

Minimum-Flip Supertrees: Complexity and Algorithms

Duhong Chen, Oliver Eulenstein, David Fernández-Baca, and Michael Sanderson

Abstract—The input to a supertree problem is a collection of phylogenetic trees that intersect pairwise in their leaf sets; the goal is to construct a single tree that retains as much as possible of the information in the input. This task is complicated by inconsistencies due to errors. We consider the case where the input trees are rooted and are represented by the clusters they exhibit. The problem is to find the minimum number of *flips* needed to resolve all inconsistencies, where each flip moves a taxon into or out of a cluster. We prove that the minimum-flip problem is \mathcal{NP} -complete, but show that it is fixed-parameter tractable and give approximation algorithms for special cases.

Index Terms—Phylogenetic tree, supertree, tree assembly, NP-completeness.



1 INTRODUCTION

CONSTRUCTION of the phylogenetic tree of life for all 1.7 million species on earth has recently emerged as a “grand challenge” in the biological sciences. Conventionally, phylogenetic trees are reconstructed by solving certain optimization problems (e.g., maximum parsimony, maximum likelihood) that use a set of homologous biomolecular sequences assembled from all species in the study. Because the conventional algorithms to solve these problems are unlikely to be able to handle data at the scale of the whole tree of life, divide-and-conquer approaches seem interesting [1]. A relatively new approach, known as *phylogenetic supertree construction*, has received the most attention from biologists, culminating in a recent compendium volume [2]. Supertree construction involves combining the information from a collection of rooted *input trees* to construct a larger rooted tree, a *supertree*, that includes all or most of the taxa from the input trees while preserving the phylogenetic information from those trees [3], [4], [5], [6].¹ Supertrees make statements about relationships not possible from any single input tree alone. Although biologists have been constructing such supertrees informally for many years [8], [9], the development of formal supertree methods has permitted rigorous and unprecedented analyses of large phylogenetic groups in the last few years [10], [11], [12], [13], [14].

In addition, many biologists interested in the ecology or evolution of traits have been drawn to supertree construction because of its power to integrate their sample of data

with information from disparate phylogenetic trees. For example, a recent study of genome size evolution in pines [15] began with data on genome sizes in eight species, for which no single existing phylogeny was available, and constructed a supertree from 20 published phylogenies. This permitted the authors to place their genome size data into a single tree in its proper phylogenetic context.

In practice, the main obstacle to fast and accurate supertree construction is that input trees invariably conflict with one another, that is, errors are present. In conventional phylogenetic methods, which infer a tree from a collection of discrete character data like DNA sequences, conflict between characters can be interpreted as error in scoring of those characters or as “real” evolutionary events that confuse inference, such as multiple gains and losses of the same character state. Both of these are generally referred to as “homoplasy” and some inference methods, maximum parsimony in particular, try to directly minimize this homoplasy. The rationale is that the assumption of unnecessary extra evolutionary changes in characters beyond those required to explain the data is ad hoc.

In the supertree problem, conflicts between input trees do not have this nice evolutionary interpretation, but they still reflect error. In this paper, we study a method to construct supertrees that addresses error directly, through a notion of *error correction*, rather than by the analogy that parsimony inference has provided in the justification for other supertree methods, such as matrix representation parsimony (MRP). Our approach starts from the same matrix representation of the input as is used in MRP, but it proceeds with a different rationale. The matrix representation treats the input as a collection of incomplete binary characters in which each character represents a cluster (clade) from one of the input trees [10], [16], [17], [18]. Taxa present in the cluster are scored 1, those absent are scored 0, and those not sampled on that input tree are scored by a ?. One notion of *error* is the presence of an incorrect taxon in a cluster or the absence of one that should be present. Thus, errors correspond to *flips* from $0 \rightarrow 1$ or $1 \rightarrow 0$; these flips prevent the matrix from representing any phylogenetic tree

1. It is shown in [7] that no *reasonable* supertree method exists for when the input trees are unrooted. Consequently, in this paper, we will restrict our attention to rooted trees.

- D. Chen, O. Eulenstein, and D. Fernández-Baca are with the Department of Computer Science, Iowa State University, Ames, IA 50011-1040. E-mail: {jackie, oeulens, fernande}@iastate.edu.
- M. Sanderson is with the Section of Evolution and Ecology, University of California, Davis, CA 95616. E-mail: mjsanderson@ucdavis.edu.

Manuscript received 13 July 2004; revised 18 Apr. 2005; accepted 22 July 2005; published online 1 May 2006.

For information on obtaining reprints of this article, please send e-mail to: tcb@computer.org, and reference IEEECS Log Number TCBB-0078-0704.

perfectly. A natural optimization problem is to find the minimum number of flips that converts the matrix into one that does represent a phylogenetic tree; we call this the *minimum-flip problem* and the resulting trees *minimum-flip supertrees*.

We show that the minimum-flip problem is \mathcal{NP} -complete, even when restrictions are placed, such as allowing flips in just one direction or only permitting input trees with the same taxon set. On the positive side, we prove that, when the input trees have identical taxon sets, the problem is fixed-parameter tractable, fixed-ratio approximable, and that an approximation scheme exists if the objective is to maximize the *similarity* between the clusters in the input tree and the clusters of the supertree. In computational studies published elsewhere [19], it is shown that minimum-flip supertrees tend to be more accurate than those produced by other known methods, particularly in the situation where supertrees are most relevant: when the overlap between source trees is small.

1.1 Related Work

Most versions of the supertree problem are *consensus tree problems*, which apply only to phylogenies with the same taxon set [20], [21], [22]. Only a few methods apply to phylogenies that do not necessarily share all taxa. In the absence of incompatibility among the input trees, a supertree can be built in polynomial time using the algorithms of Aho et al. [23] and Henzinger et al. [24]. Semple and Steel [6] modified this algorithm to handle incompatible input. Their procedure, the *MinCutSupertree algorithm* (MC), resolves conflicts that prevent the Aho et al. algorithm from proceeding by deleting a minimum amount of information. The approach is thus guided by a local optimization criterion, although some global properties can be shown for the output tree. Recently, a version of MC was proposed [25] that retains more of the uncontradicted information of the input trees and still runs in polynomial time.

For biologists, the most popular supertree method is *matrix representation using parsimony* (MRP), which seeks the most parsimonious tree(s) for the matrix representation of the input trees [10], [16], [17], [18]. While computing a most parsimonious tree is \mathcal{NP} -complete [26], the wide availability of reasonably fast maximum-parsimony software [27] has made MRP the clear choice of phylogeneticists working with real data. However, as mentioned above, although MRP is operationally similar to maximum parsimony, it cannot rest on the same kind of rationale in terms of minimizing homoplasy. Homoplasy on a supertree derived via MRP represents conflicts between clades rather than between individual evolutionarily novel character states and, therefore, although popular, MRP is, in no strong sense, any better justified than other supertree methods.

Another relative of the minimum-flip problem is the *fractional character compatibility problem* (FCC) of Kearney et al. [21]. Given a set of partitions of a taxon set, FCC aims to find an unrooted tree T such that the total *similarity* between the partitions and T is maximized. Although FCC was not intended to be used as supertree construction method, it can be viewed as the flip problem for unrooted phylogenies over the same taxon set. Kearney et al. have shown that FCC is \mathcal{NP} -complete, but have given an approximation scheme for a special case.

