Parameterized Algorithms for Maximum Agreement Forest on Multiple Trees*

Feng Shi[1], Jianer Chen[1,2], Qilong Feng[1], and Jianxin Wang[1]

[1] School of Information Science and Engineering, Central South University, China
[2] Department of Computer Science and Engineering, Texas A&M University, USA

Abstract. The Maximun Agreement Forest problem (MAF) asks for a largest common subforest of a collection of phylogenetic trees. The MAF problem on two binary phylogenetic trees has been studied extensively in the literature. In this paper, we present the first group of fixed-parameter tractable algorithms for the MAF problem on multiple (i.e., two or more) binary phylogenetic trees. Our techniques work fine for the problem for both rooted trees and unrooted trees. The computational complexity of our algorithms is comparable with that of the known algorithms for two trees, and is independent of the number of phylogenetic trees for which a maximum agreement forest is constructed.

1 Introduction

Phylogenetic trees have been widely used in the study of evolutionary biology to represent the tree-like evolution of a collection of species. However, different methods often lead to different trees. In order to facilitate the comparison of different phylogenetic trees, several distance metrics have been proposed, such as Robinson-Foulds [11], NNI [10], TBR and SPR [9,13].

A graph theoretical model, the *maximum agreement forest* (MAF) of two phylogenetic trees, has been formulated for the TBR distance and the SPR distance [8] for phylogenetic trees. Define the *order* of a forest to be the number of connected components in the forest.[1] Allen and Steel [1] proved that the TBR distance between two unrooted binary phylogenetic trees is equal to the order of their MAF minus 1, and Bordewich and Semple [3] proved that the rSPR distance between two rooted binary phylogenetic trees is equal to the order of their rooted version of MAF minus 1. In terms of computational complexity, it is known that computing the order of an MAF is NP-hard for two unrooted binary phylogenetic trees [8], as well as for two rooted binary phylogenetic trees [3].

Thus, the order of an MAF measures the "difference" between the two phylogenetic trees constructed from the same collection of species, which can be

* This work is supported by the National Natural Science Foundation of China under Grants (61103033, 61173051, 70921001), and the Doctoral Discipline Foundation of Higher Education Institution of China under Grant (20090162110056).

[1] The definitions for the study of maximum agreement forests have been kind of confusing. If *size* denotes the number of edges in a forest, then for a forest, the size is equal to the number of vertices minus the order. In particular, when the number of vertices is fixed, a forest of a large size means a small order of the forest.

D.-Z. Du and G. Zhang (Eds.): COCOON 2013, LNCS 7936, pp. 567–578, 2013.

small in practice. This observation has motivated the study of parameterized algorithms for the MAF problem, where the problem is parameterized by the order k of an MAF. A parameterized problem is *fixed-parameter tractable* [6] if it is solvable in time $f(k)n^{O(1)}$. In particular, for small values of the parameter k, such an algorithm may solve the problem more effectively. Allen and Steel [1] showed that the MAF problem on unrooted binary phylogenetic trees is fixed-parameter tractable. Hallett and McCartin [7] developed a faster parameterized algorithm of running time $O(4^k k^5 + n^{O(1)})$ for the MAF problem on two unrooted binary phylogenetic trees. Whidden and Zeh [15] further improved the time complexity to $O(4^k k + n^3)$ or $O(4^k n)$. A further faster algorithm has been announced recently by Chen, Fan, and Sze [5], which runs in time $O(3^k n)$ and is currently the fastest algorithm for the MAF problem on two unrooted binary phylogenetic trees. For the MAF problem on two rooted binary phylogenetic trees, Bordewich *et al.* [2] developed a parameterized algorithm of running time $O(4^k k^4 + n^3)$. Whidden *et al.* [14] improved this bound and developed an algorithm of running time $O(2.42^k k + n^3)$. This is currently the fastest algorithm for the MAF problem on two rooted binary phylogenetic trees.

On the other hand, the computational complexity for the MAF problem on more than two phylogenetic trees has not been studied as extensively as that on two trees. Note that it makes perfect sense to investigate the MAF problem on more than two phylogenetic trees: we may construct the phylogenetic trees for the same collection of species using more than two methods. However, it seems much more difficult to construct an MAF for more than two trees than that for two trees. For example, while there have been several polynomial-time approximation algorithms of ratio 3 for the MAF problem on two rooted binary phylogenetic trees [12,14] (the same ratio even holds true for the MAF problem on two unrooted multifurcating trees [5]), the best polynomial-time approximation algorithm [4] for the MAF problem on more than two rooted binary phylogenetic trees has a ratio 8. Similarly, while there have been more than half-dozen fixed-parameter tractable algorithms for the MAF problem on two (rooted or unrooted) binary phylogenetic trees [1,2,5,7,14,15], to our best knowledge, it is still unknown whether the MAF problem on more than two (rooted or unrooted) binary phylogenetic trees is fixed-parameter tractable.

