COMMON STRUCTURE AMONG MAXIMUM AGREEMENT FORESTS OF PHYLOGENETIC TREES

by

Jordan Dempsey

Submitted in partial fulfillment of the requirements
for the degree of Master of Computer Science

at

Dalhousie University
Halifax, Nova Scotia
August 2021

To my great aunt "Kee."

For all the love, support, and encouragement you have provided, and for inspiring me to be a better person.

Until we meet again.

Table of Contents

List of Figures

Abstract

Reconciling sets of phylogenetic trees has become a problem of particular interest in both theoretical computer science and bioinformatics. The use of the maximum agreement forest as a tool by which to attain insight into the discrepancies in evolutionary paths between trees that model the same set of taxa has been well researched. However, to date there has not been any method by which to sample from the potentially large space of possible MAFs, and thus crucial information may be missed. To this end we introduce the notion of core maximum agreement forest, which provides us with the set of components preserved across the MAFs of a set of phylogenetic trees. Through the use of the established techniques of cluster partitioning and branching rules, we prove that this problem is fixed parameter tractable and present an efficient algorithm for computing this new structure in $\mathcal{O}(2.27^k \cdot n)$ time.

List of Abbreviations and Symbols Used

Abbreviations

AF Agreement Forest

FPT Fixed Parameter Tractable

LGT Lateral Gene Transfer

MAF Maximum Agreement Forest

SPR Subtree Prune and Regraft

Symbols

$(\mathcal{F}_1, \mathcal{F}_2)$ A pair of forests obtained by cuttings edges in (T_1, T_2)

(a, c) A sibling pair of a tree, called a cherry.

(T_1, T_2) A pair of phylogenetic trees.

$(T_1^{\downarrow}(Y), T_2^{\downarrow}(Y))$ The bottom cluster of (T_1, T_2) with respect to $Y \subseteq X$.

$(T_1^{\uparrow}(X \setminus Y), T_2^{\uparrow}(X \setminus Y))$ The top cluster of (T_1, T_2) with respect to $Y \subseteq X$.

$\mathbb{R}$ The set of real numbers.

$\mathcal{E}$ A cut edge set for a forest.

$\mathcal{F}//(x, y)$ The forest obtained by cutting the edge (x, y) in $\mathcal{F}$.

$\mathcal{F}$ A collection of trees, called a forest.

$\mathcal{L}(T)$ The leaf set of T.

ρ The outgroup of a phylogenetic tree.

$ab \mid c$ A triplet a, b, c of a tree.

$d_{rSPR}(T_1, T_2)$ The rooted SPR distance between T_1 and T_2.

$E(T)$ The edge set of T.

e_x The parent edge of node x.

k The number of components in an MAF of a pair of trees.

n The number of leaves in a tree.

$T - \mathcal{E}$ Remove the edges in $\mathcal{E}$ from T.

T A tree.

$T(U)$ The minimal subtree of T that connects all of $U \subseteq \mathcal{L}(T)$

$T\,|_U$ The tree obtained by applying forced contractions to all nodes in $T(U)$ that have exactly one child and exactly one parent.

$V(T)$ The vertex set of T.

X-tree A tree with label set X.

Acknowledgements

I would like to thank first and foremost my supervisor, Norbert Zeh, without whom this thesis would not exist, and who has and continues to serve as both a mentor and role model. Norbert's guidance, attention to detail, and feedback not only made this thesis possible, but have provided me with invaluable skills for both academia and beyond. The laughs shared during our off-topic meeting discussions also helped make getting through this challenge possible.

I would like to thank Rob Beiko and Dirk Arnold for the knowledge and insight provided in the courses which I was fortunate enough to have had them as instructors for during my masters program. I would further like to thank Meng He and Malcolm Heywood for providing me the opportunity to work with and learn from them as a teaching assistant in their respective courses.

I would also like to thank Rob as well as Chris Whidden for being readers for this thesis. Your comments and suggestions were crucial to not only completing this thesis, but were also helpful in providing skills by which to improve my writing in the future. Thank-you as well to Travis Gagie for serving as the moderator for my defense.

Thank-you to John Irving, for providing me with invaluable skills with respect to writing mathematical documents as well as inspiring my interest in combinatorics. To Bert Hartnell and Paul Muir, for all of your insight and guidance over the course of my time as an undergrad, and to Rachael Collins, for providing a great deal of advice and encouragement with respect to pursuing academics at the graduate level.

Finally, I would like to thank my parents, family, and friends who have offered, and continue to offer so much support over the course of my time as a "professional student."

Chapter 1

Introduction

Reconciling phylogenetic trees which model the evolutionary paths of a set of taxa has become an increasingly important area of study for bioinformaticians. *Lateral gene transfer (LGT)* is a non-tree like evolutionary process by which species become composites of genes derived from different ancestors. This process (and similar ones, collectively known as reticulation events) allow species to more easily acquire useful traits, such as antibiotic resistance [8]. As well it is an area that offers a host of interesting graph theoretical problems. The use of maximum agreement forests (MAFs) to compute the rooted subtree prune and regraft (rSPR) distance and better understand LGT has become a key tool in this respect [4]. An issue with computing MAFs in general is that there are potentially an exponentially large number to choose from, and we may not end up with one that is particularly insightful. To date there has not been work done into finding an efficient method by which to sample from the set of possible maximum agreement forests.

Motivated by the aforementioned issue in sampling from the set of all MAFs, we present a new concept, that of the core maximum agreement forest, wherein we preserve components that are present in a particular percentage of possible MAFs. To this end, we focus on the strict case of the core MAF, wherein we aim to obtain an AF which is comprised of components preserved across all possible MAFs for our input trees. We further narrow our focus to the case of rooted, binary phylogenetic trees, as a starting point for research into the realm of core MAFs, and the more general problem of sampling from the space of possible MAFs.

We discuss first and foremost of the technique of cluster partitioning. This has been shown to not only be safe in the case of MAFs, but indeed that it is an essential part of efficient algorithms to compute them, both in theory and in practice [5]. Hence we demonstrate that it is safe to make use of this technique to compute the core MAF. We show that this then implies that we can restrict the exponential part of our running time to be relative to the size of the largest core MAF among the set of core MAFs for each cluster, rather than the size of the core MAF of the input trees.

Finally, we present three branching algorithms for computing the core MAF of

two phylogenetic trees. The first algorithm presents a naive approach wherein we effectively enumerate all MAFs in order to determine the core MAF. While not a particularly impressive algorithm, it is sufficient to demonstrate that the exponential part of our running time can be restricted to the size of an MAF, which, in general, can be much smaller than the size of the core MAF. The second algorithm introduces improved branching rules to better address the worst case scenario that arises in the naive approach. Finally, we present a final branching rule, the depth rule used by our branching process, and the notion of edge protection, to obtain our most efficient algorithm. Each successive algorithm improves upon the previous to obtain an end result running time bound of $\mathcal{O}(2.27^k \cdot n)$, where k is the number of components in an MAF of our input trees, and n is the size of our input trees.

Chapter 2

Preliminaries and Related Work

2.1 Phylogenetic Trees and Agreement Forests

We begin by providing the basic definition of a phylogenetic tree, particularly as it is from a graph-theoretic perspective. To this end we first present the definition of X-trees in general, of which phylogenetic trees are a special case of.

Definition 2.1 (X-tree)**.** An *X-tree* is a pair $\mathcal{T} = (T, \phi)$ where $T = (V, E)$ is a tree and $\phi : X \to V$ is a mapping with the properties that $|X| < \infty$ and for any $v \in V$ such that $deg(v) \leq 2$ we have that $v \in \phi(X)$ (ϕ need not be injective nor surjective). Furthermore, we say X-trees $\mathcal{T}_1 = (T_1, \phi_1)$ and $\mathcal{T}_2 = (T_2, \phi_2)$ are isomorphic if there is a graph isomorphism Ψ between T_1 and T_2 such that $\phi_2 = \Psi \circ \phi_1$ [7]. A *rooted X-tree* is a pair $\mathcal{T} = (T, \phi)$, where $T = (V, E)$ has one vertex distinguished as the root, and $\phi : X \to V$

Definition 2.2 (Phylogenetic Tree)**.** A *phylogenetic tree* is an X-tree $\mathcal{T} = (T, \phi)$ such that ϕ is a bijective map from the label set X to the leaf set of T. A *rooted phylogenetic tree* is a rooted X-tree where ϕ is a bijective map from the label set X to the leaf set of T with the condition that the root has a child vertex which is a leaf labelled ρ. If we also have that the root has degree 2, and every other interior node is of degree 3, then we call $\mathcal{T}$ a *rooted binary phylogenetic tree* [7]. In the context of these trees our label set is a set of species and, in the rooted case, ρ represents the outgroup.

For the purpose of the current study, we focus on rooted binary phylogenetic trees, which from here on we refer to as phylogenetic trees for simplicity. We also note that, in previous work, ρ was used to denote the root. For the purposes of this research, we have altered this so that ρ is a leaf node that represents the outgroup of our tree and is a child of the root. This also allows us to maintain that only leaf vertices are labelled. In the unrooted case, it is possible to assign a rooting of the tree, however doing this correctly is generally poorly understood and often an arduous task.

Definition 2.3 (Ancestry and Depth)**.** We say that a node v is a *descendant* of a node u (and consequently that u is an *ancestor* of v) if u is on the path from v to the root of our tree. The *depth* of node is equal to its number of ancestors.

Definition 2.4 (Forced Contraction)**.** A *forced contraction* is an operation on a tree in which we delete a vertex v having a single parent u and child w and replace the two edges incident to v by a single edge (u, w) [1].

We write $\mathcal{L}(T)$ to refer to the leaf set of a tree T. Given a set $U \subseteq \mathcal{L}(T)$ we write $T(U)$ to denote the minimal subtree of T that connects all of U. We write $T|_U$ to denote the tree obtained from $T(U)$ by applying forced contractions to remove all vertices which have exactly one parent and one child.

For much of what follows, we let T_1, T_2 be phylogenetic trees such that $\mathcal{L}(T_1) = \mathcal{L}(T_2) = X$.

Definition 2.5 (Edge Cuts)**.** We denote by $\mathcal{F}//(x, y)$ the operation of *cutting* the edge $(x, y) \in V(\mathcal{F})$, whereby we remove (x, y) from $\mathcal{F}$ and apply forced contractions to any degree two vertices that arise afterwards. If $\mathcal{E}$ is a set of edges, then we denote by $T - \mathcal{E}$ the result cutting each edge from $\mathcal{E}$ in order in T.

Let x be a non-root node in a tree T. We denote by e_x the parent edge of x in T.

Definition 2.6 (Cut Edge Set)**.** Let $\mathcal{F}$ be a forest obtained from T_1. We say $\mathcal{E}$ is the *cut edge set* of $\mathcal{F}$ if $T_1 - \mathcal{E} = \mathcal{F}$. A *minimal cut edge set* of $\mathcal{F}$ is a cut edge set of minimal cardinality.

Definition 2.7 (Rooted Subtree Prune and Regraft (rSPR))**.** Let T be an X-tree. A *subtree prune and regraft* operation on T cuts an edge (x, y), where y is the parent of x. We then have two subtrees of T, T_x and T_y. It then introduce a new node y' in T_y that subdivides an edge of T_y, and adds the edge (x, y') Finally, y is supressed. This process is illustrated in Figure 2.1.

This provides us with the distance measure known as the *rooted SPR distance*. By $d_{rSPR}(T_1, T_2)$ we denote the minimum number of SPR operations required to transform T_1 into T_2 [4, 8].

Definition 2.8 (Agreement Forest)**.** An *agreement forest (AF)* for T_1, T_2 is a collection of binary trees $\mathcal{F} = \{t_1, t_2, ..., t_k\}$ such that if $\mathcal{L}_j := \mathcal{L}(t_j)$ for $j \in \{1, ..., k\}$ then

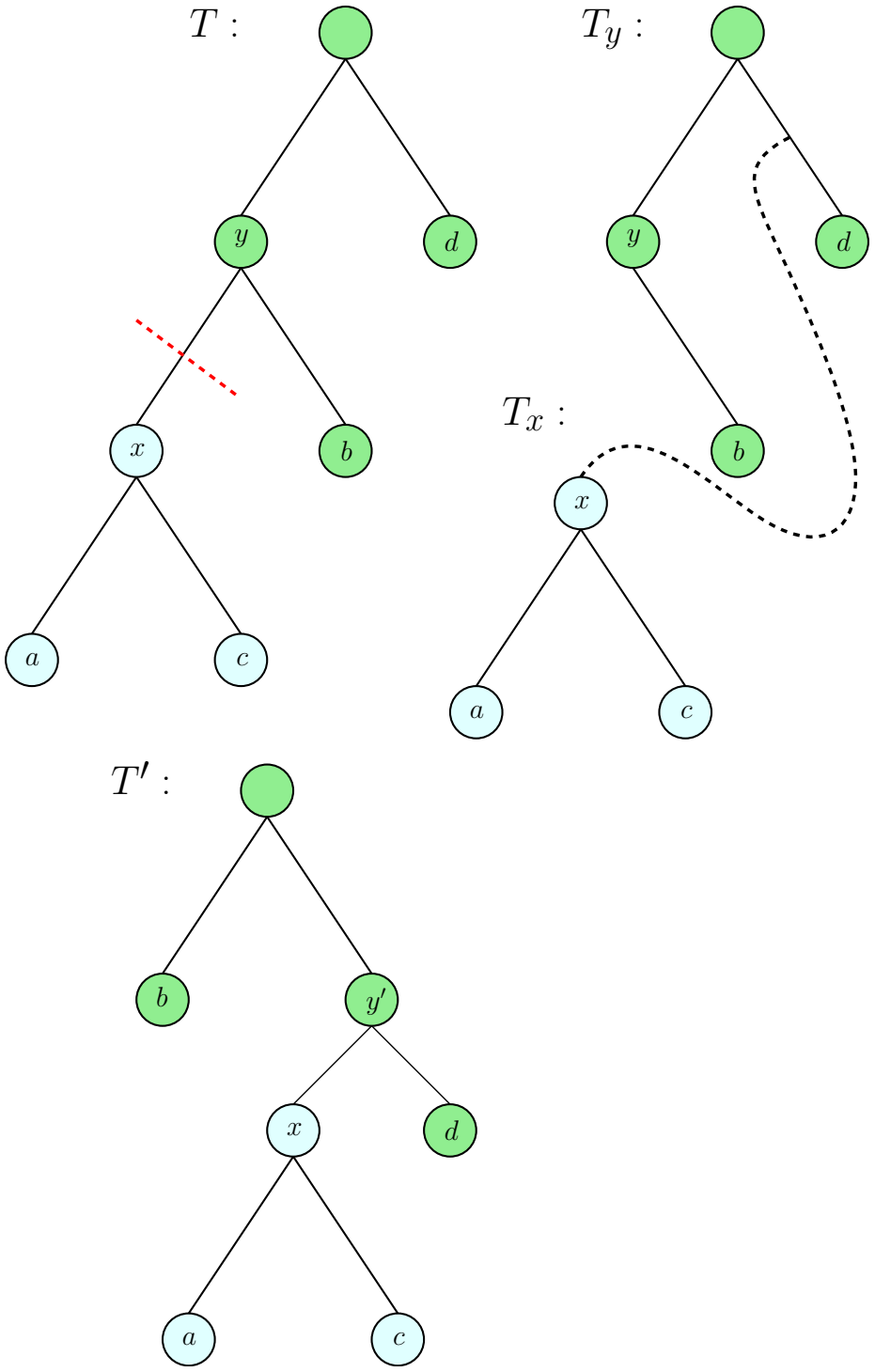


Figure 2.1: An SPR operation on a phylogenetic tree, T, and the resulting tree T'.

1. $\mathcal{L}_1, ..., \mathcal{L}_k$ partitions X,
2. $t_j = T_1|_{\mathcal{L}_j} = T_2|_{\mathcal{L}_j}$ for $j \in \{1, ..., k\}$, and
3. for both $i = 1$ and $i = 2$ we have that $\{T_i(\mathcal{L}_j) : j = 1, ..., k\}$ are edge-disjoint subtrees of T_i [1].

For the third requirement, we note that, in the case of binary trees, this also implies vertex-disjointness. A *maximum agreement forest (MAF)* is an AF such that $|\mathcal{F}|$ is minimum. An example of two rooted binary phylogenetic trees, T_1 and T_2, and one potential MAF for (T_1, T_2) is shown in Figure 2.2. The number of components in an MAF of (T_1, T_2) is equal to $d_{rSPR}(T_1, T_2) - 1$ [2].

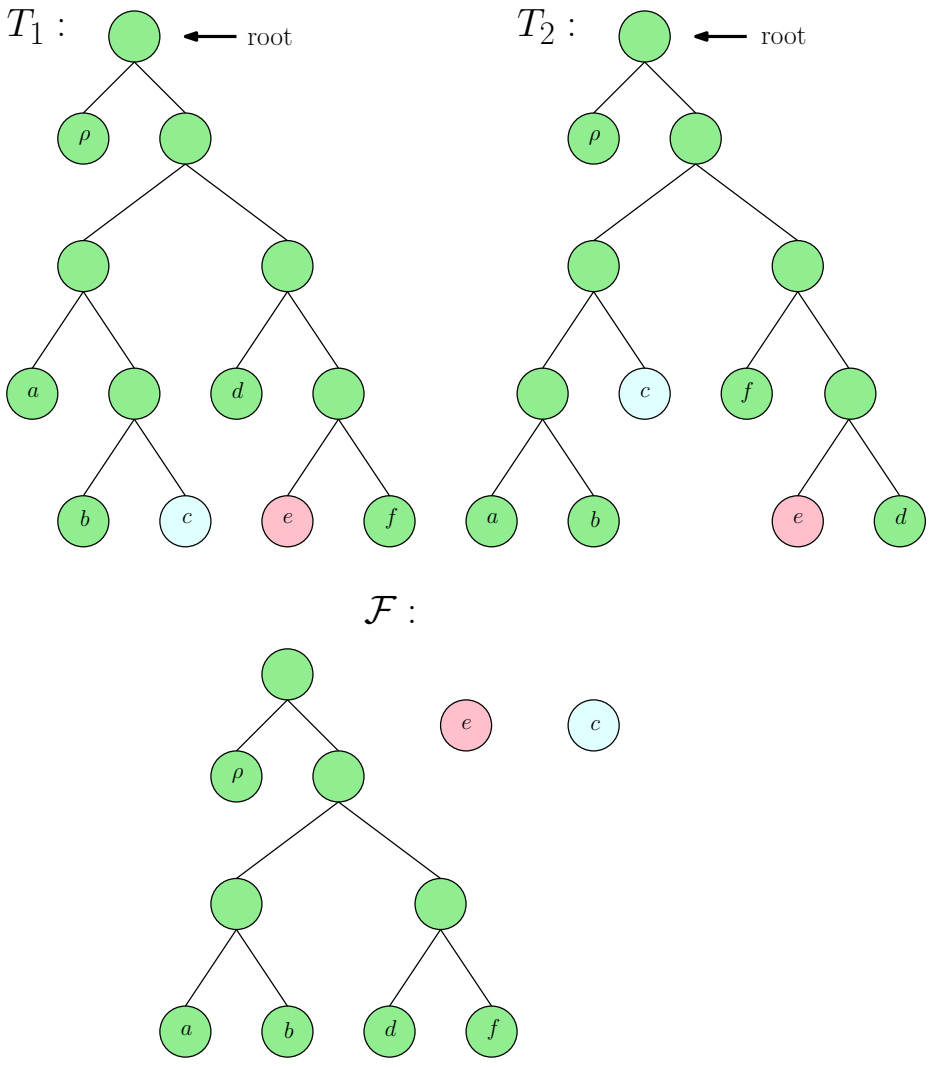


Figure 2.2: A pair of phylogenetic trees, (T_1, T_2), and an MAF, $\mathcal{F}$, for these trees. $\mathcal{F}$ can be obtained by cutting the parent edges of c and e in both T_1 and T_2.

Definition 2.9 (Cherries and Trivial Cherries)**.** A *cherry* of a tree T is a pair (a, c) such that a and c are leaves having the same parent in T. We say (a, c) is a *trivial cherry* of a pair of trees (T_1, T_2), if this cherry is present in both. Figure 2.3 gives examples of cherries and trivial cherries.