1.2 Organization of the Paper

Section 2 provides basic definitions and notation. In particular, flipping is introduced in set and graph-theoretic terms. The complexity of flipping is studied in Section 3. Section 4 presents a fixed-ratio approximation algorithm for the case where there is a bound on the number of taxa in a cluster. Also in that section is an approximation scheme for a similarity-based variant of the minimum-flip problem. Section 5 shows that the minimum-flip problem is solvable in polynomial time when the number of allowed flips is bounded by a constant. It should be noted that, with the exception of the approximation scheme under similarity-based scoring, the results in Sections 4 and 5 apply only in the case where all trees have the same set of taxa. Section 6 concludes the paper with comments on experimental results and open problems.

2 PRELIMINARIES

This section has three parts. In the first, we introduce the basic concepts used throughout the paper, including the notions of character compatibility and flipping. In the second, we discuss the minimum-flip problem as a supertree construction method. In the third part, we present a graph-theoretic view of compatibility and flipping that is useful for both \mathcal{NP} -completeness proofs and algorithms.

2.1 Basic Definitions and Notation

Throughout the paper, M denotes a set of n taxa. For a rooted tree T , $\mathcal{L}(T)$ denotes the leaf set of T .

Definition 2.1 (Phylogenetic trees and clusters). A phylogenetic tree, or phylogeny for short, over set $M' \subseteq M$ is a rooted tree T such that $\mathcal{L}(T) = M'$. The set of all leaves that descend from the same nonroot node of T is a cluster in T ; $\mathcal{L}(T)$ is itself a cluster, called the trivial cluster.

Definition 2.2 (Characters). A (directed binary) character is a pair $C = (N, O)$, where $N \subseteq M$ and $O \subseteq N$. Set N is the taxon set of C , O is the 1-state of C , and $N - O$ is the 0-state of C . C is complete if $N = M$. A completion of C is a complete character $C' = (M, O')$ such that $O \subseteq O'$ and $N - O \subseteq M - O'$ (that is, the 1-state and the 0-state of C are contained in the 1-state and 0-state of C' , respectively).

From this point forward, $\mathcal{C} = (C_1, C_2, \dots, C_r)$ denotes a tuple of characters where $C_i = (N_i, O_i)$ for $i \in \{1, \dots, r\}$. Similarly, $\mathcal{C}' = (C'_1, \dots, C'_r)$ denotes a character tuple (usually derived from \mathcal{C}) where $C'_i = (N'_i, O'_i)$.

Definition 2.3 (Consistency and compatibility). Let T be a phylogeny and $C = (N, O)$ be a character. C is consistent with T if and only if T has a cluster X such that $X \cap N = O$. Character tuple \mathcal{C} is compatible if and only if there exists a phylogeny T consistent with C_i for each $i \in \{1, \dots, r\}$.

The compatibility of a set of characters can be tested in polynomial time [23], [28]. For complete characters, the following result is well-known [29], [30].

Theorem 2.1. Character tuple \mathcal{C} is compatible if and only if $O_i \cap O_j \in \{\emptyset, O_i, O_j\}$ for every $i, j \in \{1, \dots, r\}$.

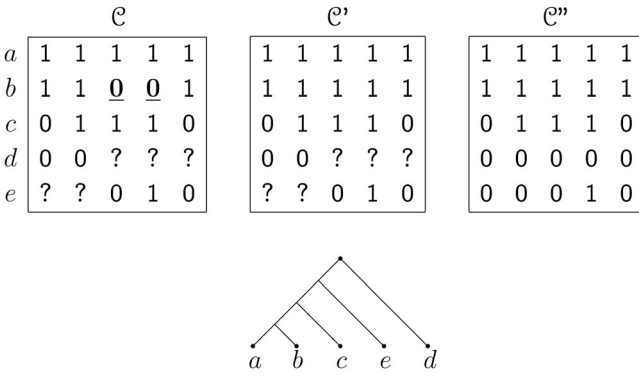


Fig. 1. Top, from left to right: A tuple \mathcal{C} of five incompatible characters on taxon set $M = \{a, b, c, d, e\}$, given by its matrix representation. \mathcal{C}' is the compatible matrix that results from flipping the underlined entries of \mathcal{C} ; the associated flip tuple is $\mathcal{F} = (\emptyset, \emptyset, \{b\}, \{b\}, \emptyset)$. \mathcal{C}'' is a completion of \mathcal{C}' . Bottom: The tree corresponding to \mathcal{C}'' .

Let $A\Delta B$ denote the symmetric difference between the sets A and B , that is, $A\Delta B = (A - B) \cup (B - A)$.

Definition 2.4 (Flipping). The flip operator \triangleleft takes two operands, a character $C = (N, O)$ and a set $F \subseteq N$; $C \triangleleft F$ is the character $C' = (N', O')$, where $N' = N$ and $O' = O \Delta F$. Set F is called a flip set for character C or simply a flip set when C is clear from the context. Flip set F is called a deletion flip or d-flip if $F \subseteq O$ and an insertion flip or i-flip if $F \subseteq N - O$. The flip operator extends to character tuples \mathcal{C} and tuples $\mathcal{F} = (F_1, \dots, F_r)$ such that, for each $i \in \{1, \dots, r\}$, F_i is a flip set for character C_i , as follows: $\mathcal{C} \triangleleft \mathcal{F} = (C_1 \triangleleft F_1, \dots, C_n \triangleleft F_r)$. \mathcal{F} is called a flip tuple for \mathcal{C} . If all flip sets in \mathcal{F} are either d-flips or i-flips, then \mathcal{F} is a d-flip tuple or i-flip tuple, respectively. The size of \mathcal{F} is $s(\mathcal{F}) = \sum_{i=1}^r |F_i|$.

Thus, a d-flip for a character only removes taxa from the 1-state, while an i-flip only adds taxa to the 1-state.

Definition 2.5 (Flip problems). The minimum-flip problem is, given a character tuple \mathcal{C} , find a minimum-size flip tuple \mathcal{F} such that $\mathcal{C} \triangleleft \mathcal{F}$ is compatible. We define three variants of the problem:

- FP is the decision version of the minimum-flip problem, where one is additionally given an integer k and the question is whether there exists a flip tuple \mathcal{F} where $s(\mathcal{F}) \leq k$ such that $\mathcal{C} \triangleleft \mathcal{F}$ is compatible.
- DFP, the d-flip problem, is the version of FP where the flip tuple \mathcal{F} is required to be a d-flip tuple.
- IFP, the i-flip problem, is the version of FP where \mathcal{F} is required to be an i-flip tuple.

In Fig. 1, we illustrate some of the preceding definitions using a matrix representation of the characters, explained next. Assume that $M = \{m_1, \dots, m_n\}$.