In the current paper, we will be focused on parameterized algorithms for the MAF problem on multiple (i.e., two or more) binary phylogenetic trees, for both the version of rooted trees and the version of unrooted trees. Our main contributions include an $O(3^k n)$-time parameterized algorithm for the MAF problem on multiple rooted binary phylogenetic trees, and an $O(4^k n)$-time parameterized algorithm for the MAF problem on multiple unrooted binary phylogenetic trees. Our algorithms show that these problems are fixed-parameter tractable.

Our algorithms are based on the following simple ideas that, however, require a careful and efficient implementation. Let $\mathcal{C} = \{T_1, T_2, \ldots, T_m\}$ be a collection of rooted or unrooted binary phylogenetic trees. Note that an MAF of order k for the trees in $\mathcal{C}$ must be an agreement forest for the first two trees T_1 and T_2, which although may not be necessarily maximum. Therefore, if we can essentially

examine *all* agreement forests of order bounded by k for the trees T_1 and T_2, then we can easily check if any of them is an MAF for all the trees in $\mathcal{C}$ (note that checking if a forest is a subgraph of a tree in $\mathcal{C}$ is easy). In order to implement this idea, however, we must overcome the following difficulties. First, we must ensure that no agreement forest in our concern is missing. This in fact requires new and non-trivial techniques: *all* MAF algorithms for two trees proposed in the literature are based on resolving conflicting structures in the two trees, and do not guarantee examining all agreement forests of order bounded by k. The conflicting structures help to identify edges in the trees whose removal leads to the construction of the MAF. Therefore, if the two trees T_1 and T_2 do not conflict much (in an extreme case, T_1 and T_2 are isomorphism), then an MAF for T_1 and T_2 may not help much for constructing an MAF for all the trees in $\mathcal{C}$. Secondly, with the assurance that essentially all concerned agreement forests for T_1 and T_2 are examined, we must make sure that our algorithms are sufficiently efficient. This goal has also been nicely achieved: compared with the algorithms *published* in the literature, our $O(3^k n)$-time algorithm for the MAF problem on multiple rooted binary phylogenetic trees is asymptotically faster than the best published algorithm for the MAF problem on two rooted binary phylogenetic trees, which runs in time $O(4^k n^{O(1)})$ [3], and our $O(4^k n)$-time algorithm for the MAF problem on multiple unrooted binary phylogenetic trees matches the computational complexity of the best published algorithm for the MAF problem on two unrooted binary phylogenetic trees [7]. Only very recent work on two rooted trees [14] and on two unrooted trees [5], still in the status of unpublished manuscripts, has slightly improved these bounds, which, however, do not seem to be extendable to the problems on more than two trees. On the other hand, our algorithms work fine for the MAF problems for an arbitrary number of trees.

2 Definitions and Problem Formulations

A tree is a *single-vertex tree* if it consists of a single vertex, which is the leaf of the tree. A tree is a *single-edge tree* if it consists of a single edge. A tree is *binary* if either it is a single-vertex tree or each of its vertices has degree either 1 or 3. The degree-1 vertices are *leaves* and the degree-3 vertices are *non-leaves* of the tree. There are two versions in our discussion, one is on unrooted trees and the other is on rooted trees. We first give the terminologies on the unrooted version, then remark on the differences for the rooted version. Let X be a fixed *label-set*.

Unrooted X-Trees and X-Forests

A binary tree is *unrooted* if no root is specified in the tree – in this case no ancestor-descendant relation is defined in the tree. For the label-set X, an unrooted *binary phylogenetic X-tree*, or simply an unrooted *X-tree*, is an unrooted binary tree whose leaves are labeled bijectively by the label-set X (all non-leaves are not labeled). An unrooted X-tree will also be called an (unrooted) *leaf-labeled tree* if the label-set X is irrelevant. A *subforest* of an unrooted X-tree T is a subgraph of T, and a *subtree* of T is a connected subgraph of T. An unrooted *X-forest* F is a subforest of an unrooted X-tree T that contains all

leaves of T such that each connected component of F contains at least one leaf in T. Thus, an unrooted X-forest F is a collection of leaf-labeled trees whose label-sets are disjoint such that the union of the label-sets is equal to X. Define the *order* of the X-forest F, denoted $\mathrm{Ord}(F)$, to be the number of connected components in F. For a subset X' of the label-set X, the *subtree induced by* X' in an unrooted X-tree T, denoted by $T[X']$, is the minimal subtree of T that contains all leaves with labels in X'.