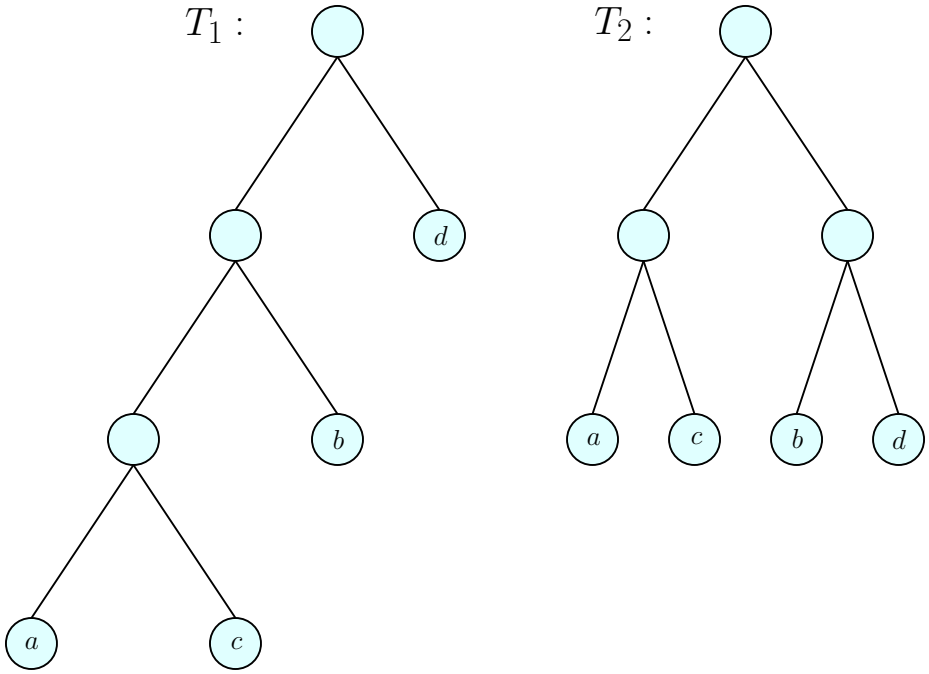


Figure 2.3: (a, c) is a trivial cherry, as it is present in both T_1 and T_2; (b, d), however, is only a cherry of T_2.

Definition 2.10 (Triplets)**.** A *triplet* $ab|c$ in a forest $\mathcal{F}$ is defined by a set of three leaves $\{a, b, c\}$ that are in the same connected component of $\mathcal{F}$ and such that the path from a to b is vertex disjoint from the path from c to the root of the component. A triplet of a forest $\mathcal{F}_1$ is *compatible* with a forest $\mathcal{F}_2$ if it is also a triplet of $\mathcal{F}_2$, otherwise this triplet is *incompatible* with $\mathcal{F}_2$ [8].

For $\mathcal{F}$ to be an MAF of (T_1, T_2), every triplet of $\mathcal{F}$ must be compatible with both T_1 and T_2.

Finally, we formalize the concept of the core MAF, the problem central to this thesis. As stated, the goal is that this (or some variant thereof) may be used as a tool by which we can infer common structure amongst the complete set of MAFs for two phylogenetic trees.

Definition 2.11 (Core Maximum Agreement Forest)**.** The *core maximum agreement forest (core MAF)* for (T_1, T_2) is an AF $\mathcal{S}$ of (T_1, T_2) such that, for every component

$t \in \mathcal{S}$ and every MAF $\mathcal{F}$ of (T_1, T_2), there exists some $t' \in \mathcal{F}$ such that $\mathcal{L}(t) \subseteq \mathcal{L}(t')$. Furthermore, $|\mathcal{S}|$ is minimal among all AFs which satisfy this property. An example of the core MAF for a pair of phylogenetic trees is shown in Figure 2.4.

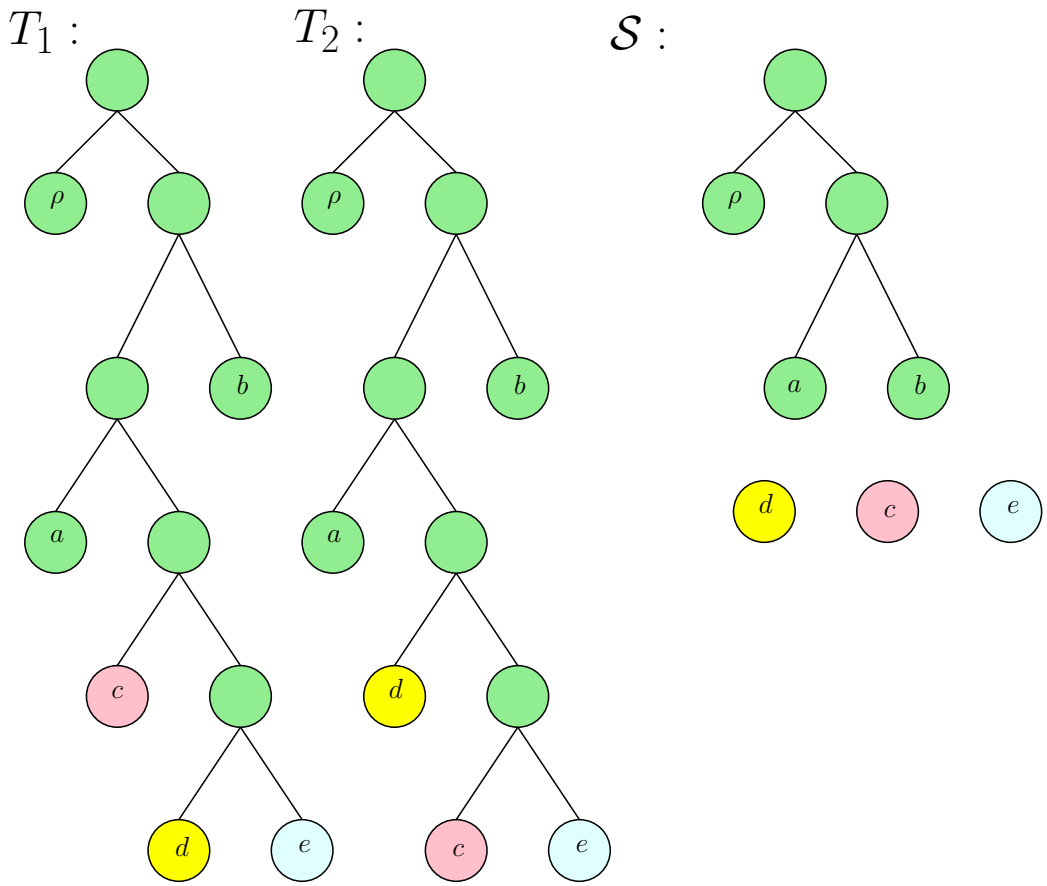


Figure 2.4: Phylogenetic trees T_1 and T_2 and their core MAF $\mathcal{S}$.

2.2 Parameterized Complexity

It is well known that it is NP-hard to compute the rooted subtree prune and regraft (rSPR) distance of two phylogenetic trees T_1 and T_2 (denoted $d_{rSPR}(T_1, T_2)$) [2]. Furthermore, it is known that $d_{rSPR}(T_1, T_2) = |\mathcal{F}| - 1$, where $\mathcal{F}$ is an MAF for T_1, T_2 [10]. Hence computing an MAF for two phylogenetic trees is an NP-hard problem. The decision variant of the AF problem is NP-complete. That is, given as input (T_1, T_2, k), we cannot decide in polynomial time whether there is an AF for T_1, T_2 of size at most k, unless $P = NP$.

Definition 2.12 (Fixed Parameter Tractability)**.** We say that a decision problem $\mathcal{P}$ is *fixed parameter tractable (FPT)* if, given some instance (x, k) of $\mathcal{P}$, we can decide in $\mathcal{O}(f(k) \cdot |x|^{\mathcal{O}(1)})$ time whether (x, k) is a yes-instance, where k is some parameter and f is a computable function depending only on k.

Example 2.13 (Vertex Cover)**.** Consider the well known vertex cover problem. In this problem we are given a graph G and we want a set $C \subseteq V(G)$ of vertices such that for any edge $(u, v) \in E(G)$, we have that either $u \in C$, $v \in C$, or both. Our goal is to find such a vertex cover C such that $|C|$ is minimum. In the decision variant, we are given a graph G and parameter k and want to decide whether or not G has a vertex cover of size at most k. It is shown in [3] that there is an algorithm which solves the vertex cover problem in $\mathcal{O}(n\sqrt{m} + 1.4656^k k^{\mathcal{O}(1)})$ time, via the use of branching and some reduction rules. Now note that we have only polynomial factors in the input size, and our exponential running time is restricted to our parameter k, the size of the vertex cover we are looking for. This shows that vertex cover is FPT.

In simpler terms, if a problem in NP is FPT, then we can restrict the exponential running time of our algorithm to some (ideally small) hardness parameter and incur only a polynomially bounded running time in the size of the original input. It was shown in [2] that finding an MAF for two phylogenetic trees is FPT when parameterized by the size of an optimal solution.

2.3 Related Work

To date, there has been an extensive amount of research into developing efficient algorithms (both theoretically and in practice) for computing MAFs for a pair of phylogenetic trees. As mentioned, Bordewich and Semple first proved that this problem was FPT when parameterized by the size of an MAF [2].

Several efficient branching algorithms exist for computing MAFs of rooted phylogenetic trees. The algorithm described in [10] made use of a simple branching rule to compute an MAF of (T_1, T_2) in $\mathcal{O}(3^k \cdot n)$ time. While not the most impressive bound obtained to date, it remains not only an FPT algorithm, but one that can in fact find any MAF of the input trees. This algorithm was improved upon in [8] to obtain a bound of $\mathcal{O}(2.42^k \cdot n)$ time by breaking the existing rule into three different branching rules. Further improvements to branching rules as well as the depth rule for selecting cherries to branch on, and the use of edge protection were introduced [11] to achieve a running time bound of $\mathcal{O}(2^k \cdot n)$. Similar strategies have also been used to yield an

algorithm for multifurcating (non-binary) phylogenetic trees that has a running time bound of $\mathcal{O}(2.42^k \cdot n)$.

Linz and Semple proposed the technique of cluster partitioning (or cluster reduction), whereby we can split our input trees into subtrees with common leaf label sets in order to partition the problem into smaller subproblems. We can then compute MAFs for these smaller trees independently and combine the results to obtain a global MAF [6]. This initial version of cluster reduction required a weight function in order to guarantee that the end result would indeed be a global MAF. Whidden, Zeh, and Beiko then proposed a modification to the original cluster partitioning technique, wherein instead of requiring a weight function, preference is given to MAFs which cut their root edge [12].

Our aim is to make use of the general ideas and techniques used in the existing branching algorithms and cluster reduction strategies, and adapt them to work in the context of the core MAF problem. Intuitively, these techniques should work for the core MAF problem. However, the improved branching algorithms for computing MAFs fairly explicitly exploit that they do not need to find every MAF, something our core MAF algorithm has to do at least implicitly. Similarly, the cluster reduction technique in its original form may miss some MAFs and thus needs to be adapted if we want to use it to speed up core MAF algorithms.

Chapter 3

Cluster Reduction and the Core Maximum Agreement Forest

3.1 Clusters and MAFs

We now turn our attention to the technique of cluster partitioning originally proposed by Linz and Semple [6]. The key idea here is that we can break our input trees into smaller subtrees and recursively compute MAFs of these smaller instances. Upon solving these simpler problems, we can then piece together a solution for the input trees. The definitions and lemmas which follow mainly focus on having two clusters. However, as we later show, this can easily be extended to work for the potentially many clusters that the optimal cluster partition of our input trees could have.

Definition 3.1 (Clusters)**.** Given a vertex $v \in V(T)$, we denote by X_v the set of leaves that have v as an ancestor in T. If we have trees T_1 and T_2 with nodes $v_1 \in V(T_1)$ and $v_2 \in V(T_2)$ such that $X_{v_1} = X_{v_2} \neq X$, then we define the top and bottom clusters for (T_1, T_2) as follows. Let $Y = X_{v_1} = X_{v_2}$. The bottom cluster, $(T_1^{\downarrow}(Y), T_2^{\downarrow}(Y))$ is obtained by taking $T_1(Y)$ and $T_2(Y)$ and adding the outgroup node, ρ, which has one incident edge that connects it a new root node that is in turn connected to the root in $T_1(Y)$ and $T_2(Y)$. The top cluster, which we denote as $(T_1^{\uparrow}(X \setminus Y), T_2^{\uparrow}(X \setminus Y))$ is obtained by first taking $T_1(X \setminus Y)$ and $T_2(X \setminus Y)$. Next, let u_1 and u_2 be the parent nodes of v_1 and v_2 respectively. We append to $T_1(X \setminus Y)$ and $T_2(X \setminus Y)$ a new leaf, α, whose parent is u_i in $T_i(X \setminus Y)$, for $i \in \{1, 2\}$. An example of a cluster partitioning of two phylogenetic trees, T_1 and T_2 is given in Figure 3.1.

In the case of the top cluster, α represents the entirety of $T_i^{\downarrow}(Y)$ in $T_i^{\uparrow}(X \setminus Y)$, for $i = \{1, 2\}$. We refer to the test to see whether $(T_1^{\downarrow}(Y), T_2^{\downarrow}(Y))$ has an MAF that isolates ρ as the *ρ-test*. It can be done by computing MAFs of $(T_1^{\downarrow}(Y) - \rho, T_2^{\downarrow}(Y) - \rho)$ and $(T_1^{\downarrow}(Y), T_2^{\downarrow}(Y))$. There exists an MAF of $(T_1^{\downarrow}(Y), T_2^{\downarrow}(Y))$ that isolates ρ if and only if the MAF of $(T_1^{\downarrow}(Y), T_2^{\downarrow}(Y))$ has one component more than the MAF of $(T_1^{\downarrow}(Y) - \rho, T_2^{\downarrow}(Y) - \rho)$.

Thus to compute an MAF for (T_1, T_2) we must find a minimal cut edge set for T_1 such that the forest obtained is also a forest obtainable from T_2, and this forest has the smallest possible number of connected components. Referring back to Figure 2.2,

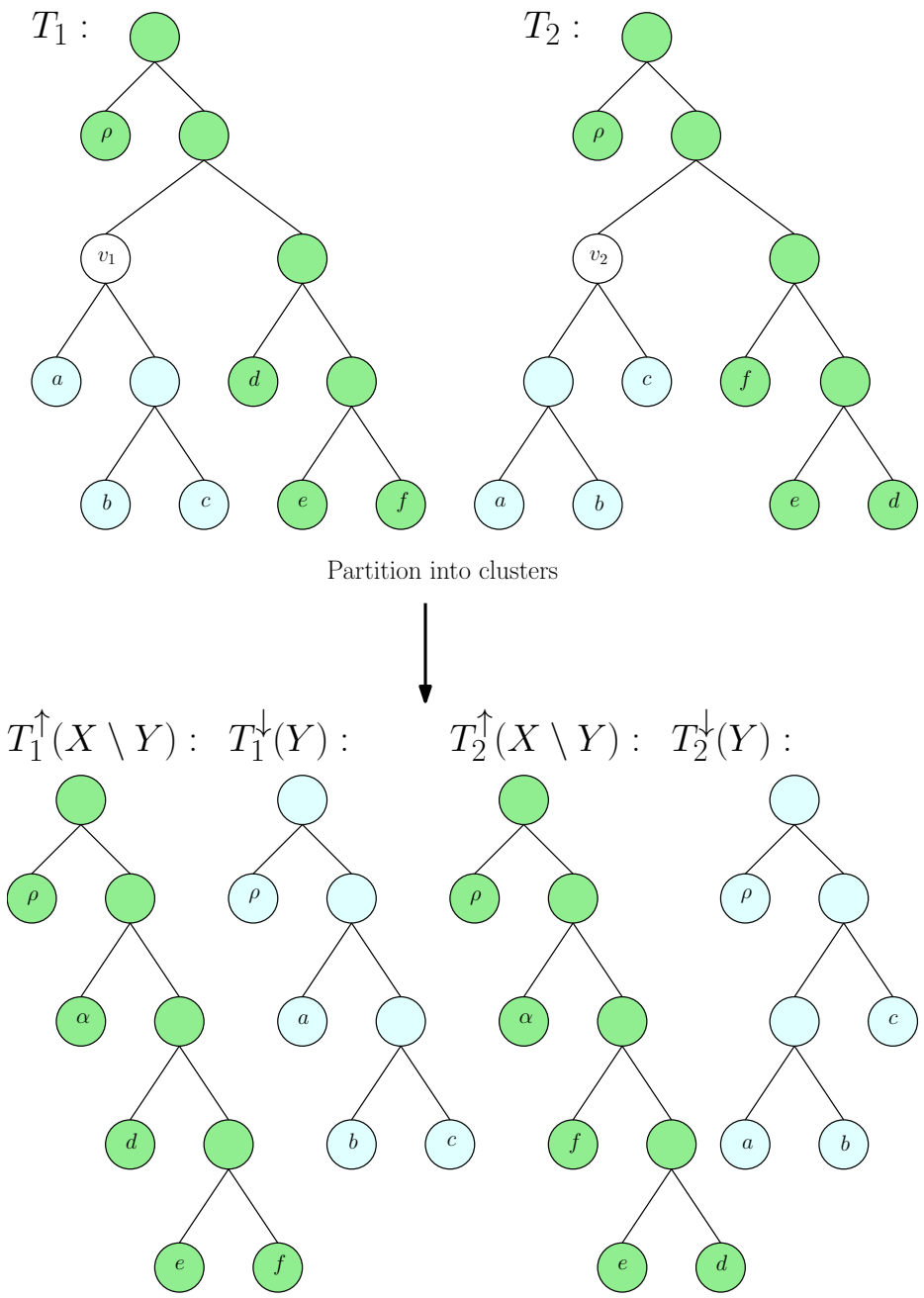


Figure 3.1: A set of phylogenetic trees (T_1, T_2) with leaf set $X = \{a, b, c, d, e, f\}$ (top) and a cluster partition of the trees with respect to the subtrees induced by v_1 and v_2 in T_1 and T_2, respectively, which have the common leaf set $Y = \{a, b, c\}$.

a cut edge set for $\mathcal{F}$, with respect to both T_1 and T_2, is $\{e_c, e_e\}$, where e_c and e_e are the parent edges of c and e respectively.

We now define the operation of gluing together AFs of the top and bottom clusters, which is vital to being able to use the MAFs obtained for each of our clusters in order to compute a global MAF for our input trees.

Definition 3.2 (Gluing Operation)**.** Let $\mathcal{F}^\uparrow$ and $\mathcal{F}^\downarrow$ be AFs of the clusters $(T_1^\uparrow(X\backslash Y), T_2^\uparrow(X \setminus Y))$ and $(T_1^\downarrow(Y), T_2^\downarrow(Y))$ of (T_1, T_2) respectively. Let $\mathcal{E}^\uparrow$ and $\mathcal{E}^\downarrow$ be their cut edge sets. If $\{\alpha\} \in \mathcal{F}^\uparrow$ or $\{\rho\} \in \mathcal{F}^\downarrow$ then let $\mathcal{E}_{\alpha,\rho} = \{e\}$, where e is the parent edge of v_1 in T_1, with $X_{v_1} = Y$. If neither α in the top cluster nor ρ in the bottom cluster are isolated, then let $\mathcal{E}_{\alpha,\rho} = \emptyset$. Then $\mathcal{F}^\uparrow \oplus \mathcal{F}^\downarrow = T_1 - (\mathcal{E}^\uparrow \cup \mathcal{E}^\downarrow \cup \mathcal{E}_{\alpha,\rho})$. The gluing operation is illustrated in Figure 3.2.

Definition 3.3 (Crossing the Boundary)**.** We say that a component $C \in \mathcal{F}$ crosses the boundary between the two clusters $(T_1^\uparrow(X \setminus Y), T_2^\uparrow(X \setminus Y))$ and $(T_1^\downarrow(Y), T_2^\downarrow(Y))$ if $(X \setminus Y) \cap \mathcal{L}(C) \neq \emptyset$ and $Y \cap \mathcal{L}(C) \neq \emptyset$. An example of a component that crosses the boundary of two clusters is given in Figure 3.2.