Definition 2.6 (Matrix representation of character tuples). The matrix representation of character tuple \mathcal{C} is the $n \times r$ matrix $A = [a_{ij}]$, where

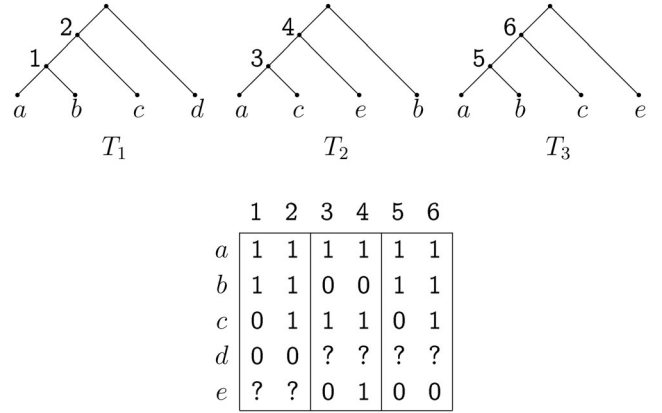


Fig. 2. Top: A collection of trees $\mathcal{T} = \{T_1, T_2, T_3\}$. Bottom: A matrix representation of \mathcal{T} ; each column is labeled by the cluster to which it corresponds.

$$a_{ij} = \begin{cases} 1 & \text{if } m_i \in O_j \\ 0 & \text{if } m_i \in N_j - O_j \\ ? & \text{otherwise.} \end{cases}$$

Let $\mathcal{F} = (F_1, \dots, F_r)$ be a flip tuple for character tuple \mathcal{C} , and let $\mathcal{C}' = \mathcal{C} \triangleleft \mathcal{F}$. Then, the matrix representation of \mathcal{C}' is a matrix $A' = [a'_{ij}]$, where

$$a'_{ij} = \begin{cases} 0 & \text{if } a_{ij} = 1 \text{ and } m_i \in F_j \\ 1 & \text{if } a_{ij} = 0 \text{ and } m_i \in F_j \\ a_{ij} & \text{otherwise.} \end{cases}$$

Thus, the flip operator indeed flips 1s to 0s and 0s to 1s.

2.2 Supertrees and Flipping

We can express a class of supertree problems, including finding minimum-flip supertrees, as optimization problems.

Definition 2.7 (Representations of tree tuples). Let $\mathcal{T} = (T_i)_{i=1}^l$ be a tree tuple such that $\bigcup_{i=1}^l \mathcal{L}(T_i) = M$. For each $i \in \{1, \dots, l\}$, let X_{i1}, \dots, X_{ip_i} denote the nontrivial clusters of T_i , indexed arbitrarily. A character representation of \mathcal{T} is a character tuple

$$\mathcal{C} = (C_{11}, \dots, C_{1p_1}, \dots, C_{i1}, \dots, C_{ip_i}, \dots, C_{l1}, \dots, C_{lp_l}),$$

where, for each $i \in \{1, \dots, l\}$, $j \in \{1, \dots, p_i\}$, $C_{ij} = (N_{ij}, O_{ij})$, where $N_{ij} = \mathcal{L}(T_i)$ and $O_{ij} = X_{ij}$. The matrix representation of \mathcal{C} is a matrix representation of \mathcal{T} .

Fig. 2 illustrates the preceding definitions.

Definition 2.8 (Supertree problem). Let \mathcal{T} be a tuple of phylogenies whose leaf sets union to M , and D be a function that measures the fit of \mathcal{T} to a phylogeny T over M . The supertree problem is to find a phylogeny T over M such that $D(\mathcal{T}, T)$ is minimum.

A function D as in the definition above is called a *supertree measure*. Different such measures can be defined. For example, in MRP, the distance $D(\mathcal{T}, T)$ is the parsimony score of T [31], [32], assuming that the taxa are represented by the rows of the matrix representation of \mathcal{T} . We now introduce another distance measure based on the notion of flipping.

Definition 2.9 (Flip distance). The flip distance from character tuple \mathcal{C} to a phylogeny T over M , denoted $\text{dist}(\mathcal{C}, T)$, is the minimum value of $s(\mathcal{F})$ over all flip tuples \mathcal{F} for \mathcal{C} such that every character in $\mathcal{C} \triangleleft \mathcal{F}$ is consistent with T .

Definition 2.10 (Minimum-flip supertrees). Let \mathcal{T} be a tuple of phylogenies whose leaf sets union to M and T be a phylogeny over M . The flip distance from \mathcal{T} to T is $\text{dist}(\mathcal{T}, T) = \text{dist}(\mathcal{C}, T)$, where \mathcal{C} is a character representation of \mathcal{T} . A minimum-flip supertree for \mathcal{T} is a phylogeny T over M that minimizes $\text{dist}(\mathcal{T}, T)$.

Notice that any character representation \mathcal{C} of \mathcal{T} can be used to define dist since flip distance is independent of the order of the characters in a tuple.

2.3 A Graph-Theoretic View of Flipping

Here, we establish a connection between the minimum-flip problem on complete characters and a class of edge modification problems on bipartite graphs.

Definition 2.11 (Bipartite edge modification problems). A graph property is a set of graphs Γ . Graph G is a Γ -graph if $G \in \Gamma$. Let Γ be a graph property, $G = (X, Y, E)$ be a bipartite graph, and k an integer. Let $A = \{\{x, y\} \mid x \in X, y \in Y\}$. The Γ -edit problem, $\text{EP}(\Gamma)$, is to determine whether there exists a set $F \subseteq A$, where $|F| \leq k$, such that the graph $G' = (X, Y, E \Delta F)$ is in Γ . We also have the following two restricted versions of the problem:

- $\text{DP}(\Gamma)$, the Γ -deletion problem, is the version of the Γ -edit problem where we require that $F \subseteq E$.
- $\text{IP}(\Gamma)$, the Γ -insertion problem, is the version of the Γ -edit problem where we require that $F \subseteq (A - E)$.

Thus, in the edit problem, we are allowed to insert and delete edges, while, in the insertion (deletion) problem, we can only insert (delete) edges.

We now return to character compatibility and flipping. A collection of complete characters can be described by a bipartite graph [28].

Definition 2.12 (Character graph). The character graph for a complete character tuple \mathcal{C} is the bipartite graph $G = (X, Y, E)$, where $X = \{1, \dots, r\}$, $Y = M$, and $E = \{\{x, y\} \mid x \in X, y \in O_x\}$.

The set-theoretic condition for compatibility of complete characters of Theorem 2.1 is expressed graph-theoretically by the M -free property in character graphs.

Definition 2.13 (M -free graph property). A M -graph is a cycle-free path of length 4. The M -free property is the set of all bipartite graphs $G = (X, Y, E)$ that do not have an induced M -graph whose degree-one nodes are in Y .

Fig. 3 illustrates the above notions.

Theorem 2.2.

1. A tuple \mathcal{C} of complete characters is compatible if and only if its character graph is M -free.
2. For complete characters the following are polynomially-equivalent pairs of problems: a) The M -free edit

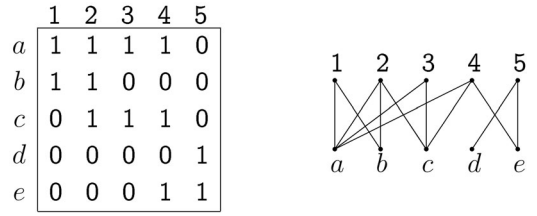


Fig. 3. Left: A tuple of incompatible complete characters. Right: The corresponding character graph; the latter contains the induced M -graph defined by the path $(b, 2, c, 4, e)$.

problem and FP, b) the M -free deletion problem and DFP, and 3) the M -free insertion problem and IFP.

Proof. Part 1 is proven in [28]. For 2, note that there exist polynomial-time transformations from character tuples to character graphs and vice versa. Given part 1, it suffices to observe that a taxon flip in a character is equivalent to an edge deletion or insertion in the character graph and vice versa. \square

3 THE COMPLEXITY OF FLIPPING

In this section, we show that FP, DFP, and IFP are \mathcal{NP} -complete. All characters considered here are assumed to be complete. The \mathcal{NP} -completeness for general characters follows directly.

3.1 FP and DFP are \mathcal{NP} -Complete

Our \mathcal{NP} -completeness proofs rely on two technical lemmas.

