (in order of non-decreasing distance from the root) by $a^1_{b_1,b_2,\ldots,b_k}, a^2_{b_1,b_2,\ldots,b_k}, a^1_{b_1,b_2,\ldots,b_k+1}, a^2_{b_1,b_2,\ldots,b_k+1}, \ldots, a^1_{d_1,d_2,\ldots,d_k}, a^2_{d_1,d_2,\ldots,d_k}$. Define $\bar{a}([b_1..d_1], [b_2..d_2], \ldots, [b_k..d_k])$ as the rooted caterpillar $a([b_1..d_1], [b_2..d_2], \ldots, [b_k..d_k])$ but where the leaf ordering has been reversed. For every leaf in R_i labeled by a, such a leaf is called *the jth occurrence of a in R_i* if, according to pre-order traversal of R_i, it is the jth visited leaf which is labeled by a.

[7] A *rooted caterpillar* is a rooted, unordered, distinctly leaf-labeled binary tree where every internal node has at least one child which is a leaf.

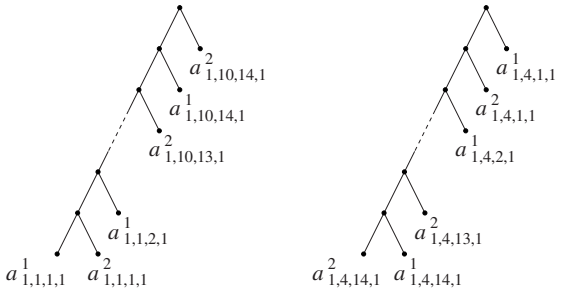

Fig. 3.3. For the problem instance shown in Figure 3.1, we obtain $\gamma_a[1] = 1$, $\gamma_a[2] = 10$, $\gamma_a[3] = 14$, and $\gamma_a[4] = 1$. Hence, when constructing P_1, the single occurrence of leaf a in R_1 will be replaced by the caterpillar $\bar{a}([1..1], [1..10], [1..14], [1..1])$ shown on the left. Similarly, when constructing P_2, the fourth occurrence of leaf a in R_2 will be replaced by the caterpillar $a([1..1], [4..4], [1..14], [1..1])$ shown on the right.

For $i = 1, 2, \ldots, k$, the tree P_i is constructed from R_i by replacing, for every $a \in S$, the leaves labeled by a with caterpillars $a()$ or $\bar{a}()$ as follows (see Figure 3.3 for an example):

1. Set $P_i = R_i$.
2. For every $a \in S$,
   — if T_i is the first tree containing a among $T_1, T_2, \ldots, T_i$, then (in this case, P_i contains exactly one a, that is, $\gamma_a[i] = 1$) replace a in P_i by the caterpillar $\bar{a}([1..\gamma_a[1]], \ldots, [1..\gamma_a[i-1]], [1..1], [1..\gamma_a[i+1]], \ldots, [1..\gamma_a[k]])$;
   — else for $j = 1, 2, \ldots, \gamma_a[i]$, replace the jth occurrence of a in P_i by the caterpillar $a([1..\gamma_a[1]], \ldots, [1..\gamma_a[i-1]], [j..j], [1..\gamma_a[i+1]], \ldots, [1..\gamma_a[k]])$.

Based on the construction, it is easy to check that all trees P_i are distinctly labeled by $\{a^1_{b_1,b_2,\ldots,b_k}, a^2_{b_1,b_2,\ldots,b_k} \mid a \in S, 1 \le i \le k, 1 \le b_i \le \gamma_a[i]\}$. Also, for every label $a \in S$, there exists exactly one tree P_i which contains the caterpillar $\bar{a}()$ while the rest of the trees in $\mathcal{P}$ contain caterpillars of the form $a()$. Below, more properties of $\mathcal{P}$ are described.