Recall that $T_i(X \setminus Y)$ and $T_i(Y)$ are subtrees of T_i, for $i \in \{1, 2\}$. The gluing operation removes the additional α and ρ leaves, and we are left with components that are common to (T_1, T_2). If we were to have an incompatible triplet with T_i, then this would result in a triplet incompatible with either $T_i^\uparrow(X \setminus Y)$ or $T_i^\downarrow(Y)$, for $i \in \{1, 2\}$. No component that does not cross the boundary can overlap, as this would have resulted in an overlap in its respective cluster. If there is some component C that does cross the boundary of our clusters, then again there can be no overlap, since we remove the additional α and ρ leaves when gluing, and $T_i(X \setminus Y)$ and $T_i(Y)$ are vertex-disjoint subtrees of T_i, for $i \in \{1, 2\}$.

Lemma 3.4 (Linz and Semple [6])**.** *Let (T_1, T_2) be a pair of phylogenetic trees, and let $(T_1^\uparrow(X \setminus Y), T_2^\uparrow(X \setminus Y))$ and $(T_1^\downarrow(Y), T_2^\downarrow(Y))$ be the clusters thereof. If $\mathcal{F}^\uparrow$ and $\mathcal{F}^\downarrow$ are AFs of $(T_1^\uparrow(X\setminus Y), T_2^\uparrow(X\setminus Y))$ and $(T_1^\downarrow(Y), T_2^\downarrow(Y))$ respectively, then $\mathcal{F} = \mathcal{F}^\uparrow \oplus \mathcal{F}^\downarrow$ is an AF of (T_1, T_2).*

Definition 3.5 (Restriction of an AF)**.** Given an AF, $\mathcal{F}$, of (T_1, T_2), we write $\mathcal{F}|_{(X\setminus Y)\uparrow}$ and $\mathcal{F}|_{Y\downarrow}$ to denote the restriction of the components of $\mathcal{F}$ to the top and bottom clusters defined by $Y \subseteq X$, respectively. Specifically, if $\mathcal{E}$ is the cut edge set of $\mathcal{F}$ and $E^\uparrow$, $E^\downarrow$ are the edge sets of $T_1^\uparrow(X \setminus Y)$ and $T^\downarrow(Y)$, respectively, then $\mathcal{F}|_{(X\setminus Y)\uparrow} = T_1^\uparrow(X \setminus Y) - (E^\uparrow \cap \mathcal{E})$ and $\mathcal{F}|_{Y\downarrow} = T_1^\downarrow(Y) - (E^\downarrow \cap \mathcal{E})$. If $e \in \mathcal{E}$, where e is the edge that

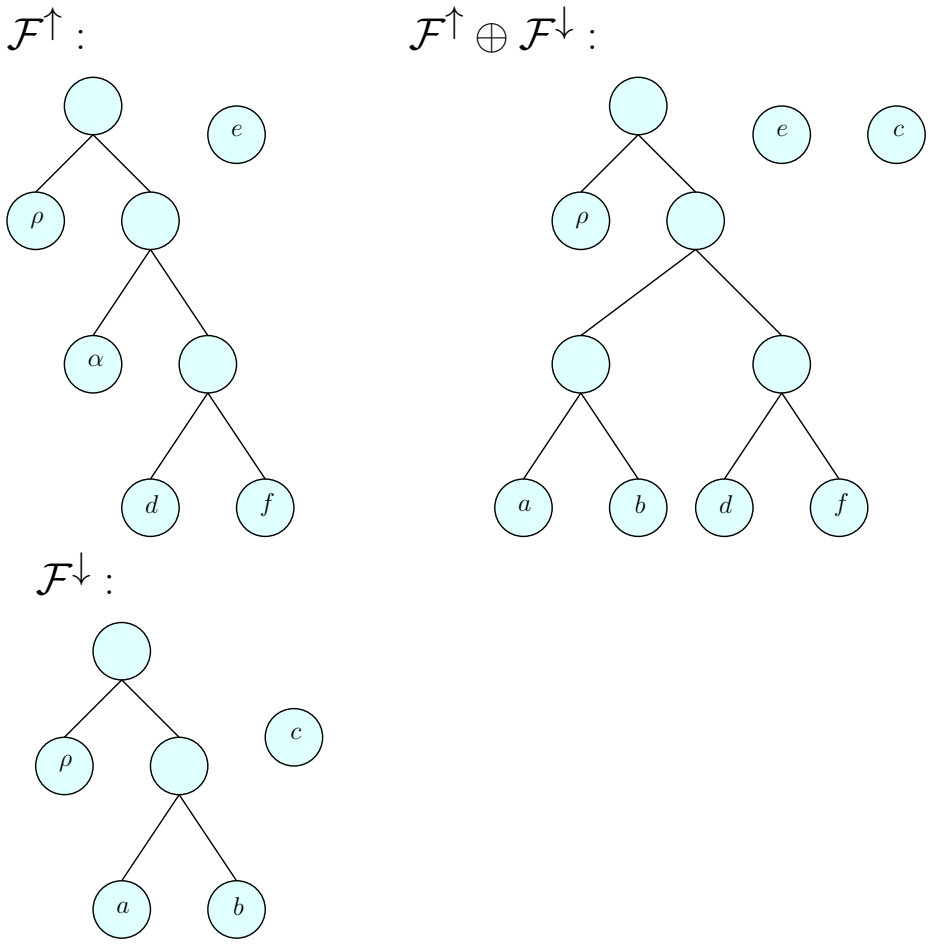


Figure 3.2: MAFs $\mathcal{F}^{\uparrow}$ and $\mathcal{F}^{\downarrow}$ of the clusters $(T_1^{\uparrow}(X \setminus Y), T_2^{\uparrow}(X \setminus Y))$ and $(T_1^{\downarrow}(Y), T_2^{\downarrow}(Y))$ from Figure 3.1 (left). Applying the gluing operation to these cluster MAFs yields the forest $\mathcal{F}^{\uparrow} \oplus \mathcal{F}^{\downarrow}$ (right). The non-singleton component in $\mathcal{F}^{\uparrow} \oplus \mathcal{F}^{\downarrow}$ crosses the boundary of the clusters.

crosses the boundary of our clusters, then we cut the parent edge of α in $\mathcal{F}|_{(X\setminus Y)\uparrow}$ and the parent edge of ρ in $\mathcal{F}|_{Y\downarrow}$. Restriction is illustrated in Figure 3.3.

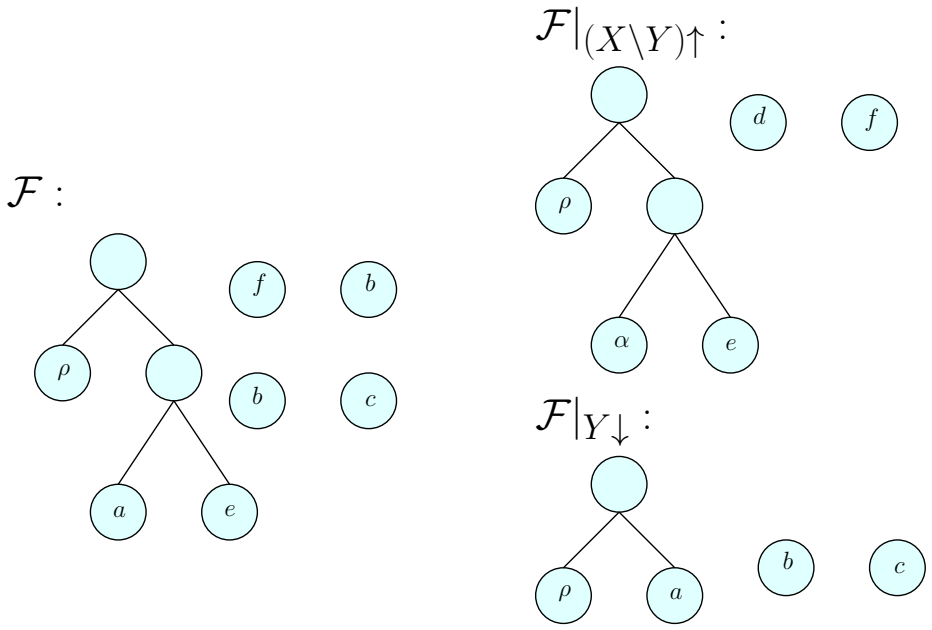


Figure 3.3: An AF, $\mathcal{F}$, of (T_1, T_2) from Figure 3.1 (left). Restricting $\mathcal{F}$ to $(T_1^{\uparrow}(X \setminus Y), T_2^{\uparrow}(X \setminus Y))$ and $(T_1^{\downarrow}(Y), T_2^{\downarrow}(Y))$ results in $\mathcal{F}|_{(X\setminus Y)\uparrow}$ and $\mathcal{F}|_{Y\downarrow}$ respectively (right).

The lemmas presented in the remainder of this section will be essential in achieving our goal of showing that it is indeed safe to use cluster reduction in the context of the core MAF problem.

Lemma 3.6. *Let $\mathcal{F}$ be an AF of (T_1, T_2). Then $\mathcal{F}|_{(X\setminus Y)\uparrow}$ and $\mathcal{F}|_{Y\downarrow}$ are AFs of $(T_1^{\uparrow}(X \setminus Y), T_2^{\uparrow}(X \setminus Y))$ and $(T_1^{\downarrow}(Y), T_2^{\downarrow}(Y))$, respectively.*

Proof. We show that $\mathcal{F}|_{(X\setminus Y)\uparrow}$ is an AF of $(T_1^{\uparrow}(X \setminus Y), T_2^{\uparrow}(X \setminus Y))$ and the result follows similarly for $\mathcal{F}|_{Y\downarrow}$ and $(T_1^{\downarrow}(Y), T_2^{\downarrow}(Y))$. It is sufficient to show that no two components of $\mathcal{F}|_{(X\setminus Y)\uparrow}$ overlap in $T_1^{\uparrow}(X \setminus Y)$ nor $T_2^{\uparrow}(X \setminus Y)$ and that every triplet is compatible with $T_1^{\uparrow}(X \setminus Y)$ and $T_2^{\uparrow}(X \setminus Y)$. Let A and B be two different arbitrary components of $\mathcal{F}|_{(X\setminus Y)\uparrow}$. Since $\mathcal{F}|_{(X\setminus Y)\uparrow}$ is a restriction of $\mathcal{F}$, it is evident that A and B are subtrees of A' and B', where A' and B' are components of $\mathcal{F}$. Assume that there is some overlap between A and B, that is there is a subtree C common to both. Then we have that C is also a subtree of both A' and B' and thus A' and B' overlap in $\mathcal{F}$ which could not have been an AF to begin with. Therefore, no two components of $\mathcal{F}^{\uparrow}$ overlap.

Now consider an arbitrary triplet, $ab|c$ of $\mathcal{F}|_{(X\setminus Y)\uparrow}$. We then have two cases

1. $\alpha \notin \{a, b, c\}$

2. $\alpha \in \{a, b, c\}$.

In case 1 it is clear that $ab|c$ must also be a triplet of $\mathcal{F}$ and so if it is not consistent with $T_1^{\uparrow}(X \setminus Y)$ or $T_2^{\uparrow}(X \setminus Y)$, then it cannot have been consistent with T_1 or T_2. For case 2 we know that there is some component $C \in \mathcal{F}$ that crosses the boundary of our clusters. Let $y \in C$ be an arbitrary leaf such that $y \in Y$. If we now replace α with y, then we have a triplet t of $\mathcal{F}$. If $ab|c$ is not compatible with $T_1^{\uparrow}(X \setminus Y)$ or $T_2^{\uparrow}(X \setminus Y)$, then t is not compatible with T_1 or T_2. □

If we focus on subtrees and triplets of a single phylogenetic tree then we obtain from Lemma 3.6 the following corollary.

Corollary 3.7. *Let $Y \subseteq X$ be a subset of the leaf label set of T. Let $\mathcal{L}(A), \mathcal{L}(B) \subseteq Y$ be the leaf sets associated with subtrees A and B of T. Then A and B overlap in T if and only if they overlap in $T|_Y$. Similarly, if we consider any triplet $ab|c$ of T such that $a, b, c \in Y$, then $ab|c$ is also a triplet of $T|_Y$.*

We now discuss the cases that arise when comparing the size of cluster AFs to the global AFs that result from the gluing process. If we isolate ρ in $\mathcal{F}^{\downarrow}$ and we isolate α in $\mathcal{F}^{\uparrow}$, then $|\mathcal{F}^{\downarrow} \oplus \mathcal{F}^{\uparrow}| = |\mathcal{F}^{\downarrow}| + |\mathcal{F}^{\uparrow}| - 2$, as both singletons disappear when we glue the AFs together. If $\mathcal{F}^{\downarrow}$ does not isolate ρ, but $\mathcal{F}^{\uparrow}$ does isolate α or vice-versa, then $|\mathcal{F}^{\downarrow} \oplus \mathcal{F}^{\uparrow}| = |\mathcal{F}^{\downarrow}| + |\mathcal{F}^{\uparrow}| - 1$ as the singleton component disappears after gluing. Finally, if we do not isolate α in $\mathcal{F}^{\uparrow}$ nor ρ in $\mathcal{F}^{\downarrow}$, then $|\mathcal{F}^{\downarrow} \oplus \mathcal{F}^{\uparrow}| = |\mathcal{F}^{\downarrow}| + |\mathcal{F}^{\uparrow}| - 1$, as the component which crosses the boundary is split amongst the two cluster AFs. Now let $\mathcal{F}$ be an MAF of (T_1, T_2). Then $\mathcal{F} = \mathcal{F}|_{(X \setminus Y)^{\uparrow}} \oplus \mathcal{F}|_{Y^{\downarrow}}$, and these restrictions are both AFs of their respective cluster by Lemma 3.6. This yields the following observation.

Observation 3.8. *Let $\mathcal{F}$ be an MAF of (T_1, T_2) and let $\mathcal{F}^{\uparrow}, \mathcal{F}^{\downarrow}$ be MAFs of $(T_1^{\uparrow}(X \setminus Y)$,*
$T_2^{\uparrow}(X \setminus Y))$ and $(T_1^{\downarrow}(Y), T_2^{\downarrow}(Y))$ respectively. Then

$$|\mathcal{F}| < |\mathcal{F}^{\uparrow}| + |\mathcal{F}^{\downarrow}| \leq |\mathcal{F}| + 2.$$

Building on Lemma 3.6 we now discuss the size of AFs obtained by restricting a global MAF to our clusters. In particular, we discuss the restriction of global MAFs resulting in cluster MAFs, or MAFs of the clusters with either e_α or e_ρ precut.

Lemma 3.9. *Let $\mathcal{F}$ be an MAF of (T_1, T_2) and let $k^{\downarrow}$ be the size of an MAF of $(T_1^{\downarrow}(Y),\ T_2^{\downarrow}(Y))$. Then $|\mathcal{F}|_{Y\downarrow}| \leq k^{\downarrow} + 1$. Similarly, let $k^{\uparrow}$ be the size of an MAF of $(T_1^{\uparrow}(X \setminus Y), T_2^{\uparrow}(X \setminus Y))$. Then $|\mathcal{F}|_{(X\setminus Y)\uparrow}| \leq k^{\uparrow} + 1$.*

Proof. We give the proof for the case of the restriction to the bottom cluster. The proof for the case of restricting to the top cluster follows similarly. It is sufficient to show that either $\mathcal{F}|_{Y\downarrow}$ is an MAF of $(T_1^{\downarrow}(Y), T_2^{\downarrow}(Y))$ or it is an AF that is one component larger than an MAF. By Lemma 3.6 we know that $\mathcal{F}|_{Y\downarrow}$ is an AF of $(T_1^{\downarrow}(Y), T_2^{\downarrow}(Y))$. Assume, by way of contradiction, that $|\mathcal{F}_{Y\downarrow}| > k^{\downarrow} + 1$. By Observation 3.8 we know that $|\mathcal{F}| < k^{\uparrow} + k^{\downarrow} \leq |\mathcal{F}| + 2$. We also know that $\mathcal{F} = \mathcal{F}|_{(X\setminus Y)\uparrow} \oplus \mathcal{F}|_{Y\downarrow}$ and thus $|\mathcal{F}| < |\mathcal{F}|_{(X\setminus Y)\uparrow}| + |\mathcal{F}|_{Y\downarrow}| \leq |\mathcal{F}| + 2$. Now even if $|\mathcal{F}|_{(X\setminus Y)\uparrow}| = k^{\uparrow}$, that is, it is an MAF of $(T_1^{\uparrow}(X \setminus Y), T_2^{\uparrow}(X \setminus Y))$, and $|\mathcal{F}_{Y\downarrow}| = k^{\downarrow} + 2$, then we have $|\mathcal{F}| < k^{\uparrow} + k^{\downarrow} + 2 \leq |\mathcal{F}| + 2$, which cannot be true. $\Rightarrow\Leftarrow$ □

In what follows we discuss MAFs of $(T_1^{\uparrow}(X \setminus Y)//\alpha, T_2^{\uparrow}(X \setminus Y)//\alpha)$ and $(T_1^{\downarrow}(Y)//\rho, T_2^{\downarrow}(Y)//\rho)$. We say $\mathcal{F}^{\uparrow-\alpha}$ is an MAF of $(T_1^{\uparrow}(X \setminus Y)//\alpha, T_2^{\uparrow}(X \setminus Y)//\alpha)$ if $|\mathcal{F}^{\uparrow-\alpha}|$ is minimum among all AFs of the top cluster where e_α is precut. Similarly, $\mathcal{F}^{\downarrow-\rho}$ is an MAF of $(T_1^{\downarrow}(Y)//\rho, T_2^{\downarrow}(Y)//\rho)$ if $|\mathcal{F}^{\downarrow-\rho}|$ is minimum among all AFs of the bottom cluster where e_ρ is precut.

Lemma 3.10. *Let (T_1, T_2) be a pair of phylogenetic trees and let $(T_1^{\uparrow}(X\setminus Y),\ T_2^{\uparrow}(X\setminus Y))$ and $(T_1^{\downarrow}(Y), T_2^{\downarrow}(Y))$ be the top and bottom clusters, respectively. If $\mathcal{F}$ is an MAF of (T_1, T_2) that has no component that crosses the boundary of the clusters, then $\mathcal{F}|_{(X\setminus Y)\uparrow}$ and $\mathcal{F}|_{Y\downarrow}$ are MAFs of $(T_1^{\uparrow}(X \setminus Y)//\alpha, T_2^{\uparrow}(X \setminus Y)//\alpha)$ and $(T_1^{\downarrow}(Y)//\rho, T_2^{\downarrow}(Y)//\rho)$, respectively.*

Proof. We give the proof for the case of $\mathcal{F}_{Y\downarrow}$ being an MAF of $(T_1^{\downarrow}(Y)//\rho, T_2^{\downarrow}(Y)//\rho)$. The proof for $\mathcal{F}_{(X\setminus Y)\uparrow}$ being an MAF of $(T_1^{\uparrow}(X \setminus Y)//\alpha, T_2^{\uparrow}(X \setminus Y)//\alpha)$ follows similarly.

Since $\mathcal{F}$ has no component that crosses the boundary, we know that $\mathcal{F}_{Y\downarrow}$ isolates ρ, hence we have an AF of $(T_1^{\downarrow}(Y)//\rho, T_2^{\downarrow}(Y)//\rho)$ by Lemma 3.6. Suppose, towards contradiction, that this is not an MAF with e_ρ precut. Then there exists some AF $\mathcal{F}^{\downarrow-\rho}$ of $(T_1^{\downarrow}(Y)//\rho, T_2^{\downarrow}(Y)//\rho)$ such that $|\mathcal{F}^{\downarrow-\rho}| < |\mathcal{F}|_{Y\downarrow}|$. If we take $\mathcal{F}' = \mathcal{F}|_{(X\setminus Y)\uparrow} \oplus \mathcal{F}^{\downarrow-\rho}$, then we claim that $\mathcal{F}'$ is an AF of (T_1, T_2), such that $|\mathcal{F}'| < |\mathcal{F}|$. Indeed, precutting e_ρ in the bottom cluster does not matter since α being isolated in the top cluster will result in no components crossing the boundary. Since $|\mathcal{F}| = |\mathcal{F}|_{(X\setminus Y)\uparrow}| + |\mathcal{F}|_{Y\downarrow}| - 2$ and $|\mathcal{F}'| = |\mathcal{F}|_{(X\setminus Y)\uparrow}| + |\mathcal{F}^{\downarrow-\rho}| - 2$ and $|\mathcal{F}^{\downarrow-\rho}| < |\mathcal{F}|_{Y\downarrow}|$, we have that $|\mathcal{F}'| < |\mathcal{F}|$. $\Rightarrow\Leftarrow$ □

Lemma 3.11. *Let (T_1, T_2) be a pair of phylogenetic trees and let $(T_1^{\uparrow}(X \setminus Y), T_2^{\uparrow}(X \setminus Y))$ and $(T_1^{\downarrow}(Y), T_2^{\downarrow}(Y))$ be the top and bottom clusters, respectively. If $\mathcal{F}$ is an MAF of (T_1, T_2) that has a component that crosses the boundary of the clusters, then $\mathcal{F}|_{(X\setminus Y)\uparrow}$ and $\mathcal{F}|_{Y\downarrow}$ are MAFs of $(T_1^{\uparrow}(X \setminus Y), T_2^{\uparrow}(X \setminus Y))$ and $(T_1^{\downarrow}(Y), T_2^{\downarrow}(Y))$, respectively.*

Proof. We give the proof for the case of $\mathcal{F}|_{Y\downarrow}$ being an MAF of $(T_1^{\downarrow}(Y), T_2^{\downarrow}(Y))$. The proof for $\mathcal{F}|_{(X\setminus Y)\uparrow}$ being an MAF of $(T_1^{\uparrow}(X \setminus Y), T_2^{\uparrow}(X \setminus Y))$ follows similarly.