A subtree T' of an unrooted X-tree may contain unlabeled vertices of degree less than 3. In this case we apply the *forced contraction* operation on T', which replaces each degree-2 vertex v and its incident edges with a single edge connecting the two neighbors of v, and removes each unlabeled vertex that has degree smaller than 2. Note that the forced contraction does not change the order of an X-forest. An X-forest F is *strongly reduced* if the forced contraction does not apply to F. It has been well-known that the forced contraction operation does not affect the construction of an MAF for X-trees (see, for example, [2,7]). Therefore, we will assume that the forced contraction is applied immediately whenever it is applicable. Thus, the X-forests in our discussion are always assumed to be strongly reduced. With this assumption, a unlabeled vertex in an unrooted X-trees is always of degree 3. If a leaf-labeled forest F' is isomorphic to a subforest of an X-forest F (up to the forced contraction), then we will simply say that F' is a subforest of F.

Rooted X-Trees and X-Forests

A binary tree is *rooted* if a particular leaf is designated as the root (so it is *both* a root and a leaf), which specifies a unique ancestor-descendant relation in the tree. A rooted *X-tree* is a rooted binary tree whose leaves are labeled bijectively by the label-set X. The root of an X-tree will always be labeled by a special symbol ρ in X. A subtree T' of a rooted X-tree T is a connected subgraph of T which contains at least one leaf in T. In order to preserve the ancestor-descendant relation in T, we should define the root of the subtree of T. If T' contains the leaf ρ, certainly, it is the root of the subtree; if T' does not contain the leaf ρ, the node in T' which is the least common ancestor of the leaves in T' is defined to be the root of T'. A subforest of a rooted X-tree T is defined to be a subgraph of T. A (rooted) *X-forest* F is a subforest of a rooted X-tree T that contains a collection of subtrees whose label-sets are disjoint such that the union of the label-sets is equal to X. Thus, one of the subtrees in a rooted X-forest F must have the vertex labeled ρ as its root.

We again assume that the forced contraction is applied immediately whenever it is applicable. However, if the root r of a subtree T' is of degree 2, then the operation will *not* be applied on r, in order to preserve the ancestor-descendant relation in T. Therefore, after the forced contraction, the root of a subtree T' of a rooted X-tree is either an unlabeled vertex of degree-2, or the vertex labeled ρ of degree-1 , or a labeled vertex of degree-0. All unlabeled vertices in T' that is not the root of T' have degree 3. We say that a leaf-labeled forest F' is a subforest of a rooted X-forest F if F' is isomorphic to a subforest of the X=forest F (up to the forced contraction).

Agreement Forests

The following terminologies are used for both rooted and unrooted versions.

An X-forest F is an *agreement forest* for a collection $\{F_1, F_2, \ldots, F_m\}$ of X-forests if F is a subforest of F_i, for all i. A *maximum agreement forest* (abbr. *MAF*) F^* for $\{F_1, F_2, \ldots, F_m\}$ is an agreement forest for $\{F_1, F_2, \ldots, F_m\}$ with a minimum $\mathrm{Ord}(F^*)$ over all agreement forests for $\{F_1, F_2, \ldots, F_m\}$.

The problems we are focused on are parameterized versions of the Maximum Agreement Forest Problem for an arbitrary number of X-trees, with a rooted version and an unrooted version, which are formally given as follows.

ROOTED MAXIMUM AGREEMENT FOREST (rooted-MAF)
Input: A set $\{F_1, \ldots, F_m\}$ of rooted X-forests, and a parameter k
Output: an agreement forest F^* for $\{F_1, \ldots, F_m\}$ with $\mathrm{Ord}(F^*) \leq k$, or report that no such an agreement forest exists

UNROOTED MAXIMUM AGREEMENT FOREST (unrooted-MAF)
Input: A set $\{F_1, \ldots, F_m\}$ of unrooted X-forests, and a parameter k
Output: an agreement forest F^* for $\{F_1, \ldots, F_m\}$ with $\mathrm{Ord}(F^*) \leq k$, or report that no such an agreement forest exists

When each of the X-forests F_1, ..., F_m is an X-tree, the above problems become the standard Maximum Agreement Forest Problems on multiple binary phylogenetic trees, for the rooted version and the unrooted version, respectively.

The following concept on two X-forests will be important in our discussion, which applies to both rooted version and the unrooted version.

Definition 1. Let F_1 and F_2 be two X-forests (either both rooted or both unrooted). An agreement forest F for F_1 and F_2 is a *maximal agreement forest* (*maximal-AF*) for F_1 and F_2 if there is no agreement forest F' for F_1 and F_2 such that F is a subforest of F' and $\mathrm{Ord}(F') < \mathrm{Ord}(F)$.

By definition, an MAF for two X-forests F_1 and F_2 is also a maximal-AF for F_1 and F_2. Note that *every* agreement forest for two X-forests F_1 and F_2 is a subforest of a maximal-AF F' for F_1 and F_2, but F' may not be unique.

3 Maximal-AF for Two X-Forests

Fix a label-set X. Because of the bijection between the leaves in an X-forest F and the elements in the label-set X, sometimes we will use, without confusion, a label in X to refer to the corresponding leaf in F, or vice versa.