LEMMA 3.3. *For every P_i, $|\Lambda(P_i)| = O((2n^2)^{k^2})$.*

PROOF. For every $a \in S$ and $1 \le i \le k$, $\gamma_a[i] \le (2n^2)^k$ by Lemma 3.1. Note that P_i is created by replacing every leaf in R_i by a caterpillar having at most $2 \cdot \gamma_a[1] \cdots \gamma_a[i-1] \cdot \gamma_a[i+1] \cdots \gamma_a[k] = O(2^{1+k(k-1)} \cdot n^{2k(k-1)})$ leaves. By Lemma 3.1, each R_i contains $O((2n^2)^k)$ leaves. The lemma follows. □

LEMMA 3.4. *For any $a \in S$, an MAST of $\mathcal{P}$ has at most two leaves of the form $a^\ell_{b_1,b_2,\ldots,b_k}$.*

PROOF. By the problem definition, the label a appears in at least two trees, say, T_i and T_j where $i < j$. By our construction, all leaves of the form $a^\ell_{b_1,b_2,\ldots,b_k}$ in P_i must appear in exactly one caterpillar $\bar{a}([1..\gamma_a[1]], \ldots, [1..\gamma_a[i-1]], [1..1], [1..\gamma_a[i+1]], \ldots, [1..\gamma_a[k]])$ while they appear in caterpillars of the form $a()$ in P_j. Since the leaves belonging to two

caterpillars $a()$ and $\bar{a}()$ are in reverse order relative to each other, any agreement subtree of P_i and P_j can contain at most two leaves of the form $a^{\ell}_{b_1,b_2,\ldots,b_k}$. As an MAST of $\mathcal{P}$ must be an agreement subtree of P_i and P_j, the lemma follows. □

LEMMA 3.5. *For any integer x, the size of the MAST of $\mathcal{R}$ is $\geq x$ if and only if the size of the MAST of $\mathcal{P}$ is $\geq 2x$.*

PROOF. ($\rightarrow$) Suppose X is an agreement subtree of $\mathcal{R}$ such that $|\Lambda(X)| \geq x$. Then X is a homeomorphic subtree of R_i for $i = 1, 2, \ldots, k$. For every leaf ℓ in X which is labeled by some $a \in S$, we associate with it a vector $(j_1^a, j_2^a, \ldots, j_k^a)$ such that, for any i, ℓ is the j_i^ath occurrence of a in R_i according to pre-order traversal. We construct Y by replacing the leaf ℓ in X by a tree with two child leaves which are labeled by $a^1_{j_1^a,j_2^a,\ldots,j_k^a}$ and $a^2_{j_1^a,j_2^a,\ldots,j_k^a}$. It is easily checked that Y is an agreement subtree of $\mathcal{P}$, whose size is $\geq 2x$.

($\leftarrow$) Let Y be is a maximum agreement subtree of $\mathcal{P}$ such that $|\Lambda(Y)| \geq 2x$. Let $L'(a) = \{a^{\ell}_{b_1,b_2,\ldots,b_k} \in \Lambda(Y)\}$ and $\mathcal{L} = \{L'(a) \mid |L'(a)| \neq 0 \text{ and } a \in S\}$. By Lemma 3.4, the size of each set $L'(a)$ in $\mathcal{L}$ is at most two. Suppose $\mathcal{L}$ contains at most $x - 1$ sets. Then $|\Lambda(Y)| = \sum_{L' \in \mathcal{L}} |L'| \leq (x-1)2 < 2x \leq |\Lambda(Y)|$, which is a contradiction, so $\mathcal{L}$ contains at least x sets. For each $L' \in \mathcal{L}$, we choose a particular $a^{\ell}_{b_1,b_2,\ldots,b_k}$ and denote it by $ch(L')$. Let $L_{Y'} = \bigcup_{L' \in \mathcal{L}} ch(L')$ and $Y' = Y|L_{Y'}$. Then $|L_{Y'}| = |\mathcal{L}| \geq x$. Relabel every leaf $a^{\ell}_{b_1,b_2,\ldots,b_k}$ in Y' by a and generate a tree X. It is easy to verify that X is a homeomorphic subtree of $R_1, R_2, \ldots, R_k$. This direction follows. □

A maximum agreement supertree of $\mathcal{T}$ can now be computed by applying the algorithm of Bryant [6] or Farach et al. [10] (see Section 1.2) to $\mathcal{P}$. Since the number of leaves in $\mathcal{P}$ is $O((2n^2)^{k^2})$ and all trees are binary, we obtain the main theorem of this section.

THEOREM 3.6. *Given a set of k binary trees $\mathcal{T}$ which are labeled by n distinct labels, their maximum agreement supertree can be computed in $O(k(2n^2)^{3k^2})$ time.*

4. MASP with $D = 2$ Is NP-Hard. Theorem 2.3 states that MASP is an NP-hard problem for any fixed $k \geq 3$ when D is unrestricted. We now show that MASP remains NP-hard if restricted to instances with $D = 2$ but where k is left unrestricted. In fact, we prove that MASP is NP-hard even if all of the input trees are required to be rooted triplets. Our NP-hardness proof consists of a polynomial-time reduction from the independent set problem which is known to be NP-hard (see, e.g., [15]).