By Lemma 3.6 we know that $\mathcal{F}|_{Y\downarrow}$ is indeed an AF of $(T_1^{\downarrow}(Y), T_2^{\downarrow}(Y))$. Suppose, towards contradiction, that there exists some AF $\mathcal{F}^{\downarrow}$ of $(T_1^{\downarrow}(Y), T_2^{\downarrow}(Y))$ such that $|\mathcal{F}^{\downarrow}| < |\mathcal{F}|_{Y\downarrow}|$. Then, if we take $\mathcal{F}' = \mathcal{F}|_{(X\setminus Y)\uparrow} \oplus \mathcal{F}^{\downarrow}$, we claim that $\mathcal{F}'$ is an AF of (T_1, T_2), such that $|\mathcal{F}'| < |\mathcal{F}|$. Indeed, since $|\mathcal{F}| = |\mathcal{F}|_{(X\setminus Y)\uparrow}| + |\mathcal{F}|_{Y\downarrow} - 1$ and $|\mathcal{F}'| = |\mathcal{F}|_{(X\setminus Y)\uparrow}| + |\mathcal{F}^{\downarrow}| - 1$ and $|\mathcal{F}^{\downarrow}| < |\mathcal{F}|_{Y\downarrow}|$, we have that $|\mathcal{F}'| < |\mathcal{F}|$. $\Rightarrow\Leftarrow$ □

Recall that Observation 3.4 states that gluing together any two cluster AFs results in a global AF. We now discuss the three scenarios wherein we can guarantee that our gluing operation will result in a global MAF, given particular AFs of our clusters.

Lemma 3.12. *Let (T_1, T_2) be a pair of phylogenetic trees and let $(T_1^{\uparrow}(X \setminus Y), T_2^{\uparrow}(X \setminus Y))$ and $(T_1^{\downarrow}(Y), T_2^{\downarrow}(Y))$ be the top and bottom clusters respectively. Assume that no MAF of (T_1, T_2) has a component that crosses the boundary of the clusters. Let $\mathcal{F}^{\uparrow-\alpha}$ be an MAF of $(T_1^{\uparrow}(X \setminus Y)//\alpha, T_2^{\uparrow}(X \setminus Y)//\alpha)$ and let $\mathcal{F}^{\downarrow-\rho}$ be an MAF of $(T_1^{\downarrow}(Y)//\rho, T_2^{\downarrow}(Y)//\rho)$. Then $\mathcal{F} = \mathcal{F}^{\uparrow-\alpha} \oplus \mathcal{F}^{\downarrow-\rho}$ is an MAF of (T_1, T_2).*

Proof. Since no MAF of (T_1, T_2) has a component that crosses the boundary of the clusters, it is clear that restricting any MAF of (T_1, T_2) to $(T_1^{\uparrow}(X \setminus Y), T_2^{\uparrow}(X \setminus Y))$ will result in α being isolated and restricting to $(T_1^{\downarrow}(Y), T_2^{\downarrow}(Y))$ will result in ρ being isolated.

Suppose, towards contradiction, that $\mathcal{F}$ is not an MAF of (T_1, T_2). Then there exists some AF, $\mathcal{F}'$ such that $|\mathcal{F}'| < |\mathcal{F}|$ and where $\mathcal{F}'$ has no component that crosses the boundary of the clusters. Now, if we take both $\mathcal{F}'|_{(X\setminus Y)\uparrow}$ and $\mathcal{F}'|_{Y\downarrow}$, both of which are AFs by Lemma 3.6, and which isolate α and ρ respectively, then these are also AFs of $(T_1^{\uparrow}(X \setminus Y)//\alpha, T_2^{\uparrow}(X \setminus Y)//\alpha)$ and $(T_1^{\downarrow}(Y)//\rho, T_2^{\downarrow}(Y)//\rho)$ respectively. Since $|\mathcal{F}'| < |\mathcal{F}|$, and since $\mathcal{F}'|_{(X\setminus Y)\uparrow} \oplus \mathcal{F}'|_{Y\downarrow} = \mathcal{F}'$, we know that either $|\mathcal{F}'|_{(X\setminus Y)\uparrow}| < |\mathcal{F}^{\uparrow-\alpha}|$ or $|\mathcal{F}'|_{Y\downarrow}| < |\mathcal{F}^{\downarrow-\rho}|$, thus contradicting the minimality of either $|\mathcal{F}^{\uparrow-\alpha}|$ or $|\mathcal{F}^{\downarrow-\rho}|$. $\Rightarrow\Leftarrow$ □

Lemma 3.13. *Let (T_1, T_2) be a pair of phylogenetic trees and let $(T_1^{\uparrow}(X \setminus Y), T_2^{\uparrow}(X \setminus Y))$ and $(T_1^{\downarrow}(Y), T_2^{\downarrow}(Y))$ be the top and bottom clusters respectively. Assume that*

every MAF of (T_1, T_2) has a component that crosses the boundary of the clusters. Let $\mathcal{F}^\uparrow$ be an MAF of $(T_1^\uparrow(X \setminus Y), T_2^\uparrow(X \setminus Y))$ such that $\{\alpha\} \notin \mathcal{F}^\uparrow$ and let $\mathcal{F}^\downarrow$ be an MAF of $(T_1^\downarrow(Y), T_2^\downarrow(Y))$ such that $\{\rho\} \notin \mathcal{F}^\downarrow$. Then $\mathcal{F} = \mathcal{F}^\uparrow \oplus \mathcal{F}^\downarrow$ is an MAF of (T_1, T_2).

Proof. Since every MAF of (T_1, T_2) has a component that crosses the boundary of our clusters, it is clear that restricting any MAF of (T_1, T_2) to $(T_1^\uparrow(X \setminus Y), T_2^\uparrow(X \setminus Y))$ will result in α not being isolated and restricting to $(T_1^\downarrow(Y), T_2^\downarrow(Y))$ will result in ρ not being isolated.

Suppose, towards contradiction, that $\mathcal{F}$ is not an MAF of (T_1, T_2). Then there exists some AF, $\mathcal{F}'$ such that $|\mathcal{F}'| < |\mathcal{F}|$, and such that $\mathcal{F}'$ has some component that crosses the boundary of the clusters. Now, if we take both $\mathcal{F}'|_{(X\setminus Y)\uparrow}$ and $\mathcal{F}'|_{Y\downarrow}$, then both of theses are AFs of their respective clusters, by Lemma 3.6. Since $|\mathcal{F}'| < |\mathcal{F}|$, and since $\mathcal{F}'|_{(X\setminus Y)\uparrow} \oplus \mathcal{F}'|_{Y\downarrow} = \mathcal{F}'$, we know that either $|\mathcal{F}'|_{(X\setminus Y)\uparrow}| < |\mathcal{F}^\uparrow|$ or $|\mathcal{F}'|_{Y\downarrow}| < |\mathcal{F}^\downarrow|$, thus contradicting the minimality of either $|\mathcal{F}^\uparrow|$ or $|\mathcal{F}^\downarrow|$. $\Rightarrow\Leftarrow$ □

Lemma 3.14. *Let (T_1, T_2) be a pair of phylogenetic trees and let $(T_1^\uparrow(X \setminus Y), T_2^\uparrow(X \setminus Y))$ and $(T_1^\downarrow(Y), T_2^\downarrow(Y))$ be the top and bottom clusters respectively. Assume that there exists at least one MAF of (T_1, T_2) that has a component that crosses the boundary of the clusters and at least one MAF of (T_1, T_2) that has no such component. Let $\mathcal{F}^\uparrow$ be an MAF of $(T_1^\uparrow(X \setminus Y), T_2^\uparrow(X \setminus Y))$ and let $\mathcal{F}^\downarrow$ be an MAF of $(T_1^\downarrow(Y), T_2^\downarrow(Y))$. Then $\mathcal{F} = \mathcal{F}^\uparrow \oplus \mathcal{F}^\downarrow$ is an MAF of (T_1, T_2).*

Proof. Unlike with Lemmas 3.12 and 3.13, we cannot guarantee that the restriction of an MAF of (T_1, T_2) to the clusters will or will not result in α or ρ being isolated in $(T_1^\uparrow(X \setminus Y), T_2^\uparrow(X \setminus Y))$ and $(T_1^\downarrow(Y), T_2^\downarrow(Y))$ respectively. Furthermore, we cannot guarantee that restricting $\mathcal{F}$ back to the clusters will result in the cluster MAFs we started with, indeed if only one of $\mathcal{F}^\uparrow$ or $\mathcal{F}^\downarrow$ isolate α or ρ, then the restriction of $\mathcal{F}$ to the clusters will result in both α and ρ being isolated, and thus one of the restricted AFs will be off from an MAF by one component. This does not pose any issues though, as gluing these restrictions will still result in $\mathcal{F}$ in the end.

Suppose, towards contradiction, that $\mathcal{F}$ is not an MAF of (T_1, T_2). Then there exists some AF, $\mathcal{F}'$ such that $|\mathcal{F}'| < |\mathcal{F}|$, $\mathcal{F}'$ may or may not have a component that crosses that boundary of the clusters. Now, if we take both $\mathcal{F}'|_{(X\setminus Y)\uparrow}$ and $\mathcal{F}'|_{Y\downarrow}$, then both of these are AFs of their respective clusters, by Lemma 3.6. Since $\mathcal{F}' = \mathcal{F}'|_{(X\setminus Y)\uparrow} \oplus \mathcal{F}'|_{Y\downarrow}$, we know by Observation 3.8 that $|\mathcal{F}'| < |\mathcal{F}'|_{(X\setminus Y)\uparrow}| + |\mathcal{F}'|_{Y\downarrow}| \leq |\mathcal{F}'| + 2$. We also know by Observation 3.8 that $|\mathcal{F}| < |\mathcal{F}^\uparrow| + |\mathcal{F}^\downarrow| \leq |\mathcal{F}| + 2$, and hence $|\mathcal{F}'| + 1 < |\mathcal{F}^\uparrow| + |\mathcal{F}^\downarrow| \leq |\mathcal{F}'| + 3$, since $|\mathcal{F}'| < |\mathcal{F}|$. Thus, in order for both

$\mathcal{F}^{\uparrow}$ and $\mathcal{F}^{\downarrow}$ to be MAFs, it must be that $|\mathcal{F}^{\uparrow}| + |\mathcal{F}^{\downarrow}| = |\mathcal{F}'| + 2$. Since (T_1, T_2) has at least one MAF that has a component that crosses the boundary, if $\mathcal{F}'$ is such an MAF, then the proof follows, as neither α nor ρ would be isolated in $\mathcal{F}'|_{(X\setminus Y)}$ or $\mathcal{F}'|_{Y\downarrow}$ respectively, and thus $|\mathcal{F}'|_{(X\setminus Y)\uparrow}| + |\mathcal{F}'|_{Y\downarrow}| = \mathcal{F}' + 1$.

Assume then that $\mathcal{F}'$ is an AF that has no such component. Then $|\mathcal{F}'|_{(X\setminus Y)\uparrow}| + |\mathcal{F}'|_{Y\downarrow}| = \mathcal{F}' + 2$, since both α and ρ are isolated in $\mathcal{F}'|_{(X\setminus Y)\uparrow}$ and $\mathcal{F}'|_{Y\downarrow}$, respectively. We claim that one of $\mathcal{F}'|_{(X\setminus Y)\uparrow}$ and $\mathcal{F}'|_{Y\downarrow}$ is in fact an MAF, and the other is an AF that has one component more than an MAF. Let $\mathcal{F}''$ and $\mathcal{F}'''$ be MAFs of (T_1, T_2) such that $\mathcal{F}''$ has a component that crosses the boundary of the clusters, and let $\mathcal{F}'''$ be an MAF that has no such component. As before, we have $|\mathcal{F}''|_{(X\setminus Y)\uparrow}| + |\mathcal{F}''|_{Y\downarrow}| = \mathcal{F}'' + 1$. We also have $|\mathcal{F}'''|_{(X\setminus Y)\uparrow}| + |\mathcal{F}'''|_{Y\downarrow}| = \mathcal{F}''' + 2$. Now note that $\mathcal{F}'' = \mathcal{F}''|_{(X\setminus Y)\uparrow} \oplus \mathcal{F}''|_{Y\downarrow}$, and thus by Lemma 3.9 and Observation 3.8, we know that both $\mathcal{F}''|_{(X\setminus Y)\uparrow}$ and $\mathcal{F}''|_{Y\downarrow}$ are MAFs of their respective cluster. We then have

$$|\mathcal{F}''| = |\mathcal{F}'''| \tag{3.1}$$

$$|\mathcal{F}''|_{(X\setminus Y)\uparrow}| + |\mathcal{F}''|_{Y\downarrow}| - 1 = |\mathcal{F}'''|_{(X\setminus Y)\uparrow}| + |\mathcal{F}'''|_{Y\downarrow}| - 2 \tag{3.2}$$

$$|\mathcal{F}''|_{(X\setminus Y)\uparrow}| + |\mathcal{F}''|_{Y\downarrow}| = |\mathcal{F}'''|_{(X\setminus Y)\uparrow}| + |\mathcal{F}'''|_{Y\downarrow}| - 1 \tag{3.3}$$

From (3.3) we see that one of $\mathcal{F}'''|_{(X\setminus Y)}$ or $\mathcal{F}'''|_{Y\downarrow}$ is an MAF, and the other has one additional component. This means that either no MAF of $(T_1^{\uparrow}(X\setminus Y), T_2^{\uparrow}(X\setminus Y))$ isolates α or no MAF of $(T_1^{\downarrow}(Y), T_2^{\downarrow}(Y))$ isolates ρ. Now let $\mathcal{F}^* = \mathcal{F}''|_{(X\setminus Y)} \oplus \mathcal{F}'''|_{Y\downarrow}$ and $\mathcal{F}^{**} = \mathcal{F}'''|_{(X\setminus Y)} \oplus \mathcal{F}''|_{Y\downarrow}$. Then we have $|\mathcal{F}^*| = |\mathcal{F}''|_{(X\setminus Y)}| + |\mathcal{F}'''|_{Y\downarrow}| - 1$ and $|\mathcal{F}^{**}| = |\mathcal{F}'''|_{(X\setminus Y)}| + |\mathcal{F}''|_{Y\downarrow}| - 1$. One of $\mathcal{F}^*$ or $\mathcal{F}^{**}$ is then an MAF that does not have a component that crosses the boundary of our clusters. Returning back to $\mathcal{F}'$, we now know one of $\mathcal{F}'|_{(X\setminus Y)\uparrow}$ or $\mathcal{F}'|_{Y\downarrow}$ is an MAF of its respective cluster, and the other is an AF with one additional component. We can apply the strategy above to determine which one is the MAF, assume that it is $\mathcal{F}'|_{Y\downarrow}$, the argument for $\mathcal{F}'|_{(X\setminus Y)\uparrow}$ is analogous. Then let $\mathcal{F}^{\uparrow'}$ be an MAF of $(T_1^{\uparrow}(X\setminus Y), T_2^{\uparrow}(X\setminus Y))$. If we take $\mathcal{F}'''' = \mathcal{F}^{\uparrow'} \oplus \mathcal{F}'|_{Y\downarrow}$, then we have $|\mathcal{F}''''| \leq |\mathcal{F}'|$, and since $|\mathcal{F}^{\uparrow'}| + |\mathcal{F}'|_{Y\downarrow}| = |\mathcal{F}''''| + 1$ and $|\mathcal{F}^{\uparrow}| + |\mathcal{F}^{\downarrow}| = |\mathcal{F}'| + 2$, we have that at least one of $\mathcal{F}^{\uparrow}$ or $\mathcal{F}^{\downarrow}$ could not have been an MAF of its respective cluster. $\Rightarrow\Leftarrow$

□

To summarize, we have shown that, for any MAF $\mathcal{F}$ of (T_1, T_2), we can take the restriction of $\mathcal{F}$ to our clusters and glue them together to get back to $\mathcal{F}$. Said

otherwise, restricting a global MAF and then gluing the resulting cluster AFs is equivalent to the identity function. Furthermore, we have shown that we can compute MAFs, with certain restrictions in place, of our clusters and glue them together to obtain a global MAF. We now introduce the core MAF problem, which the remainder of this thesis focuses on addressing.

3.2 Clusters and the Core MAF

We now introduce the (α, ρ)-test, which allows us to determine which version of our cluster partitioning in the context of the core MAF problem.

Definition 3.15 (The (α, ρ)-test)**.** Let (T_1, T_2) be a pair of phylogenetic trees, and let $(T_1^{\uparrow}(X \setminus Y), T_2^{\uparrow}(X \setminus Y))$ and $(T_1^{\downarrow}(Y), T_2^{\downarrow}(Y))$ be the top and bottom clusters thereof respectively. The (α, ρ)-*test* is a test used to determine which of the following situations we are in:

(i) no MAF of (T_1, T_2) has a component that crosses the boundary of the clusters,

(ii) every MAF of (T_1, T_2) has a component that crosses the boundary of the clusters, or

(iii) at least one MAF of (T_1, T_2) has a component that crosses the boundary and at least one MAF has no such component.

This is achieved by doing the following. We compute two MAFs of $(T_1^{\uparrow}(X \setminus Y), T_2^{\uparrow}(X \setminus Y))$, one where we precut the parent edge of α, and one where we require that α not be isolated, $\mathcal{F}^{\uparrow -\alpha}$ and $\mathcal{F}^{\uparrow}$ respectively. Similarly, we compute MAFs of $(T_1^{\downarrow}(Y), T_2^{\downarrow}(Y))$, one in which we precut the parent edge of ρ, and one where we require that ρ not be isolated, $\mathcal{F}^{\downarrow -\rho}$ and $\mathcal{F}^{\downarrow}$ respectively. We then take $\mathcal{F}^{-\alpha,-\rho} = \mathcal{F}^{\uparrow -\alpha} \oplus \mathcal{F}^{\downarrow -\rho}$ and $\mathcal{F} = \mathcal{F}^{\uparrow} \oplus \mathcal{F}^{\downarrow}$. If $|\mathcal{F}^{-\alpha,-\rho}| < |\mathcal{F}|$, then we are in case (i). If $|\mathcal{F}| < |\mathcal{F}^{-\alpha,-\rho}|$, then we are in case (ii). Finally, if $|\mathcal{F}^{-\alpha,-\rho}| = |\mathcal{F}|$, then we are in case (iii).

In order to obtain an efficient algorithm for computing the core MAF of two phylogenetic trees, we wish to make use of the cluster reduction strategies outlined in Section 3.1. To that end, we make use of the (α, ρ)-test to determine which versions of our clusters we should use in order to find the global core MAF.

Lemma 3.16. *Let (T_1, T_2) be a pair of phylogenetic trees, and let $(T_1^{\uparrow}(X \setminus Y), T_2^{\uparrow}(X \setminus Y))$ and $(T_1^{\downarrow}(Y), T_2^{\downarrow}(Y))$ be the top and bottom clusters thereof respectively. If the*

(α, ρ)-test results in a case (i) scenario, then the core MAF of (T_1, T_2) is obtained by taking the union of the cut edge sets for the core MAFs of $(T_1^{\uparrow}(X \setminus Y)//\alpha, T_2^{\uparrow}(X \setminus Y)//\alpha)$ and $(T_1^{\downarrow}(Y)//\rho, T_2^{\downarrow}(Y)//\rho)$ along with the edge e that crosses the boundary of our clusters.