Let F_1 and F_2 be two X-forests, either both are rooted or both are unrooted. In this section, we discuss how we enumerate *all* maximal-AF for F_1 and F_2. The discussion is divided into the case for the rooted version and the case for the unrooted version.

Rooted Maximal-AF

In this case, both F_1 and F_2 are rooted X-forests. We proceed by repeatedly removing edges in F_1 and F_2 until certain condition is met. Let F^* be a fixed maximal-AF for F_1 and F_2.

Two labels a and b (and their corresponding leaves) in a forest are *siblings* if they have the common parent. We start with the following simple lemma.

Lemma 1. *Let F_1 and F_2 be two strongly reduced rooted X-forests. If F_2 contains no sibling pairs, then F_1 and F_2 has a unique maximal-AF that can be constructed in linear time.*

Proof. Let T be a connected component of F_2, which is a rooted leaf-labeled tree. If F_2 contains no sibling pairs, then neither does T. Therefore, if T does not contain the root ρ, then T must be a single-vertex tree whose leaf is a labeled vertex. If T contains ρ, then T is either a single-vertex tree whose leaf is ρ or a single-edge tree whose root is ρ with a unique child that is labeled by a label τ. Thus, all connected components of the X-forest F_2 are single-vertex trees, except at most one that is a single-edge tree whose two leaves are labeled by the elements ρ and τ in X. Therefore, if the leaves ρ and τ are in the same connected component in the X-forest F_1, then the (unique) maximal-AF for F_1 and F_2 is the X-forest F_2 itself. On the other hand, if ρ and τ are in different connected components in F_1, then the maximal-AF (again unique) for F_1 and F_2 consists of only single-vertex trees, each is labeled by an element in X. □

By Lemma 1, therefore, in the following discussion, we will assume that the rooted X-forest F_2 has a sibling pair (a, b). By definition, a and b cannot be ρ. Let p_2, which is an unlabeled vertex, be the parent of a and b in F_2. If one of a and b is a single-vertex tree in the X-forest F_1, then we can remove the edge in F_2 that is incident to the label, and break up the sibling pair in F_2. Thus, in the following discussion, we assume that none of a and b is a single-vertex tree in F_1. Let p_1 and p'_1 be the parents of a and b in F_1, respectively. We consider all possible cases for the labels a and b in the X-forest F_1.

Case 1. The labels a and b are in different connected components in F_1.

In this case, a and b cannot be in the same connected component in the maximal-AF F^*. Therefore, one of the edges $[a, p_2]$ and $[b, p_2]$ in F_2 must be removed, which forces one of the labels a and b to be a single-vertex tree in the maximal-AF F^*. Therefore, in this case, we apply the following branching step:

Step 1. (branch-1) remove the edge $[a, p_1]$ in F_1 and the edge $[a, p_2]$ in F_2 to make a a single-vertex tree in both F_1 and F_2;
(branch-2) remove the edge $[b, p'_1]$ in F_1 and the edge $[b, p_2]$ in F_2 to make b a single-vertex tree in both F_1 and F_2.

One of these branches will keep F^* a maximal-AF for the new F_1 and F_2.

Case 2. The labels a and b are also siblings in F_1, i.e., $p_1 = p'_1$.

Since F^* is a maximal-AF, in this case, a and b must be also siblings in F^*. Therefore, the structure that consists of a and b and their parent remains unchanged when we construct F^* from F_1 and F_2 by removing edges in F_1 and F_2. Thus, this structure can be regarded as a single leaf labeled by a "combined" label $\underline{ab}$ in both F_1 and F_2. To implement this, we apply the following step:

Step 2. Remove a and b, and make their parent a new leaf labeled $\underline{ab}$, in both F_1 and F_2.

We call the operation in Step 2 "shrinking a and b into a new label $\underline{ab}$". This step not only changes the structure of F_1 and F_2, but also replaces the label-set

X with a new label-set $(X \setminus \{a, b\})) \cup \{\underline{ab}\}$. If we also apply this operation in the maximal-AF F^*, then the new F^* remains a maximal-AF for F_1 and F_2.

Case 3. The labels a and b are in the same connected component in F_1 but are not siblings.

Let $P = \{a, c_1, c_2, \ldots, c_r, b\}$ be the unique path in F_1 connecting a and b, in which c_h is the least common ancestor of a and b, $1 \leq h \leq r$. Since a and b are not siblings, $r \geq 2$. See Figure 1(a) for an illustration. There are three subcases.

SUBCASE 3.1. a is a single-vertex tree in F^*. Then removing the edge incident to a in both F_1 and F_2 keeps F^* a maximal-AF for F_1 and F_2.

SUBCASE 3.2. b is a single-vertex tree in F^*. Then removing the edge incident to b in both F_1 and F_2 keeps F^* a maximal-AF for F_1 and F_2.