THE INDEPENDENT SET PROBLEM.

Instance: An undirected graph $G = (V, E)$ and a positive integer $I \leq |V|$.
Question: Is there a subset V' of V with $|V'| = I$ such that V' is an independent set, i.e., such that no two vertices in V' are joined by an edge in E?

THE MAXIMUM AGREEMENT SUPERTREE PROBLEM RESTRICTED TO ROOTED TRIPLETS, DECISION PROBLEM VERSION (MASPR-D).

Instance: A set $\mathcal{T}$ of rooted triplets with leaf set S and a positive integer $K \leq |S|$.
Question: Is there a subset S' of S with $|S'| = K$ which is consistent with $\mathcal{T}$?

THEOREM 4.1. *MASP is NP-hard even if restricted to rooted triplets.*

PROOF. Given an arbitrary instance (G, I) of the independent set problem, construct an instance of MASPR-d as follows. Let $S = V \cup \{z_e \mid e \in E\}$ and set $K = I + |E|$. For each edge e in E, include the two rooted triplets $(\{a, z_e\}, b)$ and $(\{b, z_e\}, a)$ in $\mathcal{T}$, where $e = \{a, b\}$.

CLAIM. *G has an independent set of size I if and only if there exists a subset S' of S of size K which is consistent with $\mathcal{T}$.*

PROOF OF CLAIM. Suppose there exists an independent set W in G of size I. Let $S' = W \cup \{z_e \mid e \in E\}$. Then $\mathcal{T} \mid S'$ contains no rooted triplets (if $\mathcal{T} \mid S'$ contained some rooted triplet $(\{x, z_{\{x,y\}}\}, y)$ then x and y would have to be joined by an edge in E and thus could not both belong to W, which is a contradiction) so the tree consisting of a root node with $|S'|$ children distinctly labeled by S' is a total agreement supertree of $\mathcal{T} \mid S'$. Thus, S' is consistent with $\mathcal{T}$ and $|S'| = I + |E| = K$.

Conversely, suppose there exists a consistent subset S' of S of size K. For each $\{x, y\} \in E$, if $z_{\{x,y\}} \notin S'$ but at least one of x and y belongs to S' then replace x or y in S' by $z_{\{x,y\}}$, and if none of x, y, and $z_{\{x,y\}}$ is contained in S' then replace any element in S' belonging to V by $z_{\{x,y\}}$ (such an element always exists because $K > |E|$). The resulting set S'' will have the form $W \cup \{z_e \mid e \in E\}$ with $W \subseteq V$ and $|S''| = K$, and will still be consistent with $\mathcal{T}$. Next, observe that by the construction of $\mathcal{T}$, for each $\{x, y\} \in E$ at most two of x, y, and $z_{\{x,y\}}$ can be included in any subset of S which is consistent with $\mathcal{T}$. Therefore, for each $\{x, y\} \in E$, since $z_{\{x,y\}} \in S''$ it holds that S'' cannot contain both x and y. Thus, W is an independent set and $|W| = K - |E| = I$. □

Hence, MASPR-d is NP-hard and it follows that MASP restricted to rooted triplets is NP-hard. □

5. A Polynomial-Time $(n/\log n)$-Approximation Algorithm. By the comments following Theorem 2.3, it is highly unlikely that MASP in its general form can be solved exactly or even approximated efficiently (say, within a constant factor) in polynomial time. However, we can adapt one of Akutsu and Halldórsson's approximation algorithms for the largest common subtree problem in [2] to obtain the following polynomial-time $(n/\log n)$-approximation algorithm for MASP:

> Arbitrarily partition S into $\lfloor n/\log n \rfloor$ disjoint sets $S_1, S_2, \ldots, S_{\lfloor n/\log n \rfloor}$, each of size at most $\lceil \log n \rceil + 1$. Then check every subset S_i' of every set S_i to see if S_i' is consistent with $\mathcal{T}$, and let Z be one such subset of maximum cardinality. Return Z.