Lemma 3.1. Let $\mathcal{C} = (C_1, C_2)$ be a pair of characters such that $|O_1| = |O_2| = 3$ and $|O_1 \cap O_2| \leq 1$. Suppose that $\mathcal{F} = (F_1, F_2)$ is a flip tuple such that $|F_1| = 0$, $|F_2| = 1$, and $\mathcal{C} \triangleleft \mathcal{F} = (C'_1, C'_2)$ is compatible. Then, $O'_1 \cap O'_2 = \emptyset$.

Proof. Let $d(A, B) = |A \Delta B|$ for $A \subseteq M$ and $B \subseteq M$. The function d fulfills the triangle inequality from which it follows that $d(O_1, O'_2) \geq d(O_1, O_2) - d(O'_2, O_2)$. From $|O'_2| = |O_2 \Delta F_2|$ and $|O_2| = 3$, it follows that $|O'_2| \in \{2, 4\}$. Thus, $d(O_1, O_2) \in \{4, 6\}$ and $d(O'_2, O_2) = 1$. Hence, $d(O_1, O'_2) \geq d(O_1, O_2) - d(O'_2, O_2) \geq 3$.

Now, since (C'_1, C'_2) is compatible, we have $O_1 \cap O'_2 \in \{\emptyset, O_1, O'_2\}$ by Theorem 2.1. In case $O_1 \cap O'_2 = O_1$, it follows from $d(O_1, O'_2) \geq 3$ and $|O_1| = 3$ that $|O'_2| \geq 6$, which contradicts $|O'_2| \in \{2, 4\}$. Suppose $O_1 \cap O'_2 = O'_2$. Then, it follows from $d(O_1, O'_2) \geq 3$ and $|O'_2| \in \{2, 4\}$ that $|O_1| \in \{5, 7\}$, which contradicts $|O_1| = 3$. Thus, $O_1 \cap O'_2 = \emptyset$. \square

Lemma 3.2. Let \mathcal{C} be a tuple of complete characters such that $|O_i| = 3$ and $|O_i \cap O_j| \leq 1$, for any $i, j \in \{1, \dots, r\}$, where $i \neq j$. Suppose that there exists a flip tuple $\mathcal{F} = (F_1, \dots, F_r)$ such that $|F_i| = 1$ for each $i \in \{1, \dots, r\}$ and $\mathcal{C}' = \mathcal{C} \triangleleft \mathcal{F}$ is compatible. Then, $|\bigcup_{i=1}^r O'_i| \geq 2r$.

Proof. A flip in \mathcal{F} is either a d-flip or an i-flip since $|F_i| = 1$ for any $i \in \{1, \dots, r\}$. Without loss of generality, we assume that $\mathcal{F}_d = (F_1, \dots, F_l)$ consists only of d-flips and $\mathcal{F}_i = (F_{l+1}, \dots, F_r)$ only of i-flips, for some $l \in \{0, \dots, r\}$. Thus,

$$|O'_i| = 2 \text{ for } i \leq l \text{ and } |O'_i| = 4 \text{ for } i > l. \quad (1)$$

Claim 1. $O'_i \cap O'_j = \emptyset$ when $i, j \in \{1, \dots, l\}$ and when $i, j \in \{l+1, \dots, r\}$, where $i \neq j$.

Proof of claim. By Theorem 2.1, since C'_i and C'_j are compatible, either $O'_i \subseteq O'_j$, or $O'_i \supseteq O'_j$, or $O'_i \cap O'_j = \emptyset$. We will prove the claim by showing that the first two cases are impossible. Since $|O'_i| = |O'_j|$ holds by (1), it suffices to show $O'_i \neq O'_j$. For the purpose of a contradiction, suppose that $O'_i = O'_j$. From this, it follows that $|O_i \cap O_j| \geq 2$, which is in conflict with the precondition $|O_i \cap O_j| \leq 1$. \square

Claim 2. For any $i \in \{l+1, \dots, r\}$, there exists at most one $j \in \{1, \dots, l\}$ such that $O'_i \cap O'_j \neq \emptyset$.

Proof of claim. For the purpose of a contradiction, suppose that there exist $v, w \in \{1, \dots, l\}$, $v \neq w$ such that $O'_v \cap O'_i \neq \emptyset$ and $O'_w \cap O'_i \neq \emptyset$.

We show that $\{O'_v, O'_w\}$ is a bipartition of O'_i by proving that $O'_v \cup O'_w \subseteq O'_i$. The statement follows from this since $O'_v \cap O'_w = \emptyset$ by Claim 1 and $|O'_v| + |O'_w| = |O'_i|$. To prove $O'_v \cup O'_w \subseteq O'_i$, we show that $O'_v \subset O'_i$. By Theorem 2.1, $O'_v \cap O'_i \in \{O'_v, O'_i, \emptyset\}$ since characters C'_i and C'_v are compatible. This reduces to $O'_i \cap O'_v \in \{O'_i, O'_v\}$ by our assumption that $O'_v \cap O'_i \neq \emptyset$. From this, $|O'_i| = 4$ and $|O'_v| = 2$ (by (1)). It follows that $O'_v \subset O'_i$. By a similar reasoning, $O'_w \subset O'_i$ follows and we conclude that $O'_v \cup O'_w \subseteq O'_i$.

We are prepared to give the contradiction. F_v and F_w are d-flips while F_i is an i-flip. Thus, $O'_v = O_v - F_v$, $O'_w = O_w - F_w$, and $O'_i = O_i \cup F_i$. Since $\{O'_v, O'_w\}$ is a bipartition of O'_i , $\{O_v - F_v, O_w - F_w\}$ is a bipartition of $O_i \cup F_i$. Since $|F_i| = 1$, either $F_i \subseteq O_v - F_v$ or $F_i \subseteq O_w - F_w$. Without loss of generality, we assume that $F_i \subseteq O_v - F_v$. Thus, $O_w - F_w \subseteq O_i$. From this and $|O_w - F_w| = 2$ (by (1)), it follows that $|O_w \cap O_i| \geq |(O_w - F_w) \cap O_i| = 2$, contradicting the precondition $|O_w \cap O_i| \leq 1$. \square

By Claim 1, $|\bigcup_{i \in \{1, \dots, l\}} O'_i| = 2l$ and

$$\left| \bigcup_{i \in \{l+1, \dots, r\}} O'_i \right| = 4(r-l).$$

By Claim 2, for each $i \in \{l+1, \dots, r\}$, there exists at most one $j \in \{1, \dots, l\}$ such that $O'_i \cap O'_j \neq \emptyset$. Hence, any O_i for $i \in \{l+1, \dots, r\}$ counts for at least two new elements. Therefore, $|\bigcup_{i=1}^r O'_i| \geq 2r$. \square

Theorem 3.1. *FP is \mathcal{NP} -complete.*

Proof. Clearly, $FP \in \mathcal{NP}$. We show that FP is \mathcal{NP} -hard by reduction from the following \mathcal{NP} -complete input-constrained version of the problem *exact cover by 3-sets* (problem SP2 in [33]) [21].

Constrained Exact 3-Cover (CX3C).

Instance: A set X with $|X| = 3q$ for some q and a collection $\mathcal{S} = \{S_1, \dots, S_r\}$ of 3-element subsets of X such that $|S_i \cap S_j| \leq 1$ for any $i, j \in \{1, \dots, r\} \subseteq \mathcal{S}$, $i \neq j$.