SUBCASE 3.3. Neither of a and b is a single-vertex tree in F^*. Then the two edges that are incident to a and b in F_2 must be kept in F^*. Therefore, a and b are siblings in F^*. On the other hand, in order to make a and b siblings in the X-forest F_1, all edges that are not on the path P but are incident to a vertex c_j in P, where $j \neq h$, must be removed (note that this is because the subtrees in an X-forest must preserve the ancestor-descendant relation). Note that since $r \geq 2$, there is at least one such an edge. Therefore, in this subcase, if we remove all these edges, then F^* remains a maximal-AF for F_1 and F_2.

Summarizing the above analysis, in Case 3, we apply the following step:

Step 3. (branch-1) remove the edge incident to a in both F_1 and F_2;
(branch-2) remove the edge incident to b in both F_1 and F_2;
(branch-3) remove all edges in F_1 that are not on the path P connecting a and b but are incident to a vertex in P, except the one that is incident to the least common ancestor of a and b.

One of these branches must keep F^* a maximal-AF for the new F_1 and F_2.

Therefore, for two given rooted X-forests F_1 and F_2, if we iteratively apply the above process, branching accordingly based on the cases, then the process will end up with a pair (F_1, F_2) in which F_2 contains no sibling pairs. When this occurs, the process applies the following step:

Final Step. if F_2 has no sibling pairs, then construct the maximal-AF F^* for F_1 and F_2, and convert F^* into an agreement forest for the original F_1 and F_2.

When F_2 contains no sibling pairs, by Lemma 1, we can construct the (unique) maximal-AF F^* for F_1 and F_2 in linear time. The forest F^* may not be a subforest of the original F_1 and F_2 because Step 2 shrinks labels. For this, we should "expand" the shrunk labels, in a straightforward way. Note that this expanding process may be applied iteratively.

Summarizing the above discussion, we conclude with the following lemma.

Lemma 2. *Let F_1 and F_2 be two rooted X-forests. If we apply Steps 1-3 iteratively until F_2 contains no sibling pairs, then for every maximal-AF F^* for the original F_1 and F_2, at least one of the branches in the process produces the maximal-AF F^* in its Final Step.*

Proof. Fix a maximal-AF F^* for F_1 and F_2. By the above analysis, for each of the cases, at least one of the branches in the corresponding step keeps F^*

a maximal-AF for F_1 and F_2. Moreover, when F_2 contains no sibling pairs, the maximal-AF for F_1 and F_2 becomes unique. Combining these two facts, we conclude that at least one of the branches in the process ends up with an pair F_1 and F_2 whose maximal-AF, after the final step, is F^*. Since F^* is an arbitrary maximal-AF for F_1 and F_2, the lemma is proved. □

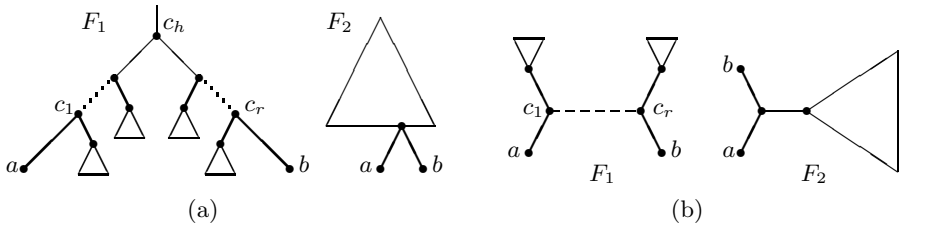

Fig. 1. The path connecting the labels a and b in F_1 when F_1 is (a) rooted; (b) unrooted

Unrooted Maximal-AF

The analysis for the unrooted version proceeds in a similar manner. However, since an unrooted tree enforces no ancestor-descendant relation in the tree, subtrees in the tree have no requirement of preserving such a relation. This fact induces certain subtle differences.

Let F_1 and F_2 be two unrooted X-forests, and let F^* be a fixed maximal-AF for F_1 and F_2. Recall that we assume F_1 and F_2 to be strongly reduced.

Two labels a and b in an unrooted X-forest F are *siblings* if either they are the two leaves of a single-edge tree in F, or they are adjacent to the same non-leaf vertex in F, which will be called the "parent" of a and b.

An unrooted X-forest with no sibling pairs has an even simpler structure: all its connected components are single-vertex trees. Thus, we again have:

Lemma 3. *Let F_1 and F_2 be two unrooted X-forests. If F_2 contains no sibling pairs, then the maximal-AF for F_1 and F_2 can be constructed in linear time.*

Thus, again we will assume that the unrooted X-forest F_2 has a sibling pair (a, b). Also we can assume that none of a and b is a single-vertex tree in F_1.

Case 1. The labels a and b are in different connected components in F_1.