To see that this algorithm always returns a solution with at least $(\log n)/n$ times the number of leaves in an optimal solution, let S^* be a maximum consistent leaf subset. Because of the pigeonhole principle, at least one of $S_1, S_2, \ldots, S_{\lfloor n/\log n \rfloor}$ contains $\geq 1/\lfloor n/\log n \rfloor$ of the elements in S^*; thus,

$$|Z| \geq \frac{|S^*|}{\lfloor n/\log n \rfloor} \geq \frac{|S^*|}{n/\log n}.$$

Before describing how to implement the algorithm, we note that the deterministic algorithm for dynamic graph connectivity employed in the algorithm for TASP of Henzinger et al. [18] can be replaced with a more recent one due to Holm et al. [19] to yield the following improvement.

LEMMA 5.1. *TASP is solvable in* $\min\{O(N \log^2 n),\ O(N + n^2 \log n)\}$ *time, where* $N = \sum_{T_i \in \mathcal{T}} |T_i|$ *is the total number of nodes in* $\mathcal{T}$.

PROOF. In the proof of Theorem 1 in [18], if the deterministic fully dynamic graph connectivity algorithm of Holm et al. [19] which takes $O(\log^2 n)$ amortized time per update and which answers each connectivity query in $O(\log n/\log\log n)$ time is used instead, then the m queries and $O(m)$ updates to Algorithm A in [18] cost a total of $O(m \log^2 n)$ time (rather than $O(mn^{1/2})$). □

To implement the approximation algorithm given above, first construct all the sets $\mathcal{T} \mid S_i$ using a total of $O((n/\log n) \cdot kn)$ time. Next, each S_i has at most $2^{\log n + 2} = O(n)$ subsets that need to be considered. Each such subset S_i' can be evaluated by computing $(\mathcal{T} \mid S_i) \mid S_i'$ in $O(k \cdot \log n)$ time and then applying the algorithm in Lemma 5.1; here there are $\widehat{n} = O(\log n)$ leaves and the total size of $\mathcal{T} \mid S_i'$ is $\widehat{N} = O(k \log n)$ so this step takes $X = \min\{O(\widehat{N} \log^2 \widehat{n}),\ O(\widehat{N} + \widehat{n}^2 \log \widehat{n})\} = \min\{O(k \log n \cdot (\log\log n)^2),\ O(k \log n + \log^2 n \cdot \log\log n)\}$ time. The total running time is therefore $O((n/\log n) \cdot kn + (n/\log n) \cdot n \cdot (k \cdot \log n + X)) = O(n^2) \cdot \min\{O(k \cdot (\log\log n)^2),\ O(k + \log n \cdot \log\log n)\}$.

If all of the input trees are rooted triplets, the running time can be further improved because then: (1) all the sets $\mathcal{T} \mid S_i$ can be obtained in $O(k)$ time; (2) each $\mathcal{T} \mid S_i$ contains $O(\log^3 n)$ rooted triplets since each S_i contains at most $\log n + 2$ leaves; and (3) each S_i' can be evaluated in $O(\log^3 n)$ time by testing each of the $O(\log^3 n)$ triplets in $\mathcal{T} \mid S_i$ for membership in $\mathcal{T} \mid S_i'$ and then running the algorithm in Lemma 5.1 (now there are $\widehat{n} = O(\log n)$ leaves and $\widehat{k} = O(\log^3 n)$ triplets, so this takes $\min\{O(\widehat{k} \log^2 \widehat{n}),\ O(\widehat{k} + \widehat{n}^2 \log \widehat{n})\} = O(\log^3 n)$ time). Hence, for this case the total running time is $O(k + (n/\log n) \cdot n \cdot \log^3 n) = O(k + n^2 \log^2 n)$.

THEOREM 5.2. *MASP can be approximated within a factor of* $n/\log n$ *in* $O(n^2) \cdot \min\{O(k \cdot (\log\log n)^2),\ O(k + \log n \cdot \log\log n)\}$ *time. MASP restricted to rooted triplets can be approximated within a factor of* $(n/\log n)$ *in* $O(k + n^2 \log^2 n)$ *time.*

Finally, we remark that MAST can be approximated within a factor of $(n/\log n)$ in $O(kn^2)$ time using the same technique.

Table 1. Summary of results.