Question: Does X contain an *exact cover*, that is, a subset $\mathcal{S}' \subseteq \mathcal{S}$ that partitions X ?

We prove that CX3C \leq_p^m FP by a modification of an argument in [21].

Construction: Given an instance (X, \mathcal{S}) of CX3C as described above, construct an instance (C, k) of FP in which $k = 2(r - q)$ and where, for every $i \in \{1, \dots, r\}$, $O_i = S_i$. This construction obviously takes polynomial time.

The \mathcal{NP} -completeness of FP derives from the following statement:

Claim: \mathcal{S} contains an exact cover for X , if and only if there exists a flip-tuple \mathcal{F} such that $s(\mathcal{F}) \leq k$ and $C \triangleleft \mathcal{F}$ is compatible.

Proof of claim: " \implies ": Let \mathcal{S}' be an exact cover for X . Without loss of generality, we assume that $\mathcal{S}' = \{S_1, \dots, S_q\}$. Let $\mathcal{F} = (F_1, \dots, F_r)$ be a flip tuple, where $F_i = \emptyset$ for every $i \in \{1, \dots, q\}$, and $F_i \subset O_i$ such that $|F_i| = 2$ for every $i \in \{q+1, \dots, r\}$. Thus, $s(\mathcal{F}) = k$. Let $C' = C \triangleleft \mathcal{F}$, where $C' = (C'_1, \dots, C'_r)$. Then, C' contains a collection of disjoint sets and sets of size one, so it is clearly compatible.

" \impliedby ": Suppose that there exists a flip-tuple \mathcal{F} such that $s(\mathcal{F}) \leq k$ and $C \triangleleft \mathcal{F} = (C'_1, \dots, C'_r)$ is compatible. Without loss of generality, let $\mathcal{F} = (F_1, \dots, F_i, F_{i+1}, \dots, F_j, F_{j+1}, \dots, F_r)$ such that $|F_k| = 0$ for $k \in \{1, \dots, i\}$, $|F_k| = 1$ for $k \in \{i+1, \dots, j\}$, and $|F_k| > 1$ for $k \in \{j+1, \dots, r\}$. Let $f_0 = i$, $f_1 = j - i$, and $f_2 = r - j$. We show that $\mathcal{S}' = \{S_1, \dots, S_q\}$ is an exact cover for X by proving that $q = f_0$. \mathcal{S}' is an exact cover for X because the character tuple (C_1, \dots, C_i) is compatible and its characters have disjoint one-sets. We are left to show that $q = f_0$.

Note that $k = 2(r - q) \geq f_1 + 2f_2$. Since

$$r = f_0 + f_1 + f_2,$$

we have

$$f_1 \geq 2(q - f_0). \quad (2)$$

On the other hand, by construction, $|\bigcup_{i \in \{1, \dots, j\}} O'_i| \leq 3q$. Lemma 3.1 implies that

$$\left| \bigcup_{i \in \{1, \dots, j\}} O'_i \right| = 3f_0 + \left| \bigcup_{i \in \{i+1, \dots, j\}} O'_i \right|.$$

Lemma 3.2 states that $|\bigcup_{i \in \{i+1, \dots, j\}} O'_i| \geq 2f_1$. Thus, $3q \geq 3f_0 + 2f_1$, that is,

$$\frac{3}{2}(q - f_0) \geq f_1. \quad (3)$$

Inequalities (2) and (3), imply that $q = f_0$, as claimed. \square

The proof of the following corollary is analogous to the proof of Theorem 3.1.

Corollary 3.1. *DFP is \mathcal{NP} -complete.*

3.2 IFP Is \mathcal{NP} -Complete

We prove the \mathcal{NP} -completeness of IFP by reduction from the chain graph insertion problem IP(chain graph) (see Definition 2.11), shown to be \mathcal{NP} -complete by Yannakakis [34]. Instead of working directly with IFP, we rely on the (by Proposition 2.2) polynomially equivalent \mathcal{M} -free insertion problem, IP(\mathcal{M} -free). We need some graph-theoretic concepts. Throughout this section, we assume that graphs

are connected and we denote by G_A the subgraph of the graph $G = (V, E)$ induced by the nodes $A \subseteq V$.

Definition 3.1 (Chain graph). Let $G = (X, Y, E)$ be a bipartite graph. For a node $v \in Y$, we define its neighborhood to be the set $\eta(v) = \{w \in X \mid \{v, w\} \in E\}$. G is a chain graph if there exists an ordering $\pi: \{1, \dots, |Y|\} \rightarrow Y$ such that $\eta(\pi(1)) \subseteq \dots \subseteq \eta(\pi(|Y|))$.

Chain graphs can be characterized by independent edges.

Definition 3.2 (Independent edges). Let $G = (V, E)$ be a graph and let $V' \subseteq V$. The subgraph of G induced by V' is the graph $G_{V'} = (V', E')$, where $E' = \{\{u, v\} \in E \mid u, v \in V'\}$. Two edges $\{x, y\}, \{u, v\}$ in G are independent if the nodes x, y, u, v are distinct and the graph $G_{\{x, y, u, v\}}$ contains exactly the edges $\{x, y\}, \{u, v\}$.

Theorem 3.2 ([35]). A bipartite graph is a chain graph if and only if it does not contain a pair of independent edges.

Corollary 3.2. A graph $G = (X, Y, E)$ is a chain graph if and only if G is \mathcal{M} -free and has a node $\alpha \in Y$ such that $\eta(\alpha) = X$.

Proof. “ \implies ”: Any \mathcal{M} -graph a, b, c, d, e contains the independent edges $\{a, b\}$ and $\{d, e\}$. Thus, by Theorem 3.2, G is \mathcal{M} -free. Since G is a chain graph there exists an ordering $\pi: \{1, \dots, |Y|\} \rightarrow Y$ such that $\eta(\pi(1)) \subseteq \dots \subseteq \eta(\pi(|Y|))$. It follows that $\eta(\pi(|Y|)) = X$ since G is connected.

“ \impliedby ”: Suppose G is not a chain graph. By Theorem 3.2, the graph G must contain two independent edges $u = \{a, b\}$ and $v = \{d, e\}$. Without loss of generality, we assume $b, d \in X$. Thus, there exist two distinct edges $\{\alpha, b\}, \{\alpha, d\} \in E$. Since the edges u and v are independent, the nodes a, b, α, d, e induce a \mathcal{M} -graph in G . \square

Theorem 3.3. *IFP* is \mathcal{NP} -complete.

Proof. Consider the polynomially equivalent problem $\text{IP}(\mathcal{M}\text{-free})$. Clearly $\text{IP}(\mathcal{M}\text{-free}) \in \mathcal{NP}$. We show that $\text{IP}(\text{chain graph}) \leq_p^m \text{IP}(\mathcal{M}\text{-free})$.

Starting from an instance (G, k) of $\text{IP}(\text{chain graph})$, where $G = (X, Y, E)$ is a bipartite graph and k an integer, we construct an instance (G', k') of $\text{IP}(\mathcal{M}\text{-free})$, where $k' = k$ and where graph $G' = (X', Y', E')$ is obtained as follows: Choose some element $\alpha \notin X \cup Y$ and let $X' = X, Y' = Y \cup \{\alpha\}$, and $E' = E \cup E_\alpha$, where $E_\alpha = \{\{u, \alpha\} \mid u \in X'\}$. This clearly takes polynomial time.

Now, we show that $(G, k) \in \text{IP}(\text{chain graph})$ if and only if $(G', k') \in \text{IP}(\mathcal{M}\text{-free})$.