In this case, again one of the labels a and b must be a single-vertex tree in the maximal-AF F^*. Therefore, we apply the following step:

Step 1. (branch-1) remove the edge incident to a in both F_1 and F_2 to make a a single-vertex tree in both F_1 and F_2;
(branch-2) remove the edge incident to b in both F_1 and F_2 to make b a single-vertex tree in both F_1 and F_2.

Case 2. The labels a and b are also siblings in F_1.

We have to be a bit more careful for this case since a sibling pair may come from a single-edge tree. There are three different cases: (1) a and b come from a single-edge tree in both F_1 and F_2; (2) a and b come from a single-edge tree in exact one of F_1 and F_2; and (3) a and b have a common parent in both F_1 and F_2. By a careful analysis and noticing that F^* is maximal, we can verify that in

all these subcases it is always safe to shrink a and b into a new label, which is implemented by the following step:

Step 2. Shrink the labels a and b in both F_1 and F_2: if a and b have a common parent, then remove the edges incident to a and b and make their parent a new leaf labeled $\underline{ab}$; if a and b come from a single-edge tree, then combine them into a single vertex labeled $\underline{ab}$.

After this process, the maximal-AF F^* for F_1 and F_2, in which the labels a and b are also shrunk, remains a maximal-AF forest for the new F_1 and F_2.

Case 3. a and b are in the same connected component in F_1 but are not siblings.

Let $P = \{a, c_1, c_2, \ldots, c_r, b\}$ be the unique path in F_1 that connects a and b, where $r \geq 2$. See Figure 1(b) for an illustration. The cases in which either a or b is a single-vertex tree in F^* again cause removing the edge incident to a or b in F_1. However, when a and b are siblings in F^*, then in F_1, at most one of the edges that are not on the path P but are incident to a vertex in P can be kept. However, since the subtree in an unrooted forest does not need to preserve any ancestor-descendant relation, we cannot decide which of these edges should be kept. On the other hand, since $r \geq 2$, we know at least one of the two edges, which are not on the path P but are incident to c_1 and c_r, respectively, must be removed. Therefore, we can branch by removing either the one incident to c_1 or the one incident to c_r. In summary, in Case 3, we apply the following step:

Step 3. (branch-1) remove the edge incident to a in both F_1 and F_2;
(branch-2) remove the edge incident to b in both F_1 and F_2;
(branch-3) remove the edge incident to c_1 but not on the path P in F_1;
(branch-4) remove the edge incident to c_r but not on the path P in F_1.

One of these branches must keep F^* a maximal-AF for the new F_1 and F_2.

Again if the unrooted X-forest F_2 contains no sibling pairs, then we apply Lemma 3 to construct the maximal-AF for F_1 and F_2 by the following step:

Final Step. If F_2 contains no sibling pairs, then construct the maximal-AF F^* for F_1 and F_2, and convert F^* into an agreement forest for the original F_1, F_2.

The above analysis finally gives the following conclusion, whose proof is exactly the same as that of Lemma 2 for the rooted version.

Lemma 4. *Let F_1 and F_2 be two unrooted X-forests. If we apply Steps 1-3 iteratively until F_2 contains no sibling pairs, then for every maximal-AF F^* for the original F_1 and F_2, at least one of the branches in the process produces the maximal-AF F^* in its Final Step.*

4 The Parameterized Algorithms

Now we are ready for presenting the parameterized algorithms for the MAF problem, for both the rooted version as well as the unrooted version. Let F_1, F_2, ..., F_m be m X-forests, either all are rooted or all are unrooted. We first give a few lemmas, which hold true for both rooted and unrooted versions. Assume $m \geq 3$.

The first lemma follows directly from the definition.

Lemma 5. *Let F' be an agreement forest for F_1 and F_2. Then every agreement forest for $\{F', F_3, \ldots, F_m\}$ is an agreement forest for $\{F_1, F_2, \ldots, F_m\}$. If F' contains an MAF for $\{F_1, F_2, \ldots, F_m\}$, then an MAF for $\{F', F_3, \ldots, F_m\}$ is also an MAF for $\{F_1, F_2, \ldots, F_m\}$.*

Lemma 6. *For every MAF F for $\{F_1, F_2, \ldots, F_m\}$, there is a maximal-AF F^* for F_1 and F_2 such that F is also an MAF for $\{F^*, F_3, \ldots, F_m\}$.*

Proof. Let F_0 be an MAF for $\{F_1, F_2, \ldots, F_m\}$. Then F_0 is an agreement forest for F_1 and F_2. Let F^* be a maximal-AF for F_1 and F_2 that has F_0 as a subforest. Then F_0 is an agreement forest for $\{F^*, F_3, \ldots, F_m\}$. Therefore, the order of an MAF for $\{F^*, F_3, \ldots, F_m\}$ is at most $\mathrm{Ord}(F_0)$. On the other hand, since F^* is a subforest of both F_1 and F_2, every agreement forest for $\{F^*, F_3, \ldots, F_m\}$ is also an agreement forest for $\{F_1, F_2, \ldots, F_m\}$. Therefore, the order of an MAF for $\{F^*, F_3, \ldots, F_m\}$ is at least $\mathrm{Ord}(F_0)$, thus must be equal to $\mathrm{Ord}(F_0)$. Since F_0 is an agreement forest for $\{F^*, F_3, \ldots, F_m\}$, F_0 must be an MAF for $\{F^*, F_3, \ldots, F_m\}$. □