MASP	$k = 2$	$k = O(1)$	k unrestricted
$D = 2$	$O(n \log n)$ ($\downarrow$)	$O(k(2n^2)^{3k^2})$ (Theorem 3.6)	NP-hard (Theorem 4.1)
$D = O(1)$	$O(n \log n)$ (Theorem 2.2)	Open	NP-hard ($\uparrow$)
D unrestricted	$O(n^{1.5})$ (Theorem 2.2)	NP-hard (Theorem 2.3)	NP-hard ($\leftarrow$ or $\uparrow$)

6. Concluding Remarks. In Table 1, we summarize our results on how restricting the parameters D and k affects the computational complexity of MASP. Arrows indicate when a result follows directly from another by generalization (e.g., MASP with $D = 2$ and unrestricted k is NP-hard, so the more general case $D = O(1)$ and unrestricted k cannot be any easier) or by specialization (e.g., the algorithm for $D = O(1)$ and $k = 2$ still works for the more restricted case $D = 2$ and $k = 2$).

We have also described a polynomial-time $(n/\log n)$-approximation algorithm for MASP (Theorem 5.2).

It is interesting to note that MASP with $D = 2$ and unrestricted k is NP-hard while, on the other hand, MAST with $D = 2$ and unrestricted k can be solved in $O(kn^3)$ time, i.e., in polynomial time, using the algorithm of Bryant [6] or Farach et al. [10] (see Section 1.2). This means that for certain restrictions on the parameters D and k, MASP and MAST cannot have the same computational complexity unless P = NP. Furthermore, although our results indicate that MASP is computationally harder than MAST, the maximum refinement subtree problem (see Section 1.2) does not seem any easier than MASP since it is NP-hard already for $k = 2$ when D is unrestricted [17].

An open problem is to determine the computational complexity of MASP with $D = O(1)$ and $k = O(1)$. We believe that this case is solvable in polynomial time. We would also like to know if the running time of our algorithm for the case $D = 2$ and $k = O(1)$ can be improved.

Addendum. Independently of this paper, Berry and Nicolas [4] recently obtained results for MASP restricted to $k = 2$ similar to our results presented in Section 2. In addition, they considered the intractability of the general case of MASP in terms of a parameter p that denotes the minimum number of leaves to remove from the input trees so that they become isomorphic, and extended their results to *the maximum refinement supertree problem* (referred to in their paper as *the maximum compatible supertree problem*), which is a natural generalization of the maximum refinement subtree problem.

References

[1] A. V. Aho, Y. Sagiv, T. G. Szymanski, and J. D. Ullman. Inferring a tree from lowest common ancestors with an application to the optimization of relational expressions. *SIAM Journal on Computing*, 10(3):405–421, 1981.

[2] T. Akutsu and M. M. Halldórsson. On the approximation of largest common subtrees and largest common point sets. *Theoretical Computer Science*, 233(1–2):33–50, 2000.

[3] A. Amir and D. Keselman. Maximum agreement subtree in a set of evolutionary trees: metrics and efficient algorithms. *SIAM Journal on Computing*, 26(6):1656–1669, 1997.

[4] V. Berry and F. Nicolas. Maximum agreement and compatible supertrees. In *Proceedings of the* 15^{th} *Annual Symposium on Combinatorial Pattern Matching* (*CPM* 2004), pages 205–219. Volume 3109 of Lecture Notes in Computer Science. Springer-Verlag, Berlin, 2004.

[5] O. Bininda-Emonds, J. Gittleman, and M. Steel. The (super)tree of life: procedures, problems, and prospects. *Annual Review of Ecology and Systematics*, 33:265–289, 2002.

[6] D. Bryant. Building Trees, Hunting for Trees, and Comparing Trees: Theory and Methods in Phylogenetic Analysis. Ph.D. thesis, University of Canterbury, Christchurch, 1997.

[7] D. Bryant. Optimal agreement supertrees. In *Proceedings of the* 1^{st}*International Conference on Biology, Informatics, and Mathematics* (*JOBIM* 2000), pages 24–31. Volume 2066 of Lecture Notes in Computer Science. Springer-Verlag, Berlin, 2001.

[8] B. Chor, M. Hendy, and D. Penny. Analytic solutions for three-taxon ML_{MC} trees with variable rates across sites. In *Proceedings of the* 1^{st}*Workshop on Algorithms in Bioinformatics* (*WABI* 2001), pages 204–213. Volume 2149 of Lecture Notes in Computer Science. Springer-Verlag, Berlin, 2001.