Proof. We begin by demonstrating that the (α, ρ)-test does in fact correctly determine that every MAF of (T_1, T_2) does not have a component that crosses the boundary of the clusters. Suppose, towards contradiction, that this is not the case. Then there exists some MAF of (T_1, T_2), say $\mathcal{F}^*$, that has some component that crosses the boundary. If we now take $\mathcal{F}^*|_{(X\setminus Y)^{\uparrow}}$ and $\mathcal{F}^*|_{Y\downarrow}$, then by Lemma 3.11 these are MAFs of $(T_1^{\uparrow}(X \setminus Y), T_2^{\uparrow}(X \setminus Y))$ and $(T_1^{\downarrow}(Y), T_2^{\downarrow}(Y))$ respectively. Now let $\mathcal{F}$ and $\mathcal{F}^{-\alpha,-\rho}$ be the AFs of (T_1, T_2) obtained from the (α, ρ)-test. In order for $\mathcal{F}^*$ to be an MAF, it must be that $|\mathcal{F}^*| \leq |\mathcal{F}^{-\alpha,-\rho}|$. However, $|\mathcal{F}^{-\alpha,-\rho}| < |\mathcal{F}|$, and hence this would imply that $|\mathcal{F}^*| < |\mathcal{F}|$ and thus either $|\mathcal{F}^*|_{(X\setminus Y)^{\uparrow}}| < |\mathcal{F}^{\uparrow}|$ or $|\mathcal{F}^*|_{Y\downarrow}| < |\mathcal{F}^{\downarrow}|$, where $\mathcal{F}^{\uparrow}$ and $\mathcal{F}^{\downarrow}$ are the MAFs of $(T_1^{\uparrow}(X \setminus Y), T_2^{\uparrow}(X \setminus Y))$ and $(T_1^{\downarrow}(Y), T_2^{\downarrow}(Y))$ computed during the (α, ρ)-test. However, as α is not isolated in $\mathcal{F}^*|_{(X\setminus Y)^{\uparrow}}$ and ρ is not isolated in $\mathcal{F}^*|_{Y\downarrow}$, this contradicts the fact that $\mathcal{F}^{\uparrow}$ and $\mathcal{F}^{\downarrow}$ were MAFs of $(T_1^{\uparrow}(X \setminus Y), T_2^{\uparrow}(X \setminus Y))$ and $(T_1^{\downarrow}(Y), T_2^{\downarrow}(Y))$ that do not isolate α or ρ respectively. Thus no MAF of (T_1, T_2) has a component that crosses the boundary of the clusters.

We now show that it is sufficient to take the union of the cut edge sets of the core MAFs of $(T_1^{\uparrow}(X\setminus Y)//\alpha, T_2^{\uparrow}(X\setminus Y)//\alpha)$ and $(T_1^{\downarrow}(Y)//\rho, T_2^{\downarrow}(Y)//\rho)$ in order to obtain the core MAF of (T_1, T_2). By Lemma 3.12 we know that, if no MAF of (T_1, T_2) has a component that crosses the boundary of the clusters, then for any two MAFs $\mathcal{F}^{\uparrow-\alpha}$ and $\mathcal{F}^{\downarrow-\rho}$ of $(T_1^{\uparrow}(X \setminus Y)//\alpha, T_2^{\uparrow}(X \setminus Y)//\alpha)$ and $(T_1^{\downarrow}(Y)//\rho, T_2^{\downarrow}(Y)//\rho)$ respectively, $\mathcal{F} = \mathcal{F}^{\uparrow-\alpha} \oplus \mathcal{F}^{\downarrow-\rho}$ is an MAF of (T_1, T_2). Furthermore, we know by Lemma 3.10 that every MAF of any pair of phylogenetic trees that has no component that crosses the boundary of the clusters, can be restricted to obtain MAFs of the top cluster with e_α precut and the bottom cluster with e_ρ precut. Thus if every MAF of (T_1, T_2) has no component that crosses the boundary of the clusters, then the union of the cut edge sets of all MAFs of (T_1, T_2) will be the union of the cut edge sets of all MAFs of $(T_1^{\uparrow}(X \setminus Y)//\alpha, T_2^{\uparrow}(X \setminus Y)//\alpha)$ and $(T_1^{\downarrow}(Y)//\rho, T_2^{\downarrow}(Y)//\rho)$. □

Lemma 3.17. *Let (T_1, T_2) be a pair of phylogenetic trees, and let $(T_1^{\uparrow}(X \setminus Y), T_2^{\uparrow}(X \setminus Y))$ and $(T_1^{\downarrow}(Y), T_2^{\downarrow}(Y))$ be the top and bottom clusters thereof respectively. If the (α, ρ)-test results in a case (ii) scenario, then the core MAF of (T_1, T_2) is obtained by taking the union of the cut edge sets for the core MAFs of $(T_1^{\uparrow}(X \setminus Y), T_2^{\uparrow}(X \setminus Y))$ and $(T_1^{\downarrow}(Y), T_2^{\downarrow}(Y))$.*

Proof. We begin by demonstrating that the (α, ρ)-test correctly determines that every MAF of (T_1, T_2) has a component that crosses the boundary of the clusters. Suppose, towards contradiction, that this is not the case. Then there exists some MAF of (T_1, T_2), say $\mathcal{F}^*$, that does not contain a component that crosses the boundary of the clusters. If we now take $\mathcal{F}^*|_{(X\setminus Y)\uparrow}$ and $\mathcal{F}^*|_{Y\downarrow}$, then by Lemma 3.10 these are MAFs of $(T_1^{\uparrow}(X \setminus Y)//\alpha, T_2^{\uparrow}(X \setminus Y)//\alpha)$ and $(T_1^{\downarrow}(Y)//\rho, T_2^{\downarrow}(Y)//\rho)$ respectively. Now let $\mathcal{F}$ and $\mathcal{F}^{-\alpha,-\rho}$ be the AFs of (T_1, T_2) obtained from the (α, ρ)-test. In order for $\mathcal{F}^*$ to be an MAF, it must be that $|\mathcal{F}^*| \leq |\mathcal{F}|$. However, $|\mathcal{F}| < |\mathcal{F}^{-\alpha,-\rho}|$, and hence this would imply that $|\mathcal{F}^*| < |\mathcal{F}^{-\alpha,-\rho}|$ and thus either $|\mathcal{F}^*|_{(X\setminus Y)\uparrow}| < |\mathcal{F}^{\uparrow-\alpha}|$ or $|\mathcal{F}^*|_{Y\downarrow}| < |\mathcal{F}^{\downarrow-\rho}|$, where $\mathcal{F}^{\uparrow-\alpha}$ and $\mathcal{F}^{\downarrow-\rho}$ are the MAFs of $(T_1^{\uparrow}(X \setminus Y)//\alpha, T_2^{\uparrow}(X \setminus Y)//\alpha)$ and $(T_1^{\downarrow}(Y)//\rho, T_2^{\downarrow}(Y)//\rho)$ computed during the (α, ρ)-test. However, if we remove α and ρ from $\mathcal{F}^*|_{(X\setminus Y)\uparrow}$ and $\mathcal{F}^*|_{Y\downarrow}$ respectively, then we are left with AFs of $(T_1^{\uparrow}(X \setminus Y)//\alpha, T_2^{\uparrow}(X \setminus Y)//\alpha)$ and $(T_1^{\downarrow}(Y)//\rho, T_2^{\downarrow}(Y)//\rho)$ respectively, at least one of which is smaller than those computed during the (α, ρ)-test, thus yielding a contradiction. Thus every MAF of (T_1, T_2) has a component that crosses the boundary.

We now show that it is sufficient to take the union of the cut edge sets of the core MAFs of $(T_1^{\uparrow}(X \setminus Y), T_2^{\uparrow}(X \setminus Y))$ and $(T_1^{\downarrow}(Y), T_2^{\downarrow}(Y))$ in order to obtain the core MAF of (T_1, T_2). By Lemma 3.13 we know that, if every MAF of (T_1, T_2) has a component that crosses the boundary of the clusters, then for any two MAFs, $\mathcal{F}^{\uparrow}$ and $\mathcal{F}^{\downarrow}$ of $(T_1^{\uparrow}(X \setminus Y), T_2^{\uparrow}(X \setminus Y))$ and $(T_1^{\downarrow}(Y), T_2^{\downarrow}(Y))$ respectively, $\mathcal{F} = \mathcal{F}^{\uparrow} \oplus \mathcal{F}^{\downarrow}$ is an MAF of (T_1, T_2). Furthermore, we know by Lemma 3.11 that every MAF of any pair of phylogenetic trees that has a component that crosses the boundary of the clusters, can be restricted to obtain MAFs of the top and bottoms clusters. Thus if every MAF of (T_1, T_2) has a component that crosses the boundary of the clusters, then the union of the cut edge sets of all MAFs of (T_1, T_2) will be the union of the cut edge sets of all MAFs of $(T_1^{\uparrow}(X \setminus Y), T_2^{\uparrow}(X \setminus Y))$ and $(T_1^{\downarrow}(Y), T_2^{\downarrow}(Y))$. □

Lemma 3.18. *Let (T_1, T_2) be a pair of phylogenetic trees, and let $(T_1^{\uparrow}(X \setminus Y), T_2^{\uparrow}(X \setminus Y))$ and $(T_1^{\downarrow}(Y), T_2^{\downarrow}(Y))$ be the top and bottom clusters thereof respectively. If the (α, ρ)-test results in a case (iii) scenario, then the core MAF of (T_1, T_2) is obtained by taking the union of the cut edge sets for the core MAFs of $(T_1^{\uparrow}(X \setminus Y), T_2^{\uparrow}(X \setminus Y))$ and $(T_1^{\downarrow}(Y), T_2^{\downarrow}(Y))$.*

Proof. We begin by proving that the (α, ρ)-test does effectively determine that there is at least one MAF of (T_1, T_2) that does not have a component that crosses the boundary of the clusters, and that there is at least one MAF that does have such a component. Unlike with cases (i) and (ii), this follows trivially from the fact that

the AF obtained from gluing together the MAFs of $(T_1^{\uparrow}(X \setminus Y), T_2^{\uparrow}(X \setminus Y))$ and $(T_1^{\downarrow}(Y), T_2^{\downarrow}(Y))$ does not differ in size from the AF obtained from gluing together the MAFs of $(T_1^{\uparrow}(X \setminus Y)//\alpha, T_2^{\uparrow}(X \setminus Y)//\alpha)$ and $(T_1^{\downarrow}(Y)//\rho, T_2^{\downarrow}(Y)//\rho)$.

We now show that it is sufficient to take the union of the cut edge sets of of the core MAFs of $(T_1^{\uparrow}(X \setminus Y), T_2^{\uparrow}(X \setminus Y))$ and $(T_1^{\downarrow}(Y), T_2^{\downarrow}(Y))$ in order to obtain the core MAF of (T_1, T_2). By Lemma 3.14 we know that, if there is at least one MAF of (T_1, T_2) that has a component that crosses the boundary of the clusters, and at least one MAF of (T_1, T_2) that has no such component, then for any two MAFs, $\mathcal{F}^{\uparrow}$ and $\mathcal{F}^{\downarrow}$ of $(T_1^{\uparrow}(X \setminus Y), T_2^{\uparrow}(X \setminus Y))$ and $(T_1^{\downarrow}(Y), T_2^{\downarrow}(Y))$ respectively, $\mathcal{F} = \mathcal{F}^{\uparrow} \oplus \mathcal{F}^{\downarrow}$ is an MAF of (T_1, T_2). As the (α, ρ)-test showed that $|\mathcal{F}| = |\mathcal{F}^{-\alpha,-\rho}|$, and $|\mathcal{F}^{-\alpha,-\rho}| = |\mathcal{F}^{\uparrow-\alpha}| + |\mathcal{F}^{\downarrow-\alpha}|$, since the α and ρ singletons are not included, we know that $|\mathcal{F}^{\uparrow-\alpha}| + |\mathcal{F}^{\downarrow-\rho}| = |\mathcal{F}^{\uparrow}| + |\mathcal{F}^{\downarrow}| + 1$. This means that one of either $\mathcal{F}^{\uparrow-\alpha} \cup \{\alpha\}$ or $\mathcal{F}^{\downarrow-\rho} \cup \{\rho\}$ is in fact an MAF of its respective cluster, and hence will be found by computing MAFs of the top and bottom clusters. Thus if there is at least one MAF of (T_1, T_2) that has a component that crosses the boundary of the clusters, and at least one MAF that has no such component, then the union of the cut edge sets of all MAFs of (T_1, T_2) will be the union of the cut edge sets of all MAFs of $(T_1^{\uparrow}(X \setminus Y), T_2^{\uparrow}(X \setminus Y))$ and $(T_1^{\downarrow}(Y), T_2^{\downarrow}(Y))$. □

Combining the results of Lemmas 3.16, 3.17, and 3.18 we now prove the following Theorem.

Theorem 3.19 (Safeness of Cluster Reduction for Core MAF)**.** *Let k be the size of the core MAF for phylogenetic trees (T_1, T_2) and k' be the maximum size among all the core MAFs in a minimal cluster partition of (T_1, T_2). An algorithm that computes the core MAF in $\mathcal{O}(f(k) \cdot \text{poly}(n))$ time implies that there is an algorithm that computes the core MAF in $\mathcal{O}(f(k') \cdot \text{poly}(n))$ time.*

Proof. Let $\mathcal{P}_C = \{(T_1^{Y_1}, T_2^{Y_1}), ..., (T_1^{Y_m}, T_2^{Y_m})\}$, $Y_1, ..., Y_m \subseteq X$, $\bigcap_{i=1}^{m} Y_i = \emptyset$, be a minimal cluster partition of (T_1, T_2). Without loss of generality, assume $(T_1^{Y_1}, T_2^{Y_1})$ is the topmost cluster which contains the original ρ.

For any leaf cluster, we compute the core MAF of this cluster if we are in case (ii) or (iii), and we compute the core MAF of this cluster with e_ρ precut if we are in a case (i) scenario. For any non-leaf cluster that is not the topmost cluster, whether we precut e_ρ is deteremined based on the (α, ρ)-test for this cluster and its parent cluster, in the case of the topmost cluster, e_ρ is never precut. For each non-leaf cluster, we precut e_{α_i}, where α_i corresponds to the i^{th} child cluster of this cluster, in accordance with the results of the (α, ρ)-test between this cluster and its i^{th} child cluster.

Once all of the necessary core MAFs, as outlined above, are computed, we take the union across all cut edge sets to obtain the core MAF of (T_1, T_2). The correctness of this process comes from combining Lemmas 3.16, 3.17, and 3.18. We need only argue the run time implication. For any cluster $(T_1^{Y_i}, T_2^{Y_i})$, we can clearly compute the required core MAF in $\mathcal{O}(f(k_i) \cdot \text{poly}(n))$ time, where k_i is the size of the core MAF. Thus if $k' = \max_{i \in [m]}(k_i)$ then the total running time for computing core MAFs is dominated by $f(k') \cdot \text{poly}(n)$. □

Chapter 4

Branching Algorithms for Core MAF

In this chapter we describe, in three stages, how to develop an efficient branching algorithm for solving the core MAF problem, along with an analysis of its running time and proof of correctness. We begin with a naive approach, wherein we effectively enumerate all MAFs. Following this, we introduce a set of three branching rules that allow us obtain a better bound on the size of our search tree, and consequently, a better running time bound for our algorithm. Finally, we address the worst case that arises using the initial three branching rules by introducing the notion of edge protection, along with a fourth branching rule and depth rule that allow us to more easily show that this protection scheme does indeed yield an improvement in our running time. To this end, we first provide some important definitions and notation.

Definition 4.1 (Expanding and contracting cherries)**.** By *contracting* a cherry (a, c), we mean that we remove both a and c and relabel their parent node as (a, c). We refer to reversing this process as *expanding* (a, c). These two operations are illustrated in Figure 4.1.

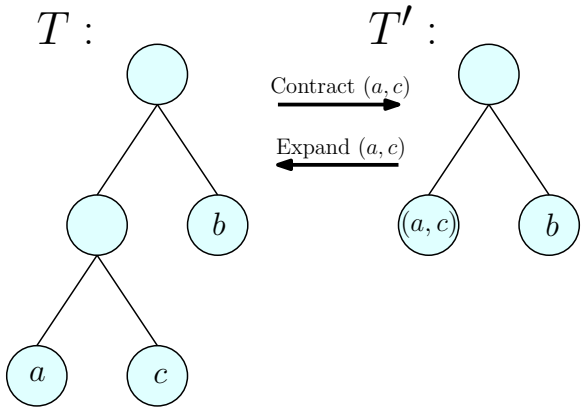


Figure 4.1: T' is obtained from T by contracting the cherry (a, c). Conversely, T is obtained from T' by expanding the leaf (a, c).

Lemma 4.2. *Let (a, c) be a trivial cherry of (T_1, T_2) and let T_i' be the tree obtained from T_i by contracting (a, c), for $i \in \{1, 2\}$. Then (a, c) is a cherry of every MAF*

of (T_1, T_2)*, and MAFs of* (T_1, T_2) *and* (T_1', T_2') *can be converted into each other by expanding or contracting* (a, c)*.*

Proof. Let $\mathcal{F}$ be an AF of (T_1, T_2). Then there are two possibilities: either c is a singleton in $\mathcal{F}$ or it is not (analogously a is either a singleton or it is not). Let $\mathcal{F}'$ be the forest obtained by contracting (a, c) in $\mathcal{F}$, if (a, c) is a cherry in $\mathcal{F}$. If $\{c\} \in \mathcal{F}$, then we remove c and relabel a as (a, c) to obtain $\mathcal{F}'$. We claim that $\mathcal{F}'$ is an AF of (T_1', T_2'). Since T_i' is a subtree of T_i, for $i \in \{1, 2\}$, $\mathcal{F}'$ is a subforest of $\mathcal{F}$, and $\mathcal{F}$ is an AF of (T_1, T_2), there can be no triplets in $\mathcal{F}'$ that are incompatible with T_1' or T_2'. Thus, it remains only to show that there are no components in $\mathcal{F}'$ that overlap in T_1' or T_2'. Since T_i' is a subtree of T_i and $\mathcal{F}'$ is a subforest of $\mathcal{F}$, if two components, $t_j, t_k \in \mathcal{F}'$ overlap in T_i', then the components of $\mathcal{F}$ that contain t_j and t_k would overlap in T_i. Hence, $\mathcal{F}'$ is an AF of (T_1', T_2'). Note that, $|\mathcal{F}'| \leq |\mathcal{F}|$. If a or c is a singleton in $\mathcal{F}$, then the inequality is strict.

Next, let $\mathcal{F}'$ be an AF of (T_1', T_2'). We claim that the forest $\mathcal{F}$ obtained by expanding (a, c) in $\mathcal{F}'$ is an AF of (T_1, T_2). As the only component of $\mathcal{F}'$ we alter is the component containing the leaf (a, c), and a and c are not present in $\mathcal{F}'$, we do not create any overlap by expanding (a, c). Thus it remains only to show that there are no triplets incompatible with T_1 or T_2. As (a, c) is a cherry in both T_1 and T_2, the only way it could be part of an incompatible triplet, $a|xy$ or $c|xy$ with T_i would be if the contracted node (a, c) belonged to some triplet, $(a, c)|xy$ or $(a, c)x|y$ in $\mathcal{F}'$ that was incompatible with T_i'. As $\mathcal{F}$ differs from $\mathcal{F}'$ only by the expansion of the cherry (a, c), there can be no other triplets in $\mathcal{F}$ that are incompatible with T_i, as this same triplet would be incompatible with T_i', since T_i is a supertree of T_i', for $i \in \{1, 2\}$. Thus $\mathcal{F}$ is indeed an AF of (T_1, T_2), and moreover $|\mathcal{F}| = |\mathcal{F}'|$.