Now consider an instance $(F_1, F_2, \ldots, F_m; k)$ of MAF, either rooted or unrooted. For a subforest F' of a forest F, we always have $\mathrm{Ord}(F) \leq \mathrm{Ord}(F')$. Thus, no maximal-AF F for F_1 and F_2 with $\mathrm{Ord}(F) > k$ can contain an MAF F' for $(F_1, F_2, \ldots, F_m)$ with $\mathrm{Ord}(F') \leq k$, so we only need to examine all maximal-AFs whose order is bounded by k. An outline of our algorithm works as follows:

Main-Algorithm
1. construct a collection $\mathcal{C}$ of agreement forests for F_1 and F_2 that contains all maximal-AF F^* for F_1 and F_2 with $\mathrm{Ord}(F^*) \leq k$;
2. **for** each agreement forest F for F_1 and F_2 constructed in step 1 **do** recursively work on the instance $(F, F_3, \ldots, F_m; k)$.

Theorem 1. *The Main-Algorithm correctly returns an agreement forest F for $\{F_1, F_2, \ldots, F_m\}$ with Ord$(F) \leq k$ if such an agreement forest exists.*

Proof. For an F' in the collection $\mathcal{C}$, by Lemma 5, a solution to $(F', F_3, \ldots, F_m; k)$ returned by step 2 is also a solution to $(F_1, F_2, \ldots, F_m; k)$. On the other hand, if $(F_1, F_2, \ldots, F_m; k)$ has a solution, then an MAF F_0 for $\{F_1, F_2, \ldots, F_m\}$ satisfies $\mathrm{Ord}(F_0) \leq k$. For the maximal-AF F^* for F_1 and F_2 that contains F_0, by Lemma 6, F_0 is also a solution to $(F^*, F_3, \ldots, F_m; k)$, which is an instance examined in step 2. On this instance, Step 2 will return a solution that, by Lemma 5, is also a solution to $(F_1, F_2, \ldots, F_m; k)$. □

In the following, we present the details for the Main-Algorithm for the rooted version. By Theorem 1,our must carefully check that all maximal-AF's F for F_1 and F_2 with $\mathrm{Ord}(F) \leq k$ be constructed in the collection $\mathcal{C}$. Also, we should develop algorithms to achieve the desired complexity bounds.

A Parameterized Algorithm for Rooted-MAF

The parameterized algorithm for rooted-MAF is a combination of the analysis given in Section 3 and the Main-Algorithm, which is given in Figure 2.

Algorithm. Rt-MAF$(F_1, F_2, \ldots, F_m; k)$
Input: a collection $\{F_1, F_2, \ldots, F_m\}$ of rooted X-forests, $m \geq 1$, and a parameter k
Output: an agreement forest F^* for $\{F_1, F_2, \ldots, F_m\}$ with $\text{Ord}(F^*) \leq k$ if F^* exists

1. **if** $(m = 1)$ **then if** $(\text{Ord}(F_1) \leq k)$ **then** return F_1 **else** return('no');
2. **if** $(\text{Ord}(F_1) > k)$ **then** return('no');
3. **if** a label a is a single-vertex tree in exactly one of F_1 and F_2
   then make a a single-vertex tree in both F_1 and F_2;
4. **if** F_2 has no sibling pairs
   then let F' be the maximal-AF for F_1 and F_2; return Rt-MAF$(F', F_3, \ldots, F_m; k)$;
5. let (a, b) be a sibling pair in F_2;
6. **if** a and b are in different connected components in F_1
   then branch:
   1. make a a single-vertex tree in both F_1 and F_2; return Rt-MAF$(F_1, F_2, \ldots, F_m; k)$;
   2. make b a single-vertex tree in both F_1 and F_2; return Rt-MAF$(F_1, F_2, \ldots, F_m; k)$;
7. **if** a and b are also siblings in F_1
   then shrink a, b into a new leaf $\underline{ab}$ in F_1 and F_2; return Rt-MAF$(F_1, F_2, \ldots, F_m; k)$;
8. let $P = \{a, c_1, \ldots, c_r, b\}$ be the unique path in F_1 connecting a and b, $r \geq 2$;
   then branch:
   1. make a a single-vertex tree in both F_1 and F_2; return Rt-MAF$(F_1, F_2, \ldots, F_m; k)$;
   2. make b a single-vertex tree in both F_1 and F_2; return Rt-MAF$(F_1, F_2, \ldots, F_m; k)$;
   3. remove all edges in F_1 not on P but incident to a vertex in P, except the one incident to the least common ancestor of a, b; return Rt-MAF$(F_1, F_2, \ldots, F_m; k)$.