“ \implies ”: Suppose $(G, k) \in \text{IP}(\text{chain graph})$, then there exists an edge set $D \subseteq \{\{u, v\} \mid u \in X, v \in Y\} - E$ such that $|D| \leq k$ and $H = (X, Y, E \cup D)$ is a chain graph. Thus, $H' = (X', Y', E' \cup D)$ is a chain graph which is \mathcal{M} -free by Corollary 3.2.

“ \impliedby ”: Suppose that there exists an edge set $D \subseteq \{\{u, v\} \mid u \in X, v \in Y\} - E'$ such that $|D| \leq k$ and $H' = (X', Y', E' \cup D)$ is \mathcal{M} -free. By construction for $\alpha \in Y'$, we have $\eta(\alpha) = X$ and it follows from Corollary 3.2 that $H = (X, Y, E \cup D)$ is a chain graph. \square

4 APPROXIMATION ALGORITHMS

We present two results. The first is a fixed-ratio approximation algorithm for the minimum-flip problem when the underlying character graph has a degree bound. The second is an approximation scheme for a variant of the minimum-flip problem where, rather than minimizing a distance function, we maximize a similarity score.

4.1 Degree-Bounded Instances

We begin by introducing the notion of a degree-bounded instance of the minimum-flip problem.

Definition 4.1 (Degree-bounded character tuples). A character tuple \mathcal{C} has degree bound d if, in its character graph $G = (X, Y, E)$, the degree of each node in X is at most d . That is, each character has at most d taxa.

Since the \mathcal{M} -free property is hereditary, the algorithm of Natanzon et al. [36] approximates $\text{EP}(\mathcal{M}\text{-free})$ and $\text{DP}(\mathcal{M}\text{-free})$ within a factor of $6d$. We adapt this algorithm for \mathcal{M} -free graphs as shown below and improve the approximation ratio to $2d$.

Algorithm APPROX($G = (X, Y, E)$)

```

1  $A \leftarrow \emptyset$ 
2 while  $G_{X-A}$  is not  $\mathcal{M}$ -free
3   do Let  $\mathcal{M}$  be an induced  $\mathcal{M}$ -graph in  $A$  described
         by path  $\langle a, b, c, d, e \rangle$ 
4      $A \leftarrow A \cup \{b, d\}$ 
5 return  $F = \{\{x, y\} \in E \mid x \in A \wedge y \in Y\}$ 

```

Theorem 4.1. APPROX approximates $\text{DP}(\mathcal{M}\text{-free})$ and $\text{EP}(\mathcal{M}\text{-free})$ within a factor of $2d$.

Proof. The proofs for the correctness and the approximation ratio are similar to Natanzon et al. [36].

Correctness: The algorithm terminates since, by deleting all edges incident to nodes b, d in a \mathcal{M} -graph a, b, c, d, e , no new \mathcal{M} -graph is created. After termination, the graph $(X, Y, E - F)$ is \mathcal{M} -free.

Approximation ratio: Let S_{opt} be an optimal solution for either $\text{EP}(\mathcal{M}\text{-free})$ or $\text{DP}(\mathcal{M}\text{-free})$. Every \mathcal{M} -graph a, b, c, d, e found by the algorithm is deleted by isolating the nodes b and d , which cannot create another \mathcal{M} -graph. Thus every \mathcal{M} -graph a, b, c, d, e found by the algorithm has to be deleted by some edge in S_{opt} . This edge is incident to either node b or d . Every \mathcal{M} -graph found by the algorithm after the \mathcal{M} -graph a, b, c, d, e was removed by the algorithm does not contain the nodes b and d . Hence, every \mathcal{M} -graph found by the algorithm has to be deleted by an edge in S_{opt} that is not incident to node b or d . So, every \mathcal{M} -graph found by the algorithm is removed by a distinct edge in S_{opt} . After termination, the number of deleted \mathcal{M} -graphs, which is $|A|/2$, is smaller than or equal to $|S_{opt}|$. Now, $|F| \leq d|A| \leq 2d|S_{opt}|$, from which the ratio of $2d$ follows. \square

4.2 Approximation under Similarity-Based Scoring

We now study a variant of the minimum-flip problem in which the goal is to maximize a similarity measure. Our definitions are adapted from [21].

Definition 4.2 (Similarity measures). Let T be a phylogeny where $\mathcal{L}(T) = M$, let v be a node of T , and let X_v denote the cluster of T corresponding to v (that is, the set of all leaves in the subtree of T rooted at v). Let $C = (N, O)$ be a character. The similarity of C to v is

$$\text{sim}(C, v) = |O \cap X_v| + |(N - O) \cap (M - X_v)|$$

and the similarity of C to T is

$$\text{sim}(C, T) = \max_{v \in T} \{\text{sim}(C, v)\}.$$

Given a character tuple \mathcal{C} , define the similarity of \mathcal{C} to T as

$$\text{sim}(\mathcal{C}, T) = \sum_{i=1}^r \text{sim}(C_i, T).$$

Let T_1, T_2 be phylogenies over M . For $i \in \{1, 2\}$, let v_i denote some vertex in T_i and let X_i denote the cluster of T_i corresponding to v_i . The similarity of v_1 to v_2 is

$$\text{sim}(v_1, v_2) = |X_1 \cap X_2| + |(M - X_1) \cap (M - X_2)|.$$

Definition 4.3. The directed fractional character compatibility problem, *DFCC*, is the following: Given a character tuple \mathcal{C} , find a phylogeny T over M that maximizes $\text{sim}(\mathcal{C}, T)$.

The following relationships are easy to establish.

Proposition 4.1. For any character tuple \mathcal{C} and any phylogeny T over M ,

1. $\text{sim}(\mathcal{C}, T) \geq (1/2) \sum_{i=1}^r |N_i|$ and
2. $\text{sim}(\mathcal{C}, T) = \sum_{i=1}^r |N_i| - \text{dist}(\mathcal{C}, T)$. Thus, T is an optimum solution to *DFCC* if and only if T is an optimum solution to the minimum-flip problem.

We present an approximation scheme for a special case of *DFCC*. The details are similar to those of the approximation scheme for *FCC* of [21]; thus, we only sketch the main points. Note that, in light of Proposition 4.1.2, an approximation scheme for *DFCC* does not provide an approximation scheme for the minimum-flip problem.

The algorithm centers on the existence of a k -bin contraction of the optimum tree.

Definition 4.4 (k -bin contraction [21]). Let T be a phylogeny for M , where $|M| = n$. Tree T_k is a k -bin contraction of T if there is a partition of M into bins M_1, M_2, \dots, M_k such that:

1. $|M_i| \leq 6n/k$ for each $i \in \{1, \dots, k\}$.
2. There is one vertex v_i , called the bin root whose $|M_i|$ children are the elements of M_i .
3. For each vertex v of T , there is a vertex v' of T_k such that $\text{sim}(v, v') \geq n - 6n/k$.

The next result follows from [21].

Lemma 4.1. Any phylogeny T over M has a k -bin contraction T_k such that $\text{sim}(\mathcal{C}, T_k) \geq \text{sim}(\mathcal{C}, T) - (6/k)rn$.

Definition 4.5 (Completions and kernels). Let K be a tree with k leaves, u_1, \dots, u_k . A completion of K is a phylogeny T for M that is obtained by partitioning M into sets M_1, M_2, \dots, M_k and, for each $i \in \{1, \dots, k\}$, making each taxon in M_i a child of u_i . Tree K is called a kernel for T .