[9] R. Cole, M. Farach-Colton, R. Hariharan, T. Przytycka, and M. Thorup. An $O(n \log n)$ algorithm for the maximum agreement subtree problem for binary trees. *SIAM Journal on Computing*, 30(5):1385–1404, 2000.

[10] M. Farach, T. Przytycka, and M. Thorup. On the agreement of many trees. *Information Processing Letters*, 55:297–301, 1995.

[11] M. Farach and M. Thorup. Fast comparison of evolutionary trees. In *Proceedings of the* 5^{th}*Annual ACM–SIAM Symposium on Discrete Algorithms* (*SODA* '94), pages 481–488, 1994.

[12] M. Farach and M. Thorup. Sparse dynamic programming for evolutionary-tree comparison. *SIAM Journal on Computing*, 26(1):210–230, 1997.

[13] C. R. Finden and A. D. Gordon. Obtaining common pruned trees. *Journal of Classification*, 2:255–276, 1985.

[14] G. Ganapathysaravanabavan and T. Warnow. Finding a maximum compatible tree for a bounded number of trees with bounded degree is solvable in polynomial time. In *Proceedings of the* 1^{st}*Workshop on Algorithms in Bioinformatics* (*WABI* 2001), pages 156–163. Volume 2149 of Lecture Notes in Computer Science. Springer-Verlag, Berlin, 2001.

[15] M. Garey and D. Johnson. *Computers and Intractability – A Guide to the Theory of NP-Completeness*. Freeman, New York, 1979.

[16] L. Gąsieniec, J. Jansson, A. Lingas, and A. Östlin. On the complexity of constructing evolutionary trees. *Journal of Combinatorial Optimization*, 3(2–3):183–197, 1999.

[17] J. Hein, T. Jiang, L. Wang, and K. Zhang. On the complexity of comparing evolutionary trees. *Discrete Applied Mathematics*, 71:153–169, 1996.

[18] M. R. Henzinger, V. King, and T. Warnow. Constructing a tree from homeomorphic subtrees, with applications to computational evolutionary biology. *Algorithmica*, 24(1):1–13, 1999.

[19] J. Holm, K. de Lichtenberg, and M. Thorup. Poly-logarithmic deterministic fully-dynamic algorithms for connectivity, minimum spanning tree, 2-edge, and biconnectivity. *Journal of the ACM*, 48(4):723–760, 2001.

[20] J. Jansson. On the complexity of inferring rooted evolutionary trees. In *Proceedings of the Brazilian Symposium on Graphs, Algorithms, and Combinatorics* (*GRACO* 2001), pages 121–125. Volume 7 of Electronic Notes in Discrete Mathematics. Elsevier, Amsterdam, 2001.

[21] M.-Y. Kao. Tree contractions and evolutionary trees. *SIAM Journal on Computing*, 27(6):1592–1616, 1998.

[22] M.-Y. Kao, T.-W. Lam, T. Przytycka, W.-K. Sung, and H.-F. Ting. General techniques for comparing unrooted evolutionary trees. In *Proceedings of the* 29^{th}*Annual ACM Symposium on the Theory of Computing* (*STOC* '97), pages 54–65, 1997.

[23] M.-Y. Kao, T.-W. Lam, W.-K. Sung, and H.-F. Ting. An even faster and more unifying algorithm for comparing trees via unbalanced bipartite matchings. *Journal of Algorithms*, 40(2):212–233, 2001.

[24] P. Kearney. Phylogenetics and the quartet method. In T. Jiang, Y. Xu, and M. Q. Zhang, editors, *Current Topics in Computational Molecular Biology*, pages 111–133. The MIT Press, Cambridge, MA, 2002.

[25] A. Meyers, R. Yangarber, and R. Grishman. Alignment of shared forests for bilingual corpora. In *Proceedings of the* 16^{th} *International Conference on Computational Linguistics* (*COLING*-96), pages 460–465, 1996.

[26] M. J. Sanderson, A. Purvis, and C. Henze. Phylogenetic supertrees: assembling the trees of life. *TRENDS in Ecology & Evolution*, 13(3):105–109, 1998.

[27] M. Steel. The complexity of reconstructing trees from qualitative characters and subtrees. *Journal of Classification*, 9(1):91–116, 1992.

[28] M. Steel and T. Warnow. Kaikoura tree theorems: computing the maximum agreement subtree. *Information Processing Letters*, 48:77–82, 1993.