Fig. 2. Algorithm for the Rooted-MAF problem

The algorithm is a branch-and-search process. Its execution can be depicted by a search tree $\mathcal{T}$ whose leaves correspond to conclusions or solutions generated by the algorithm based on different branches. Each internal node of the search tree $\mathcal{T}$ corresponds to a branch in the search process at Steps 6 or 8 based on an instance of the problem. We call a path from the root to a leaf in the search tree $\mathcal{T}$ a *computational path* in the process. The algorithm returns an agreement forest for the original input if and only if there is a computational path that outputs the forest.

The correctness and complexity of the algorithm can be verified based on the corresponding search tree $\mathcal{T}$. Due to the space limit, here we just give the concluding theorem, the entire discussion for this case will be given in a complete version.

Theorem 2. *The rooted-*MAF *problem can be solved in time* $O(3^k n)$.

A Parameterized Algorithm for Unrooted-MAF

The parameterized algorithm for unrooted-MAF proceeds in a similar way, based on the corresponding analysis given in Section 3. Due to the space limit, we only present its main result below, the specific algorithm for unrooted-MAF and the entire discussion for this case will be give in the complete version.

Theorem 3. *The unrooted-*MAF *problem can be solved in time* $O(4^k n)$.

5 Conclusion

In this paper, we presented two parameterized algorithms for the Maximum Agreement Forest problem on multiple binary phylogenetic trees: one for rooted trees that runs in time $O(3^k n)$, and the other for unrooted trees that runs in time $O(4^k n)$. To our best knowledge, these are the first group of fixed-parameter tractable algorithms for the Maximum Agreement Forest problem on multiple phylogenetic trees. Further improvements on the algorithm complexity are certainly desired to make the algorithms more practical in their applications. On the other hand, such an improvement seems to require new observations and new algorithmic techniques: the complexity of our algorithms for multiple phylogenetic trees is not much worse than that of the known algorithms for two phylogenetic trees – some of these algorithms were just developed very recently.

References

1. Allen, B., Steel, M.: Subtree transfer operations and their induced metrics on evolutionary trees. Annals of Combinatorics 5(1), 1–15 (2001)
2. Bordewich, M., McCartin, C., Semple, C.: A 3-approximation algorithm for the subtree distance between phylogenies. J. Discrete Algorithms 6(3), 458–471 (2008)
3. Bordewich, M., Semple, C.: On the computational complexity of the rooted subtree prune and regraft distance. Annals of Combinatorics 8(4), 409–423 (2005)
4. Chataigner, F.: Approximating the maximum agreement forest on k trees. Information Processing Letters 93, 239–244 (2005)
5. Chen, J., Fan, J.-H., Sze, S.-H.: Improved algoithms for the maximum agreement forest problem on general trees. In: WG 2013 (submitted, 2013)
6. Downey, R., Fellows, M.: Parameterized Complexity. Springer, New York (1999)
7. Hallett, M., McCartin, C.: A faster FPT algorithm for the maximum agreement forest problem. Theory of Computing Systems 41(3), 539–550 (2007)
8. Hein, J., Jiang, T., Wang, L., Zhang, K.: On the complexity of comparing evolutionary trees. Discrete Applied Mathematics 71, 153–169 (1996)
9. Hodson, F., Kendall, D., Tauta, P. (eds.): The recovery of trees from measures of dissimilarity. Mathematics in the Archaeological and Historical Sciences, pp. 387–395. Edinburgh University Press, Edinburgh (1971)
10. Li, M., Tromp, J., Zhang, L.: On the nearest neighbour interchange distance between evolutionary trees. Journal on Theoretical Biology 182(4), 463–467 (1996)
11. Robinson, D., Foulds, L.: Comparison of phylogenetic trees. Mathematical Biosciences 53(1-2), 131–147 (1981)
12. Rodrigues, E., Sagot, M., Wakabayashi, Y.: The maximum agreement forest problem: approximation algorithms and computational experiments. Theoretical Computer Science 374(1-3), 91–110 (2007)
13. Swofford, D., Olsen, G., Waddell, P., Hillis, D.: Phylogenetic inference. In: Molecular Systematics, 2nd edn., pp. 407–513. Sinauer Associates (1996)
14. Whidden, C., Beiko, R., Zeh, N.: Fixed-parameter and approximation algorithms for maximum agreement forests. CoRR. abs/1108.2664 (2011)
15. Whidden, C., Zeh, N.: A unifying view on approximation and FPT of agreement forests. In: Salzberg, S.L., Warnow, T. (eds.) WABI 2009. LNCS, vol. 5724, pp. 390–402. Springer, Heidelberg (2009)