We now prove that (a, c) is a cherry of every MAF of (T_1, T_2). Suppose that we do isolate c in an MAF, $\mathcal{F}$ of (T_1, T_2) (the case for isolating a is analogous). Then if we take the forest $\mathcal{F}'$ obtained by removing c from $\mathcal{F}$ and relabeling a as (a, c), we know by the above argument that $\mathcal{F}'$ is an AF of (T_1', T_2') and that $|\mathcal{F}'| < |\mathcal{F}|$. We also know by the above that, since $\mathcal{F}'$ is an AF of (T_1', T_2'), the forest $\mathcal{F}'^*$ obtained by expanding (a, c) in $\mathcal{F}'$ is an AF of (T_1, T_2). Since $|\mathcal{F}'^*| = |\mathcal{F}'| < |\mathcal{F}|$, $\mathcal{F}$ could not have been an MAF. Moreover, this means that our AF conversion always results in AFs with the same number of components if we start with an MAF; as we never delete singletons to convert an MAF of (T_1, T_2) into an AF of (T_1', T_2').

Now, let $\mathcal{F}'$ be an MAF of (T_1', T_2'). By the above argument, we know that we can expand (a, c) in $\mathcal{F}'$ to obtain an AF, $\mathcal{F}$, of (T_1, T_2) such that $|\mathcal{F}| = |\mathcal{F}'|$. We claim

that $\mathcal{F}$ is an MAF of (T_1, T_2). Suppose, towards contradiction, that this is not the case. Then there exists some AF, $\mathcal{F}^*$ of (T_1, T_2) such that $|\mathcal{F}^*| < |\mathcal{F}|$. We know that the AF $\mathcal{F}^{*'}$, obtained by contracting (a, c) in $\mathcal{F}^*$, is an AF of (T_1', T_2'). Moreover, we know that $|\mathcal{F}^{*'}| \leq |\mathcal{F}^*| < |\mathcal{F}|$ and since $|\mathcal{F}| = |\mathcal{F}'|$, this implies that $|\mathcal{F}^{*'}| < |\mathcal{F}'|$. $\Rightarrow\Leftarrow$

In a similar manner, if we start with an MAF, $\mathcal{F}$ of (T_1, T_2), then we claim that the forest $\mathcal{F}'$ obtained by contracting (a, c) in $\mathcal{F}'$ is an MAF of (T_1', T_2'). Again, assume towards contradiction that this is not the case. Then there is an AF, $\mathcal{F}'^*$ of (T_1', T_2') such that $|\mathcal{F}'^*| < |\mathcal{F}'|$. We can then expand (a, c) in $\mathcal{F}'^*$ to obtain the forest $\mathcal{F}^*$ which is an AF of (T_1, T_2). Since we know that $|\mathcal{F}^*| = |\mathcal{F}'^*|$, $|\mathcal{F}| \geq |\mathcal{F}'|$, and $|\mathcal{F}'^*| < |\mathcal{F}'|$, we have that $|\mathcal{F}^*| < |\mathcal{F}|$. $\Rightarrow\Leftarrow$ □

Lemma 4.2 shows that to find the core MAF of two phylogenetic trees, T_1 and T_2, we can start by contracting trivial cherries in both trees, find the core MAF of the trees T_1' and T_2' that result from the contractions, and finally expand the contracted cherries in the computed core MAF. Said otherwise, the cut edge sets of the core MAF of (T_1, T_2) and that of (T_1', T_2') are the same. This result will be used in order to simplify the steps of the algorithms that are presented in this chapter.

Definition 4.3 (Pendant Subtrees)**.** If B is a subtree of T with exactly one edge $(x, y) \in E(T)$ such that $x \notin B$, $y \in B$, and x is the parent of y, then B is a *pendant subtree* of T. In this case we call (x, y) the parent edge of B, and denote it as e_B. If x is on the path from a to c in T, then we call B a pendant subtree of the path from a to c. If B is a pendant subtree of the path from a to c and B contains a single leaf, b, then we call b a pendant leaf of the path from a to c. Figure 4.2 gives an example of such a subtree.

We denote by $\textsc{Maf}(\mathcal{F}_1, \mathcal{F}_2, \mathcal{E}^*, k)$ the instance of the standard MAF problem with forests $\mathcal{F}_1$ and $\mathcal{F}_2$, cut edge set $\mathcal{E}^*$, and parameter k as inputs. The goal is to decide whether there exists an AF of $(\mathcal{F}_1, \mathcal{F}_2)$ that has size at most k. Our initial invocation is $\textsc{Maf}(T_1, T_2, \emptyset, k)$, for some integer k.

A simple core MAF algorithm can be designed by way of applying the standard FPT MAF algorithm outlined in [10] exhaustively. In particular, Theorem 8 in [10] shows that one can compute an MAF for two rooted phylogenetic trees in $\mathcal{O}(3^k \cdot n)$ time, where k is the size of an MAF. We argue that this algorithm can in fact be used to enumerate all MAFs of two phylogenetic trees. By performing this enumeration, we can compute the core MAF by cutting all edges of T_2 that belong to the cut edge

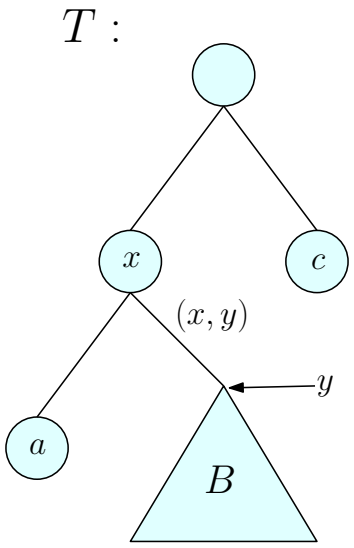


Figure 4.2: B is a pendant subtree on the path from a to c in T.

set of at least one MAF of (T_1, T_2). We outline the algorithm in Algorithm 4.1 and provide a proof of its correctness in Theorem 4.4.

The cases corresponding to lines 24–31 of Algorithm 4.1 are illustrated in Figure 4.3. Each dashed red line represents one of the cuts we make in $\mathcal{F}_2$.

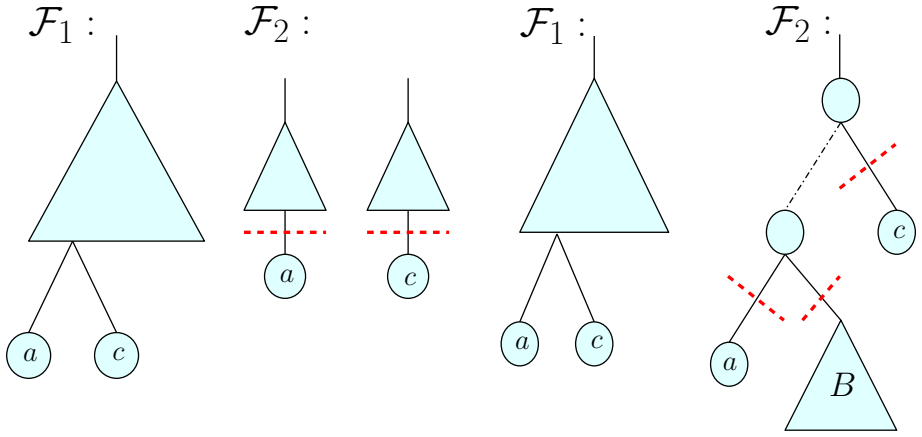


Figure 4.3: The cuts made in the recursive calls on lines 25–26 (left) and lines 29–31 (right) of Algorithm 4.1.

Theorem 4.4. *Let (T_1, T_2) be two phylogenetic trees and let k be the size of an MAF of (T_1, T_2). Then, the core MAF of (T_1, T_2) can be computed in $\mathcal{O}(3^k \cdot n)$ time.*

Proof. By Lemma 4.2, we know that every trivial cherry will appear in the same connected component of every MAF of (T_1, T_2). Thus it is always safe to contract

Algorithm 4.1: Naive Core MAF Algorithm

```
Data: Phylogenetic Trees (T_1, T_2)
Result: S, the core MAF for (T_1, T_2)
E ← ∅
k ← 0
success ← 0
do
    MAF(T_1, T_2, ∅, k)
    k ← k + 1
while success = 0;
S ← F_2//E
return S
Function MAF(F_1, F_2, E*, k):
    if k < 0 then
        return
    do
        contract (a, c) in F_1 and F_2
    while ∃ a trivial cherry (a, c);
    for each singleton c of F_2 do
        remove c from both F_1 and F_2
    if F_1 = F_2 then
        E ← E ∪ E*
        success ← 1
        return
    else
        if ∃ a non-trivial cherry (a, c) in F_1 then
            if a and c belong to separate components of F_2 then
                MAF(F_1, F_2//e_a, E* ∪ {e_a}, k − 1)
                MAF(F_1, F_2//e_c, E* ∪ {e_c}, k − 1)
            else
                let B be a pendant subtree on the path from a to c in F_2
                MAF(F_1, F_2//e_a, E* ∪ {e_a}, k − 1)
                MAF(F_1, F_2//e_B, E* ∪ {e_B}, k − 1)
                MAF(F_1, F_2//e_c, E* ∪ {e_c}, k − 1)
        else
            MAF(F_1, F_2, E*, k)
```

trivial cherries whenever one is selected by our algorithm. Furthermore, whenever we have a singleton component in $\mathcal{F}_2$, we can cut the corresponding leaf's parent edge in $\mathcal{F}_1$ and remove this leaf as we cannot obtain an AF otherwise.

We now consider the case where (a, c) is a cherry in $\mathcal{F}_1$, but a and c are neither singletons, nor do they form a cherry in $\mathcal{F}_2$. Theorem 1 in [10] shows that, in order for this to occur, either a or c must have a sibling subtree B, which may be a pendant subtree on the path from a to c in $\mathcal{F}_2$. Theorem 1 in [10] states that we can always obtain an MAF by cutting either e_a, e_B, or e_c as Algorithm 4.1 does. In order to prove that this is the case, Theorem 1 in [8] demonstrates that, for any MAF with cut edge set $\mathcal{E}$ such that $\mathcal{E} \cap \{e_a, e_B, e_c\} = \emptyset$, there exists some edge $e_x \in \mathcal{E}$ such that we can replace e_x with an edge from $\{e_a, e_B, e_c\}$ and obtain the same forest. Thus, this proves the stronger claim that *every* MAF can be obtained by cutting a set of edges that includes one of these edges.

In the case where a and c are in separate components, and thus there is no path from a to c in $\mathcal{F}_2$, we argue that only the recursive calls that cut e_a and e_c are necessary. Indeed, Lemma 3.6 in [8] demonstrates that we can always find an MAF by performing one of these cuts. In a manner similar to Theorem 1 in [10], this is demonstrated by showing that, for an MAF with cut edge set $\mathcal{E}$, we can always replace one of the edges in $\mathcal{E}$ with either e_a or e_c, whenever $\mathcal{E} \cap \{e_a, e_c\} = \emptyset$, and obtain the same forest. Thus, once again the stronger claim that every MAF can be obtained by cutting a set of edges that includes either e_a or e_c holds.

Since every MAF can be obtained by making the cuts that our algorithm makes, we can enumerate all MAFs of (T_1, T_2) by returning the MAFs that result from each successful leaf in our search tree. Since the core MAF is, by definition, obtained by making all cuts necessary to obtain every MAF, we can union the cut edges sets found across all of our successful branches to obtain the core MAF of (T_1, T_2). As Algorithm 4.1 still has a search tree with depth at most k and each invocation makes at most 3 recursive calls, each of which takes at most n^2 time, we obtain a running time bound of $\mathcal{O}(3^k \cdot n^2)$. Through a more careful implementation, which maintains a queue of all cherries in $\mathcal{F}_1$, one can improve the time used by each recursive call to be linear in n, thus yielding an algorithm which runs in $\mathcal{O}(3^k \cdot n)$ time. □

Algorithm 4.1 effectively combines the cuts from every success occurring in the search tree produced by the algorithm presented in [10] to obtain the running time stated in Theorem 4.4. Theorem 4.4 does not present an especially impressive result, but it is sufficient to yield the following corollary.

Corollary 4.5. *The core MAF problem is FPT when parameterized by the size of an MAF of the input trees.*

Corollary 4.5 shows not only that the core MAF problem is FPT, but that we need only parameterize our algorithm by the size of an MAF of our input trees; not the size of the core MAF. This is significant as the core MAF can have a far greater size than an MAF in general. By being able to parameterize the core MAF problem by the size of an MAF, we know at the very least that the exponential part of our running time is no worse than it is to compute an MAF using the same strategy. Furthermore, we use Algorithm 4.1 as a building block towards our final, significantly improved algorithm. To this end, we now present some improved branching rules for computing MAFs that can further improve our core MAF algorithm.

4.1 Better Branching Rules

It was shown in [8] that an MAF for two phylogenetic trees can be found in $\mathcal{O}(2.42^k n)$ time using improved branching rules. We can again make use of the strategy applied there in order to obtain the same running time bound for core MAF. The key improvement here is that we distinguish subcases when (a, c) is a cherry of $\mathcal{F}_1$ but the path from a to c has exactly one pendant subtree in $\mathcal{F}_2$ (**AB** case), at least two pendant subtrees in $\mathcal{F}_2$ (**ABC** case), or when no such path exists (**AC** case). These changes are outlined in Algorithm 4.2, and the cuts made in each of the three cases on which we branch are depicted in Figure 4.4.

As before we also remove singletons as they arise as well as contract trivial cherries to a single node. The **AC** case corresponds directly with the separate components case in our previous algorithm. The improvement we obtain here is a result of introducing a special case for when there is exactly one pendant subtree on the path from a to c in $\mathcal{F}_2$, what we refer to as the **AB** case, as we only make two recursive calls as opposed to three, one which cuts e_a and one which cuts e_B. The bottleneck remains the **ABC** case, as we still make three recursive calls. However, now we can guarantee that one of those recursive calls makes at least two cuts, and thus results in a better overall running time in the worst case scenario. The key difference between our branching rules and those used in [8] lies in the **AB** case. Our **AB** makes the additional call that cuts e_a, unlike the **B** case of [8], where it was only necessary to find an MAF, whereas in the context of the core MAF problem, we are required to find a cut edge set for every possible MAF.

Algorithm 4.2: Core MAF Algorithm with improved branching rules

```
Data: Phylogenetic Trees (T_1, T_2)
Result: S, the core MAF for (T_1, T_2)
E ← ∅
k ← 0
success ← 0
do
    MAF(T_1, T_2, ∅, k)
    k ← k + 1
while success = 0;
S ← T_2//E
return S
Function MAF(F_1, F_2, E*, k):
    if k < 0 then
        return
    do
        contract (a, c) in F_1 and F_2
    while ∃ a trivial cherry (a, c);
    for each singleton c of F_2 do
        remove c from F_1 and F_2
    if F_1 = F_2 then
        E ← E ∪ E*
        success ← 1
        return
    else
        if ∃ a non-trivial cherry (a, c) in F_1 then
            case AC do
                MAF(F_1, F_2//e_a, E* ∪ {e_a}, k − 1)
                MAF(F_1, F_2//e_c, E* ∪ {e_c}, k − 1)
            case AB do
                MAF(F_1, F_2//e_a, E* ∪ {e_a, e_c}, k − 1)
                MAF(F_1, F_2//e_B, E* ∪ {e_B}, k − 1)
            case ABC do
                MAF(F_1, F_2//e_a, E* ∪ {e_a}, k − 1)
                MAF(F_1, F_2//{e_B1, ..., e_Bm}, E* ∪ {e_B1, ..., e_Bm}, k − m)
                MAF(F_1, F_2//e_c, E* ∪ {e_c}, k − 1)
        else
            MAF(F_1, F_2, E*, k)
```

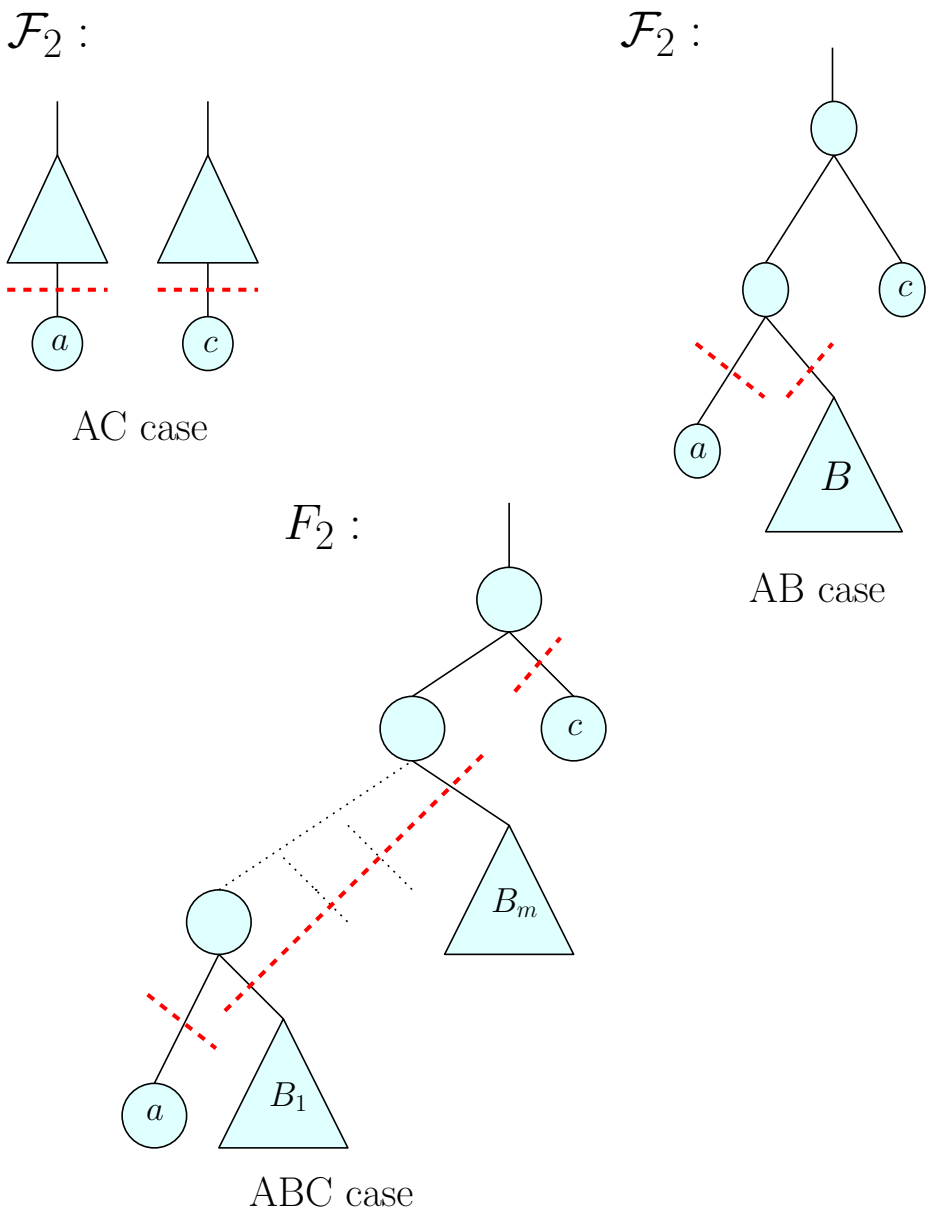


Figure 4.4: The branching rules used in Algorithm 4.2; (a, c) is a cherry in $\mathcal{F}_1$.

Lemma 4.6. *Let a and c be leaves of (T_1, T_2) that form a cherry in T_1. If there is exactly one pendant subtree, B, on the path from a to c in $\mathcal{F}_2$ then there exists an MAF of (T_1, T_2) that cuts e_a if and only if there exists an MAF of (T_1, T_2) that cuts e_c. Moreover, if we let $\mathcal{E}^a$ be the cut edge set of an MAF that cuts e_a and does not isolate c, then $\mathcal{E}^c = \mathcal{E}^a \setminus \{e_a\} \cup \{e_c\}$ is the cut edge set of an MAF that cuts e_c and vice versa.*

Proof. We focus on the case of an MAF that only cuts e_a and does not isolate c, as the claim follows trivially for an MAF that isolates both a and c. Let $\mathcal{E}^a$ be the cut edge set of an MAF, $\mathcal{F}^a$, of (T_1, T_2) such that $e_a \in \mathcal{E}^a$ and $e_c \notin \mathcal{E}^a$. Moreover, assume that $\mathcal{F}^a$ cannot be obtained by cutting both e_a and e_c ($\mathcal{F}^a$ does not isolate c). We claim that $\mathcal{E}^c = (\mathcal{E}^a \setminus e_a) \cup e_c$ is also the cut edge set of an MAF of (T_1, T_2). As we do not isolate c in our initial MAF, we know that $\{c\} \notin \mathcal{F}^a$ and hence c belongs to some non-singleton component of $\mathcal{F}^a$. It is sufficient to show that relabeling c in this component as a, and relabeling the singleton a component as c results in an agreement forest of (T_1, T_2). Clearly, having c as a singleton does not result in any incompatibility with either T_1 or T_2. Thus it remains to show that relabeling c as a is safe.