Definition 4.6 (TBA). The taxon to bin assignment problem, *TBA*, is as follows: Given a tuple \mathcal{C} of characters over M and a tree K with k leaves, find a completion T of K that maximizes $\text{sim}(\mathcal{C}, T)$.

The key to approximating *DFCC* is the result below, again derived from [21].

Lemma 4.2. For each $\epsilon > 0$, there is a polynomial time algorithm that, for any instance \mathcal{C} and K of *TBA*, produces a completion T of K such that $\text{sim}(\mathcal{C}, T) \geq \text{sim}(\mathcal{C}, \hat{T}) - \epsilon(n+r)^2$, where \hat{T} is an optimal completion of T .

In view of Lemmas 4.1 and 4.2, our strategy is to generate all possible kernels K with at most k leaves and, for each such K , solve *TBA* approximately. The tree T obtained in this fashion satisfies

$$\text{sim}(\mathcal{C}, T) \geq \text{sim}(\mathcal{C}, T^*) - (6/k)rn - \epsilon(r+n)^2, \quad (4)$$

where T^* is an optimum solution to *DFCC*. This and Proposition 4.1.1 lead to our main result.

Theorem 4.2. For each $\epsilon > 0$, there exists a polynomial-time algorithm that, for each instance \mathcal{C} of *DFCC* such that the characters in \mathcal{C} are complete and $r = \Theta(n)$, produces a tree T such that $\text{sim}(\mathcal{C}, T) \geq (1 - \epsilon)\text{sim}(\mathcal{C}, T^*)$.

In the above theorem, the condition that \mathcal{C} consist of complete characters is necessary since, in this case, Proposition 4.1.1 implies that $\text{sim}(\mathcal{C}, T^*) \geq nr/2$, which, with (4) and the assumption that $r = \Theta(n)$ combines to give us the desired result. For incomplete characters, a similar effect is achieved if there is a known lower bound on $\sum_{i=1}^r |N_i|$ (that is, if there is an upper bound on the amount of missing information). This is expressed formally below.

Definition 4.7 (Density of a character tuple). Let γ be a real number in the interval $(0, 1]$. Character tuple \mathcal{C} has density γ if $\sum_{i=1}^r |N_i| \geq \gamma nr$.

The next result follows by the same argument as Theorem 4.2.

Theorem 4.3. For each $\gamma \in (0, 1]$ and each $\epsilon > 0$, there exists a polynomial-time algorithm that, for each instance \mathcal{C} of density γ of *DFCC* such that $r = \Theta(n)$, produces a tree T such that $\text{sim}(\mathcal{C}, T) \geq (1 - \epsilon)\text{sim}(\mathcal{C}, T^*)$.

5 FIXED-PARAMETER ALGORITHMS

A number of optimization problems can be solved efficiently when one or more input parameters are fixed [37]. This motivates the following definition:

Definition 5.1. A decision problem parameterized by k is fixed parameter tractable if there is an algorithm that decides the problem in $O(f(k)p(n))$ time, for an input of size n , where $p(n)$ is a polynomial function and f an arbitrary function.

All our results are presented in the polynomially equivalent graph-theoretic framework of \mathcal{M} -free edge modification problems. Bounding the number of allowed

edge modifications by a fixed parameter leads to the following problems:

Definition 5.2. Let $G = (X, Y, E)$ be a bipartite graph, k be a positive integer, and $A = \{\{x, y\} \mid x \in X, y \in Y\}$. The \mathcal{M} -free k -edit problem is to determine if there exists a set $F \subseteq A$, where $|F| \leq k$, such that $G' = (X, Y, E \Delta F)$ is \mathcal{M} -free. We additionally have the following two restricted versions of the \mathcal{M} -free k -edit problem:

- The \mathcal{M} -free k -deletion problem is the version where we require that $F \subseteq E$.
- The \mathcal{M} -free k -insertion problem is the version where we require that $F \subseteq A - E$.

The fixed-parameter tractability of \mathcal{M} -free edge modification problems follows from a general result of Cai on graph modification problems [38] (Theorem 2). The running time of Cai's algorithm when adapted for \mathcal{M} -free edge modification problems is given by the following theorem:

Theorem 5.1. Let $G = (X, Y, E)$ be a bipartite graph, $x = |X|$ and $y = |Y|$. The \mathcal{M} -free edge modification problem versions k -edit, k -insertion, and k -deletion, can be solved in time $O(6^k xy)$, $O(2^k xy)$, and $O(4^k xy)$, respectively.

Proof. First, we give Cai's algorithm and then analyze its running time for the \mathcal{M} -free k -edit problem.

Cai's algorithm: Repeat the following two steps until either the resulting graph is \mathcal{M} -free or all k edge modifications have been done. The answer to the \mathcal{M} -free k -edit problem is "yes" in the former case and "no" otherwise.

Step 1: Find a forbidden \mathcal{M} -graph H in G .

Step 2: Modify G by either deleting or adding an edge to H .

Running time: A forbidden \mathcal{M} -graph in G can be found in time $O(xy)$ by the following observation: An \mathcal{M} -graph exists exactly when the corresponding two characters are incompatible. Character incompatibility can be detected in $O(xy)$ by Gusfield's algorithm [39], which can be adapted to return an \mathcal{M} -graph that contributed to the incompatibility. For the forbidden subgraph H , there are two ways to add an edge and four ways to delete an edge from H . Thus, the total number of graphs generated by the algorithm is at most 6^k . Hence, the overall running time is $O(6^k xy)$.

For the \mathcal{M} -free k -insertion problem, Step 2 only adds edges from which a running time of $O(2^k xy)$ follows. By only deleting edges in Step 2, the \mathcal{M} -free k -deletion problem is solved in $O(4^k xy)$ time. \square

6 CONCLUDING REMARKS

We have introduced an approach to supertree construction, the minimum-flip problem, that is based on error correction. We have shown that this problem is \mathcal{NP} -complete, but also that it can be solved approximately and that it is fixed-parameter tractable in some cases. It is open as to whether there exists a polynomial time approximation scheme for the minimum-flip problem; we have, however, shown that a related optimization problem does have such a scheme. Perhaps the most important open algorithmic problem is whether or not any positive results (approximation or

fixed-parameter algorithms) for the minimum-flip problem can be obtained when the input characters are incomplete since this is precisely the situation one faces in supertree construction. In graph theoretical terms, this is an edge modification problem in which the edge set of the input graph is only partly known. The question is whether the known part of the graph can be edited by k operations such that there is a completion of the graph with the desired property (e.g., to be \mathcal{M} -free).

Despite the inherent complexity of the minimum-flip problem, experimental studies, reported elsewhere [19], [40], [41], show that it is an effective approach for constructing supertrees. Our studies used exact algorithms for small trees (fewer than 16 taxa) [41] and heuristics for larger trees [19]. In all cases, minimum-flip supertrees were more accurate than those obtained by Sempel and Steel's MC algorithm (which, however, is polynomial). Minimum-flip supertrees were also generally more accurate than those obtained by MRP.

Our approximation and fixed-parameter algorithms for the minimum-flip problem were obtained under certain constraints that do not often hold in practice. Currently, programs for computing minimum-flip supertrees (e.g., Rainbow [42]) rely on heuristics. For practical applications, it would help to know if there exists a \mathcal{PTAS} for the minimum-flip problem and if there is a fixed-parameter algorithm for the minimum-flip problem with missing data.