As (a, c) is a cherry in $\mathcal{F}_1$, this cannot create any incompatibility with T_1, as a and c have identical sets of ancestors in $\mathcal{F}_1$, which is a forest that is obtained from T_1. In $\mathcal{F}_2$ however, B is a pendant subtree on the path from a to c, and hence we must consider the possible incompatibility of triplets with T_2 if we change some triplet, $c|xy$ or $cx|y$, to $a|xy$ or $ax|y$. One of x or y (or both) must be a leaf belonging to the subtree B, otherwise we cannot obtain an MAF as we could have just cut e_B and then contracted the cherry (a, c). If either x or y is not a leaf from B, then this leaf does not result in any incompatibility, as it would be a common ancestor of a, c, and any leaf from B. Thus we need only consider leaves from B that constitute any triplets involving the relabeled leaf a. After cutting e_a, we supress the degree 2 node that remains between B and c, and hence c now either forms a cherry with a leaf from B, or we have a triplet of the form $xy|c$, where $x, y \in \mathcal{L}(B)$. Now assume that we cut e_c instead of e_a. Again, after supressing degree 2 nodes, we have that either a forms a cherry with some leaf from B, or we have a triplet of the form $a|xy$, where $x, y \in \mathcal{L}(B)$. This same cherry or triplet is created if we relabel c as a, and hence we do not create any incompatible triplets. Moreover, we do not create any overlapping components, as a now appears only in the connected component to which c belonged, and c now exists solely as a singleton in our forest. The remainder of the components

are unchanged, and did not have overlap involving a or c to begin with. □

Lemma 4.6 implies that the **AB** case is safe in the context of computing the core MAF. This is true as Lemma 3.7 in [8] shows that we can always obtain an MAF by cutting e_B, and Lemma 4.6 shows that any MAF of (T_1, T_2) that cuts e_a and does not isolate c can be converted into an MAF that instead cuts e_c and does not isolate a. Thus, if we are in the **AB** case and obtain an MAF that cuts e_a, then we know that it is safe to add e_c to the cut edge set for our core MAF, as either there exists an MAF that cuts both e_a and e_c, or we can obtain an MAF by cutting e_c instead of e_a. The safety of the **AC** and **ABC** cases follow immediately from Theorem 4.4. Thus we arrive at the following improvement on our running time bound.

Theorem 4.7. *Let (T_1, T_2) be two phylogenetic trees and let k be the size of an MAF of (T_1, T_2). Then, the core MAF of (T_1, T_2) can be computed in $\mathcal{O}(2.42^k \cdot n)$ time.*

Proof. The correctness of the **AC** case follows directly from Theorem 4.4, while the correctness of the **AB** case follows from a combination of Theorem 4.4 and Lemma 4.6. Thus we need only argue that the **ABC** case is correct. Theorem 4.4 shows that cutting e_a, e_c, and at least one of $e_{B_1}, ..., e_{B_m}$ is safe. It remains then to show that cutting the entire set of edges $\{e_{B_i} : i \in [m]\}$ in the B branch is correct. Indeed, as (a, c) is a cherry in $\mathcal{F}_1$, the only way we can obtain an AF without cutting e_a or e_c is to make (a, c) a cherry in $\mathcal{F}_2$ as well, and the only way to achieve this is to cut each edge e_{B_i}, for $i \in [m]$.

The achieved running time results from the distinction between having multiple pendant subtrees versus a single pendant subtree. The recursive calls made are largely similar to those made in the algorithm on which Theorem 3.1 in [8] is based. Indeed the only difference is that we are required to make an additional call in the **AB** case, as we must find all edge cuts made across every possible MAF, opposed to just computing an MAF. Lemma 4.6 shows that we do not require an additional call that cuts e_c and thus we maintain a branching vector of $(1, 1)$ as in the **AC** case. Hence for both of these cases, we have branching number of 2. Thus it remains only to look at the **ABC** case, which has corresponding branching vector $(1, 1, 2)$ and, as stated in [8], this has a corresponding branching number of $1 + \sqrt{2}$. As before, a more careful implementation can attain the linear time bound for each recursive call, as opposed to the n^2 time required by Algorithm 4.2. Since our search tree still has depth bounded by k, and $1 + \sqrt{2} < 2.42$, the correctness of the stated running time bound follows. □

4.2 Protected Edges

We now introduce the concept of edge protection into our core MAF algorithm. By using this, in combination with the branching rules of Section 4.1 we are able to obtain a significant improvement on our running time for computing the core MAF. This idea was previously used in [9] to obtain a running time bound of $\mathcal{O}(2.42^k \cdot n)$ for the standard MAF problem on multifurcating trees, and in [11] to obtain a $\mathcal{O}(2^k \cdot n)$ algorithm for MAF on binary trees.

First, we introduce one additional branching rule that is necessary for our protection scheme to obtain the desired bound. This rule, which we call the **RAB** case, is presented below in Lemma 4.8 and further illustrated in Figure 4.5.

Lemma 4.8 (**RAB** case)**.** *Let (a, c) be the cherry in $\mathcal{F}_1$ selected by the depth rule and let b_1, b_2, d_1, d_2 be leaves of T_1 and T_2 (b_2 and d_2 may not exist).*

(i) If d_1 is the uncle of (a, c) in $\mathcal{F}_1$ and (c, d_1) is a cherry in $\mathcal{F}_2$, then:

$$\begin{aligned}\textsc{Maf}(\mathcal{F}_1, \mathcal{F}_2, \mathcal{E}^*, \mathcal{P}, k) = \textsc{Maf}(\mathcal{F}_1, \mathcal{F}_2 // e_a, \mathcal{E}^* \cup \{e_a\}, \mathcal{P}, k-1) \\ \cup \textsc{Maf}(\mathcal{F}_1, \mathcal{F}_2 // e_c, \mathcal{E}^* \cup \{e_c, e_{d_1}\}, \mathcal{P}, k-1).\end{aligned}$$

(ii) If (b_1, b_2) is a cherry in $\mathcal{F}_1$ and b_1 and b_2 are the sibling and uncle of a in $\mathcal{F}_2$ respectively, then:

$$\begin{aligned}\textsc{Maf}(\mathcal{F}_1, \mathcal{F}_2, \mathcal{E}^*, \mathcal{P}, k) = \textsc{Maf}(\mathcal{F}_1 \mathcal{F}_2 // e_a, \mathcal{E}^* \cup \{e_a\}, \mathcal{P}, k-1) \\ \cup \textsc{Maf}(\mathcal{F}_1, \mathcal{F}_2 // e_{b_1}, \mathcal{E}^* \cup \{e_{b_1}, e_{b_2}\}, \mathcal{P}, k-1).\end{aligned}$$

(iii) If b_1 is the uncle of (a, c) in $\mathcal{F}_1$ and (a, b_1) is a cherry in $\mathcal{F}_2$, then:

$$\begin{aligned}\textsc{Maf}(\mathcal{F}_1, \mathcal{F}_2, \mathcal{E}^*, \mathcal{P}, k) = \textsc{Maf}(\mathcal{F}_1, \mathcal{F}_2 // e_c, \mathcal{E}^* \cup \{e_c\}, \mathcal{P}, k-1) \\ \cup \textsc{Maf}(\mathcal{F}_1, \mathcal{F}_2 // e_a, \mathcal{E}^* \cup \{e_a, e_{b_1}\}, \mathcal{P}, k-1).\end{aligned}$$

(iv) If (d_1, d_2) is a cherry in $\mathcal{F}_1$ and d_1 and d_2 are the sibling and uncle of c in $\mathcal{F}_2$ respectively, then:

$$\textsc{Maf}(\mathcal{F}_1, \mathcal{F}_2, \mathcal{E}^*, \mathcal{P}, k) = \textsc{Maf}(\mathcal{F}_1, \mathcal{F}_2 // e_c, \mathcal{E}^* \cup \{e_c\}, \mathcal{P}, k-1) \\ \cup \textsc{Maf}(\mathcal{F}_1, \mathcal{F}_2 // e_{d_1}, \mathcal{E}^* \cup \{e_{d_1}, e_{d_2}\}, \mathcal{P}, k-1).$$

Proof. We give the proof for (i) and (ii), the proofs of (iii) and (iv) are analogous.

The correctness of (ii) follows immediately from the correctness of Lemma 4.6. This is because (b_1, b_2) is a cherry in $\mathcal{F}_1$, and a is the only pendant node on the path from b_1 to b_2 in $\mathcal{F}_2$. Here, e_{b_2} is added to $\mathcal{E}^*$ for the same reason that we add e_c to $\mathcal{E}^*$ in the **AB** case.

In the case of (i), we are effectively reversing the roles of $\mathcal{F}_1$ and $\mathcal{F}_2$. That is, we are treating the cherry (c, d_1) in $\mathcal{F}_2$ as the cherry we are branching on, under which circumstance, a becomes the pendant node on the path from c to d_1 in $\mathcal{F}_1$. Thus again, by Lemma 4.6, we know that it is sufficient to make two recursive calls, one which cuts e_a, and one which cuts e_c. Again, adding e_{d_1} to $\mathcal{E}^*$ is analogous to adding e_c to $\mathcal{E}^*$ in the **AB** case. □

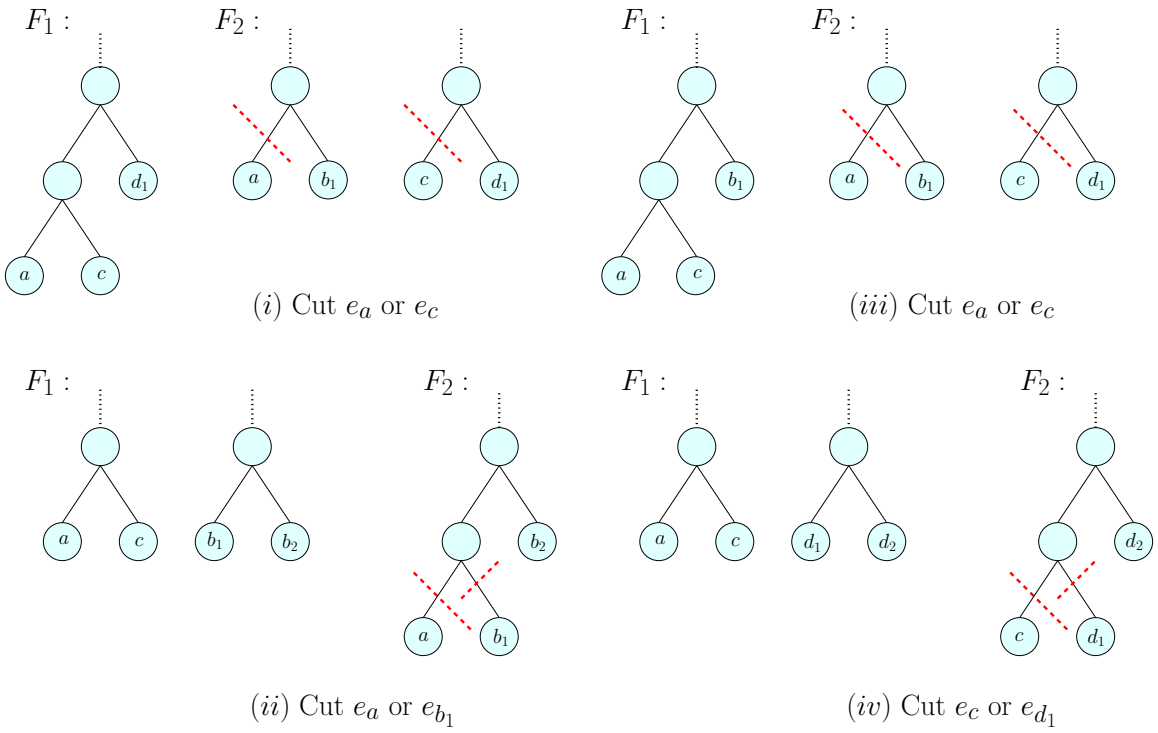


Figure 4.5: The scenarios that result in the application of the **RB** case, and the cuts made in the corresponding recursive calls.

The target of our edge protection strategy here is to improve on the 2.42^k bound that results from case **ABC**. We introduce a new argument of our invocations which we call $\mathcal{P}$ and which is a set that contains all edges which we have protected. For every edge $e_x \in \mathcal{P}$, we prohibit cutting e_x in $\mathcal{F}_2$ and thus do not add e_x to $\mathcal{E}^*$ in any descendant invocation of the recursive call that added e_x to $\mathcal{P}$. Thus, our initial invocation is now $\textsc{Maf}(T_1, T_2, \emptyset, \emptyset, k)$. Whenever we arrive at the **ABC** case, we set $\mathcal{P} = \mathcal{P} \cup \{e_a\}$ in the branch where we cut e_c. Our final algorithm, which utilizes this strategy, is presented in Algorithm 4.3. We now argue the safety of this edge protection scheme.

Lemma 4.9. *Let $\mathcal{E}$ be the cut edge set used in line 8 of Algorithm 4.3 to compute $\mathcal{S}$. For every MAF $\mathcal{F}'$ of (T_1, T_2), if $\mathcal{E}'$ is the cut edge set of $\mathcal{F}'$, then $\mathcal{E}' \subseteq \mathcal{E}$.*

Proof. The correctness of the **AC** and **AB** cases follow immediately from Theorem 4.4 and a combination of Theorem 4.4 and Lemma 4.6 respectively. Furthermore, we demonstrated that the new **RAB** case is correct in Lemma 4.8. Thus it remains to show that the **ABC** case remains correct with the introduction of edge protection. Clearly, both the a and B branches of this case remain unchanged, and hence the focus is on what changes in the c case. Since we protect the edge e_a, it is evident that we will miss any MAFs that cut both e_a and e_c in this branch. However, any MAF that can be obtained by cutting e_c and then cutting e_a can also be obtained by first cutting e_a and then cutting e_c. Such an MAF would be found in the a branch of the **ABC** case, and thus we need not be concerned about missing it in the subtree induced by the c branch. □

We also note that, while it is safe to protect e_a in the B branch, it is unnecessary, as (a, c) becomes a trivial cherry after the invocation corresponding to this particular branch. Finally, we introduce the depth rule, originally proposed in [11]. Just as in [11] this rule is crucial in the construction of our proof that edge protection does in fact yield an improvement on the running time bound for our algorithm.

Depth Rule: If $\mathcal{P} = \emptyset$, then we select our non-trivial cherry (a, c) to branch on such that the depth of a in T_1 is maximum. We break ties based on the leaf which has greater depth in T_2, and if a tie remains then we choose our cherry arbitrarily from those that remain tied. If $\mathcal{P} \neq \emptyset$ then we let z be the parent node of the smallest subtree of $\mathcal{F}_1$ that contains a protected edge in $\mathcal{P}$. The cherry (a, c) is then chosen as before, but with respect to depth in the subtree induced by z.

Algorithm 4.3: Core MAF Algorithm with edge protection and branching rules

Data: Phylogenetic Trees (T_1, T_2)
Result: $\mathcal{S}$, the core MAF for (T_1, T_2)

$\mathcal{E} \leftarrow \emptyset$
$k \leftarrow 0$
$success \leftarrow 0$
do
  $\textsc{Maf}(T_1, T_2, \emptyset, \emptyset, k)$
  $k \leftarrow k + 1$
while $success = 0$;
$\mathcal{S} \leftarrow T_2 // \mathcal{E}$
return $\mathcal{S}$

Function $\texttt{MAF}(\mathcal{F}_1,\ \mathcal{F}_2,\ \mathcal{E}^*,\ \mathcal{P},\ k)$**:**
  if $k < 0$ **then**
    return
  do
    contract (a, c) in $\mathcal{F}_1$ and $\mathcal{F}_2$
  while $\exists$ *a trivial cherry* (a, c);
  for *each singleton* c *of* $\mathcal{F}_2$ **do**
    remove c from $\mathcal{F}_1$ and $\mathcal{F}_2$
  if $\mathcal{F}_1 = \mathcal{F}_2$ **then**
    $\mathcal{E} \leftarrow \mathcal{E} \cup \mathcal{E}^*$
    $success \leftarrow 1$
    return
  else
    if $\exists$ *a non-trivial cherry in* $\mathcal{F}_1$ **then**
      select non-trivial cherry (a, c) according to the depth rule
      case AC **do**
        $\textsc{Maf}(\mathcal{F}_1, \mathcal{F}_2 // e_a, \mathcal{E}^* \cup \{e_a\}, \mathcal{P}, k-1)$
        $\textsc{Maf}(\mathcal{F}_1, \mathcal{F}_2 // e_c, \mathcal{E}^* \cup \{e_c\}, \mathcal{P}, k-1)$
      case AB **do**
        $\textsc{Maf}(\mathcal{F}_1, \mathcal{F}_2 // e_a, \mathcal{E}^* \cup \{e_a, e_c\}, \mathcal{P}, k-1)$
        $\textsc{Maf}(\mathcal{F}_1, \mathcal{F}_2 // e_B, \mathcal{E}^* \cup \{e_B\}, \mathcal{P}, k-1)$
      case ABC **do**
        $\textsc{Maf}(\mathcal{F}_1, \mathcal{F}_2 // e_a, \mathcal{E}^* \cup \{e_a\}, \mathcal{P}, k-1)$
        $\textsc{Maf}(\mathcal{F}_1, \mathcal{F}_2 // \{e_{B_1}, ..., e_{B_m}\}, \mathcal{E}^* \cup \{e_{B_1}, ..., e_{B_m}\}, \mathcal{P}, k-m)$
        $\textsc{Maf}(\mathcal{F}_1, \mathcal{F}_2 // e_c, \mathcal{E}^* \cup \{e_c\}, \mathcal{P} \cup \{e_a\}, k-1)$
      case RAB **do**
        case (i) **do**
          $\textsc{Maf}(\mathcal{F}_1, \mathcal{F}_2 // e_a, \mathcal{E}^* \cup \{e_a\}, \mathcal{P}, k-1)$
          $\textsc{Maf}(\mathcal{F}_1, \mathcal{F}_2 // e_c, \mathcal{E}^* \cup \{e_c, e_{d_1}\}, \mathcal{P}, k-1)$

```
39 case (ii) do
40 |   MAF(F1, F2//e_a, E* ∪ {e_a}, P, k − 1)
41 |   MAF(F1, F2//e_{b1}, E* ∪ {e_{b1}, e_{b2}}, P, k − 1)
42 case (iii) do
43 |   MAF(F1, F2//e_c, E* ∪ {e_c}, P, k − 1)
44 |   MAF(F1, F2//e_a, E* ∪ {e_a, e_{b1}}, P, k − 1)
45 case (iv) do
46 |   MAF(F1, F2//e_c, E* ∪ {e_c}, P, k − 1)
47 |   MAF(F1, F2//e_{d1}, E* ∪ {e_{d1}, e_{d2}}, P, k − 1)
48 else
49 |   MAF(F1, F2, E*, P, k)
```

Theorem 4.10. *Let (T_1, T_2) be two phylogenetic trees and let k be the size of an MAF of (T_1, T_2). Then, the core MAF of (T_1, T_2) can be computed in $\mathcal{O}(2.27^k \cdot n)$ time.*

Proof. To show that the stated running time bound is true, we examine the cases that arise in the c branch of the **ABC** case, as the other cases already yield a branching number of 2 or better, and thus this is where we look to improve on bounding the size of our search tree. Here we have two possibilities that can arise in $\mathcal{F}_1$ after we cut c: either we have a new cherry involving a in $\mathcal{F}_1$, say (a, a'), or a's parent in $\mathcal{F}_1$ has a descendant sibling pair (a', c'). These two cases are illustrated in Figure 4.6.

First, we consider the (a, a') case. Here our algorithm would select (a, a') as the next cherry to recurse on, as the parent node of (a, a') is z in the context of the depth rule, and thus (a, a') is the only cherry belonging to the subtree induced by z. There are two subcases here to consider, either (a, a') is a trivial cherry, or it is not. If (a, a') were a trivial cherry, then subcase (iii) of the **RAB** case would have applied to the cherry (a, c) (here a' would be b_1 in Figure 4.5). Thus, we need only consider the later case. If we now are in either the **AC** or **AB** case, then we have branching vector of $(1, 2, 2)$ as we are either going to cut e_{B_1} (note $a' = B_2$ in this case) in the **AB** case or $e_{a'}$ in the **AC** case, since e_a is protected. These additional cuts are illustrated in Figure 4.7. The second 2 in our branching vector is a result of the fact that these cuts come in addition to having already cut e_c. This branching vector has characteristic polynomial

$$P(\lambda) = \lambda^2 - \lambda - 2. \tag{4.1}$$

Setting $P(\lambda) = 0$ and solving for $\lambda \in \mathbb{R}$ we get

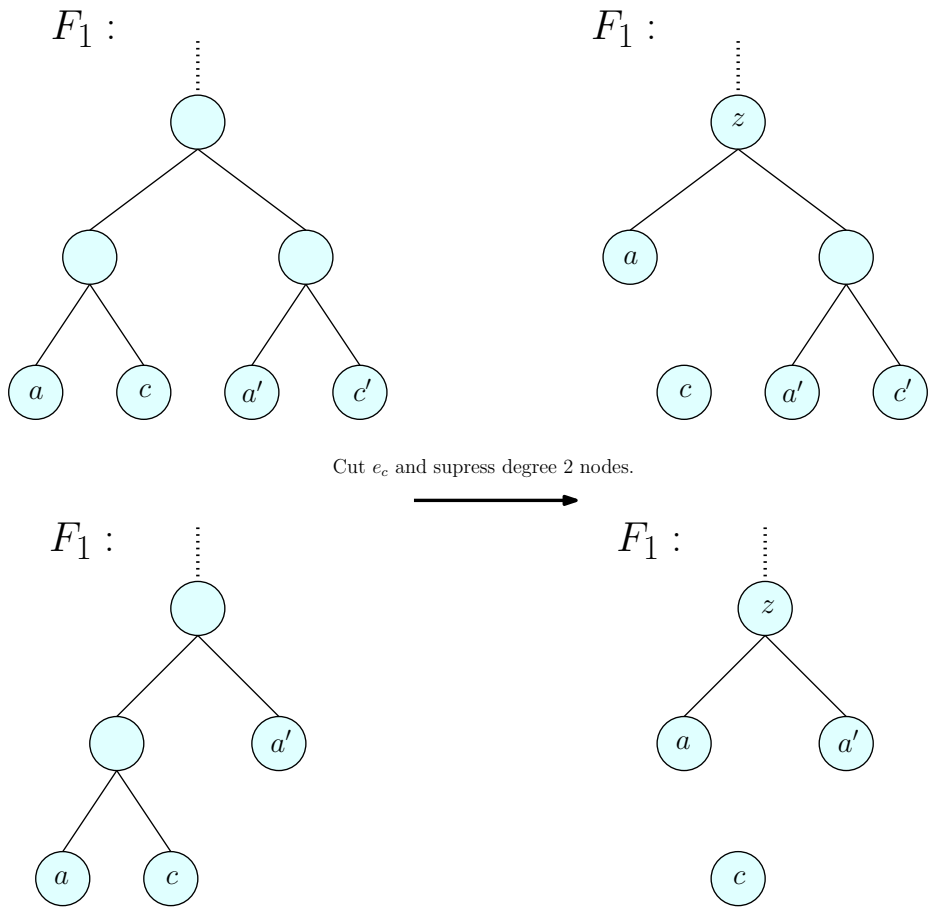


Figure 4.6: The new cherries that arise after cutting e_c.

$$\lambda^2 = \lambda + 2$$
$$\lambda = 2$$

as the unique positive real root of 4.1, and hence a branching number of 2.

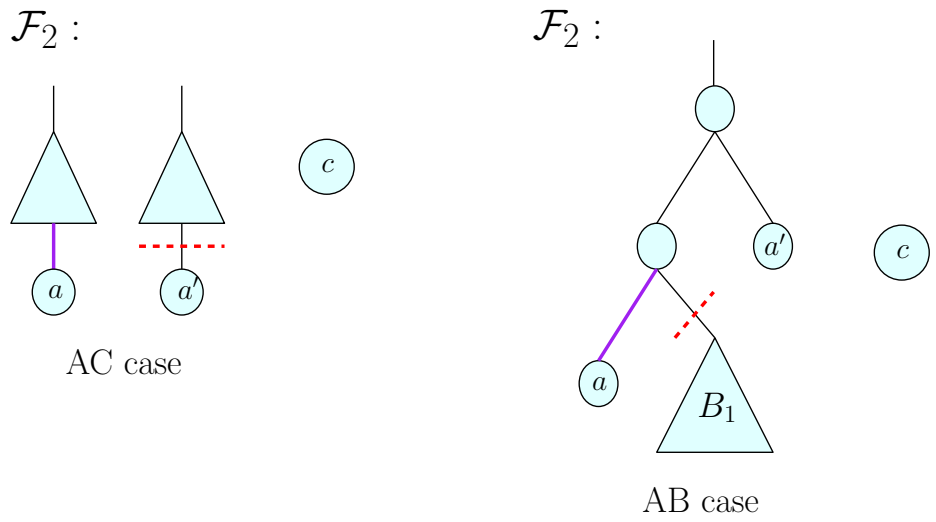


Figure 4.7: The additional cuts that occur if we are in the **AC** case (left) or **AB** case again after cutting e_c and branching on the cherry (a, a'). The purple edge represents that e_a is protected. In both cases we cut one additional edge after cutting e_c.

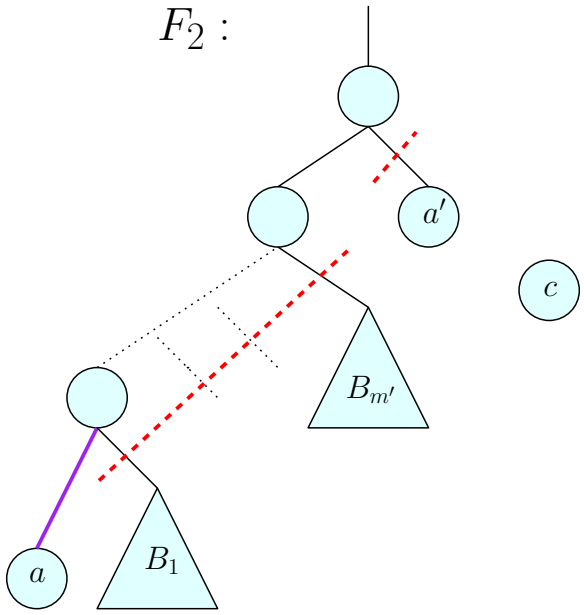
If instead we end up in the **ABC** case again, then we would either cut $e_{a'}$ or $e_{B_1}, ..., e_{B_{m'}}$, where $B_1, ..., B_{m'}$, $2 \geq m' \leq m$ are our pendant subtrees (note $a' = B_i$, for some integer $i \in [3, m]$). Thus in the original c branch we either make 2 cuts (e_c and $e_{a'}$) or at least 3 cuts (e_c and $e_{B_1}, ..., e_{B_{m'}}$). These additional cuts are illustrated in Figure 4.8. Hence our branching vector becomes $(1, 2, 2, 3)$, which has as its characteristic polynomial

$$P(\lambda) = \lambda^3 - \lambda^2 - 2\lambda - 1. \tag{4.2}$$

Setting $P(\lambda) = 0$ and solving for $\lambda \in \mathbb{R}$ we get

$$\begin{aligned}\lambda^3 &= \lambda^2 + 2\lambda + 1 \\ \lambda &= \frac{1}{3}\left(1 + \sqrt[3]{\frac{47 - 3\sqrt{93}}{2}} + \sqrt[3]{\frac{1}{2}(47 + 3\sqrt{93})}\right) \\ &< 2.15\end{aligned}$$

as the unique positive real root of 4.2. Thus our branching number for the cherry (a, a') subcase is less than 2.15.



ABC case

Figure 4.8: The additional cuts that occur if we are in the **ABC** case after cutting e_c and branching on the cherry (a, a'). The purple edge represents that e_a is protected. In this case we either cut an additional m' edges (where $m' \geq 2$) or we cut a single additional edge.

In the case where we branch on the cherry (a', c'), we can again look at each of our branching rule cases. If (a', c') puts us in either the **AB** or **RAB** case, then we make one cut and one recursive call, thus obtaining a branching vector of $(1, 2, 2)$ overall, which by 4.1 we know has a corresponding branching number of 2. If instead we are in the **AC** case, then we either cut $e_{a'}$ or $e_{c'}$ in addition to e_c. In the case where we cut $e_{c'}$ it is beneficial to protect $e_{a'}$ as we now have (a, a') as a cherry, which as we saw before, cannot be trivial. The **AC** case involving the cherry (a', c') is shown in

Figure 4.9. Since both a and a' are protected, and we know that (a, a') is not trivial, we get at least one additional cut regardless of which case we are in. Thus we have an effective branching vector of $(1, 2, 2, 3)$, which by 4.2 we know has a corresponding branching number less than 2.15.

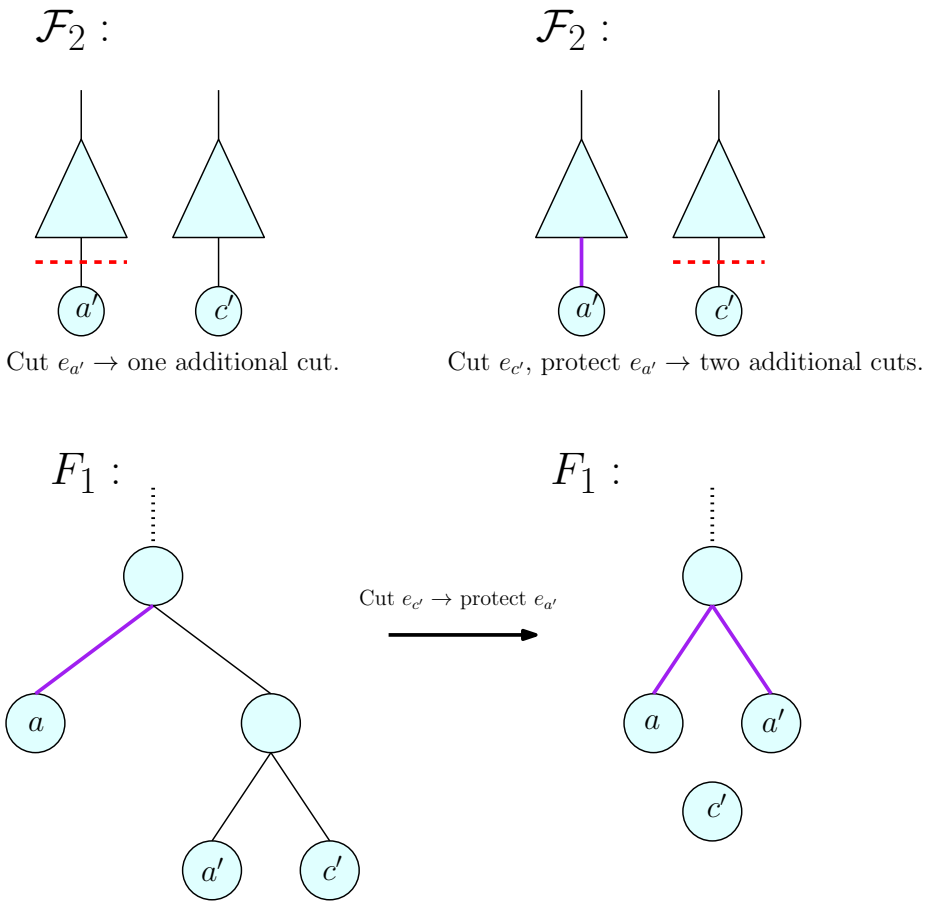


Figure 4.9: The additional cuts that occur if we are in the **AC** case after cutting e_c, protecting e_a, and branching on the cherry (a', c') (top). In the case where we cut $e_{c'}$ we protect $e_{a'}$, resulting in the non-trivial cherry (a, a') becoming protected (bottom).

Finally, we consider what happens if we again are in the **ABC** case when we branch on the cherry (a', c'). In addition to cutting e_c we also cut either $e_{a'}$, $e_{B'_1}, ..., e_{B'_{m'}}$, or $e_{c'}$. As before, if we cut $e_{c'}$, then we also protect $e_{a'}$ and we again end up having (a, a') as a cherry, where both are protected and the cherry cannot be trivial (see bottom of Figure 4.9). Thus, as in the **AC** case, we are guaranteed at least one free cut. The **ABC** case involving the cherry (a', c') is shown in Figure 4.10.

Hence the branching vector we obtain is $(1, 2, 2, 3, 3)$ which has characteristic polynomial

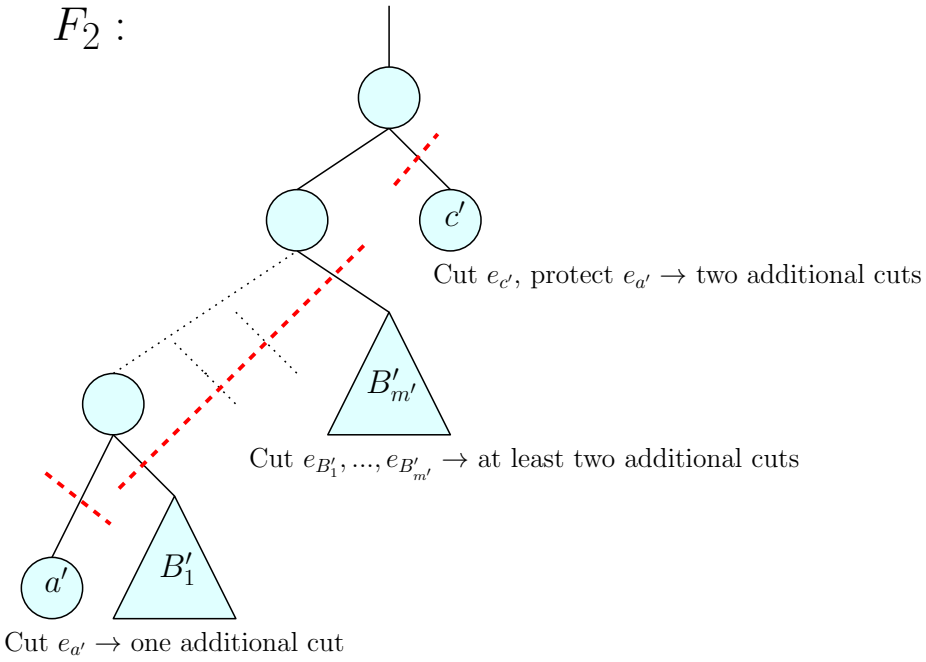


Figure 4.10: The additional cuts that occur if we are in the **ABC** case after cutting e_c and branching on the cherry (a', c').

$$P(\lambda) = \lambda^3 - \lambda^2 - 2\lambda - 2. \tag{4.3}$$

Setting $P(\lambda) = 0$ and solving for $\lambda \in \mathbb{R}$ we get

$$\begin{aligned} \lambda^3 &= \lambda^2 + 2\lambda + 2 \\ \lambda &= \frac{1}{3}\left(1 + \sqrt[3]{37 - 3\sqrt{114}} + \sqrt[3]{37 + 3\sqrt{114}}\right) \\ &< 2.27 \end{aligned}$$

as the unique positive real root of 4.3. Thus our branching number for the cherry (a', c') subcase is less than 2.27. Since our search tree as depth of at most k and each recursive call takes time linear in n (under the aforementioned careful implementation), the proof of the theorem follows.

□

Chapter 5

Conclusions

In this thesis we have presented a new problem in the realm of phylogenetics, that of the core MAF. The inspiration came about as an attempt at developing a method by which to effectively sample from the space of all MAFs for a given set of phylogenetic trees. To this end, we have formulated a structure which shows us the components that are preserved across all MAFs of our input trees. As a first step, we focused our efforts on developing efficient algorithms for solving the problem solely as it pertains to rooted binary phylogenetic trees.

We have demonstrated that the exceptionally useful technique of cluster partitioning is safe in the context of the core MAF algorithm. We have also adapted existing branching algorithms to obtain a provably efficient algorithm for computing the core MAF of two rooted binary phylogenetic trees. Coming the results of Chapters 3 and 4 we can conclude that, given an optimal cluster partition of input trees (T_1, T_2), we can, in $\mathcal{O}(2.27^{k'} \cdot \text{poly}(n))$ time, compute the core MAF of (T_1, T_2); where n is the size of our input trees, and k' is the size of the largest MAF across all of our cluster MAFs. As all of the steps involved in computing the cluster partition and merging the results appear to be implementable in linear time, we believe that a running time of $\mathcal{O}(2.27^{k'} \cdot n)$ is obtainable. However, as we have not yet implemented the algorithms presented, we cannot say this with certainty.

While it cannot be immediately inferred that the methods presented here apply more generally to other classes of phylogenetic trees, we have nonetheless laid the groundwork from which additional core MAF algorithms may be designed and implemented.

5.1 Applications and Future Work

The core MAF of a pair of phylogenetic trees tells us what taxa will be conserved when attempting to reconcile the two trees and identify LGT events. However, what applications this might have in the realm of bioinformatics remains to be seen. To this end, an implementation of Algorithm 4.3 along with cluster partitioning is crucial. The algorithm can then be applied to real world examples, and from these, the core

MAF may provide additional insight into reconciling the input trees, ideally something that traditional MAFs do not provide.

It is also important to address more general types of trees, and sets of more than two input trees. Algorithms that can compute the core MAF for unrooted trees and for multifurcating trees would be of prime interest. Moreover, algorithms for a less strict version of the core MAF, that is the set of components preserved across at least $X\%$ of trees, would also likely provide greater insight. This version of the core MAF problem would provide some measure of confidence in the LGT events that are observed. It remains to be determined if the current methods remains applicable to this problem variant.

Another variant of the core MAF problem worth considering is applying this idea to maximal agreement forests and $k + x$, $x \geq 1$, agreement forests. A final area of interest would be in determining what edge cuts are made in every MAF. Whether the techniques used to compute the core MAF are capable at addressing this problem, would also be a potentially useful area of future work.

Bibliography

[1] B. L. Allen and M. Steel. Subtree transfer operations and their induced metrics on evolutionary trees. *Annals of Combinatorics*, 5(1):1–15, Jun 2001.

[2] M. Bordewich and C. Semple. On the computational complexity of the rooted subtree prune and regraft distance. *Annals of combinatorics*, 8(4):409–423, 2005.

[3] M. Cygan, F. V. Fomin, Ł. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, and S. Saurabh. *Parameterized algorithms*, volume 4. Springer, 2015.

[4] J. Hein, T. Jiang, L. Wang, and K. Zhang. On the complexity of comparing evolutionary trees. *Discrete Applied Mathematics*, 71(1-3):153–169, 1996.

[5] Z. Li and N. Zeh. Computing maximum agreement forests without cluster partitioning is folly. In *25th Annual European Symposium on Algorithms (ESA 2017)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.

[6] S. Linz and C. Semple. A cluster reduction for computing the subtree distance between phylogenies. *Annals of combinatorics*, 15(3):465, 2011.

[7] C. Semple, M. Steel, and B. Steel. *Phylogenetics*. Oxford lecture series in mathematics and its applications. Oxford University Press, 2003.

[8] C. Whidden, R. G. Beiko, and N. Zeh. Fixed-parameter algorithms for maximum agreement forests. *SIAM Journal on Computing*, 42(4):1431–1466, 2013.

[9] C. Whidden, R. G. Beiko, and N. Zeh. Fixed-parameter and approximation algorithms for maximum agreement forests of multifurcating trees. *Algorithmica*, 74(3):1019–1054, 2016.

[10] C. Whidden and N. Zeh. A unifying view on approximation and FPT of agreement forests. In *International Workshop on Algorithms in Bioinformatics*, pages 390–402. Springer, 2009.

[11] C. Whidden and N. Zeh. Computing the SPR Distance of Binary Rooted Trees in $O(2^k n)$ time. 2020.

[12] C. Whidden, N. Zeh, and R. G. Beiko. Supertrees Based on the Subtree Prune-and-Regraft Distance. *Systematic Biology*, 63(4):566–581, 04 2014.