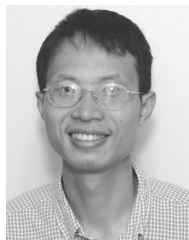
ACKNOWLEDGMENTS

The authors want to thank the anonymous referees, who gave insightful comments that have helped to improve the presentation of this work. The research of authors D. Chen, O. Eulenstein, and M. Sanderson was supported by the US National Science Foundation (NSF) under grant award number 0334832. The research of D. Fernández-Baca was supported in part by the NSF under grant award numbers *CCR9988348* and *0334832*.

REFERENCES

- [1] D.H. Huson, S.M. Nettles, and T.J. Warnow, "Disk-Covering, a Fast-Converging Method for Phylogenetic Tree Reconstruction," *J. Computational Biology*, vol. 6, nos. 3-4, pp. 369-386, 1999.
- [2] *Phylogenetic Supertrees: Combining Information to Reveal the Tree of Life*, O.R.P. Bininda-Emonds, ed. Kluwer Academic, 2004.
- [3] A.D. Gordon, "Consensus Supertrees: The Synthesis of Rooted Trees Containing Overlapping Sets of Labelled Leaves," *J. Classification*, vol. 9, pp. 335-348, 1986.
- [4] M.A. Steel, "The Complexity of Reconstructing Trees from Qualitative Characters and Subtrees," *J. Classification*, vol. 9, pp. 91-116, 1992.
- [5] M.J. Sanderson, A. Purvis, and C. Henze, "Phylogenetic Supertrees: Assembling the Trees of Life," *Trends Ecological Evolution*, vol. 13, pp. 105-109, 1998.
- [6] C. Sempel and M.A. Steel, "A Supertree Method for Rooted Trees," *Discrete Applied Math.*, vol. 105, pp. 147-158, 2000.
- [7] M.A. Steel, A.W.M. Dress, and S. Böcker, "Simple but Fundamental Limitations on Supertree and Consensus Tree Methods," *Systematic Biology*, vol. 49, pp. 363-368, 2000.
- [8] M.J. Donoghue, "Phylogenies and the Analysis of Evolutionary Sequences, with Examples from Seed Plants," *Evolution*, vol. 43, pp. 1137-1156, 1989.
- [9] A. Ortolani, "Spots, Stripes, Tail Tips and Dark Eyes: Predicting the Function of Carnivore Colour Patterns Using the Comparative Method," *Biological J. Linnean Soc.*, vol. 67, pp. 433-476, 1999.

- [10] A. Purvis, "A Modification to Baum and Ragan's Method for Combining Phylogenetic Trees," *Systematic Biology*, vol. 44, pp. 251-255, 1995.
- [11] O.R.P. Bininda-Emonds, J.L. Gittleman, and A. Purvis, "Building Large Trees by Combining Phylogenetic Information: A Complete Phylogeny of the Extant Carnivora (Mammalia)," *Biological Rev.*, vol. 74, pp. 143-175, 1999.
- [12] M.F. Wojciechowski, M.J. Sanderson, K.P. Steele, and A. Liston, "Molecular Phylogeny of the 'Temperate Herbaceous Tribes' of Papilionoid Legumes: A Supertree Approach," *Advances in Legume Systematics*, P. Herendeen and A. Bruneau, eds., vol. 9, pp. 277-298, 2000.
- [13] F.G.R. Liu, M.M. Miyamoto, N.P. Freire, P.Q. Ong, M.R. Tennant, T.S. Young, and K.F. Gugel, "Molecular and Morphological Supertrees for Eutherian (Placental) Mammals," *Science*, vol. 291, pp. 1786-1789, 2001.
- [14] T.J. Davies, T.G. Barraclough, M.W. Chase, P.S. Soltis, D.E. Soltis, and V. Savolainen, "Darwin's Abominable Mystery: Insights from a Supertree of the Angiosperms," *Proc. Nat'l Academy Sciences USA*, vol. 101, pp. 1904-1909, 2004.
- [15] E. Grotkopp, M. Rejmanek, M.J. Sanderson, and T.L. Rost, "Evolution of Genome Size in Pines (Pinus) and Its Life-History Correlates: Supertree Analyses," *Evolution*, pp. 1705-1729, 2004.
- [16] D.R. Brooks, "Hennig's Parasitological Method: A Proposed Solution," *Systems Zoology*, vol. 30, pp. 325-331, 1981.
- [17] B.R. Baum, "Combining Trees as a Way of Combining Data Sets for Phylogenetic Inference, and the Desirability of Combining Gene Trees," *Taxon*, vol. 41, pp. 3-10, 1992.
- [18] M.A. Ragan, "Phylogenetic Inference Based on Matrix Representation of Trees," *Molecular Phylogenetics and Evolution*, vol. 1, pp. 53-58, 1992.
- [19] O. Eulenstein, D. Chen, J.G. Burleigh, D. Fernández-Baca, and M.J. Sanderson, "Performance of Flip Supertree Construction with a Heuristic Algorithm," *Systematic Biology*, vol. 53, pp. 299-308, 2003.
- [20] S. Kannan, T. Warnow, and S. Yooshef, "Computing the Local Consensus of Trees," *Proc. Symp. Discrete Algorithms*, pp. 68-77, 1995.
- [21] P. Kearney, M. Li, J. Tsang, and T. Jiang, "Recovering Branches on the Tree of Life: An Approximation Algorithm," *Proc. Symp. Discrete Algorithms*, pp. 537-546, 1999.
- [22] D. Bryant, J. Tsang, P.E. Kearney, and M. Li, "Computing the Quartet Distance between Evolutionary Trees," *Proc. Symp. Discrete Algorithms*, pp. 285-286, 2000.
- [23] A.V. Aho, Y. Sagiv, T.G. Szymanski, and J.D. Ullman, "Inferring a Tree from Lowest Common Ancestors with an Application to the Optimization of Relational Expressions," *SIAM J. Computing*, vol. 10, no. 3, pp. 405-421, 1981.
- [24] M.R. Henzinger, V. King, and T. Warnow, "Constructing a Tree From Homeomorphic Subtrees, with Applications to Computational Evolutionary Biology," *Algorithmica*, vol. 24, pp. 1-13, 1999.
- [25] R.D.M. Page, "Modified Mincut Supertrees," *Proc. Int'l Workshop Algorithms in Bioinformatics (WABI)*, D. Gusfield and R. Guigó, eds., pp. 300-315, Sept. 2002.
- [26] R.L. Graham and L.R. Foulds, "Unlikelihood that Minimal Phylogenies for a Realistic Biological Study Can Be Constructed in Reasonable Computation Time," *Math. Bioscience*, vol. 60, pp. 133-142, 1982.
- [27] J. Felsenstein, "PHYLIP," <http://evolution.genetics.washington.edu/phylip.html>, 2006.
- [28] I. Pe'er, R. Shamir, and R. Sharan, "Incomplete Directed Perfect Phylogeny," *Combinatorial Pattern Matching*, R. Giancarlo and D. Sankoff, eds., Springer-Verlag, pp. 143-153, 2000.
- [29] G.F. Estabrook, C. Johnson, and F.R. McMorris, "An Idealized Concept of the True Cladistic Character?" *Math. Bioscience*, vol. 23, pp. 263-272, 1975.
- [30] D. Gusfield, *Algorithms on Strings, Trees, and Sequences*. New York: Cambridge Univ. Press, 1997.
- [31] J.S. Farris, "On Comparing the Shapes of Taxonomic Trees," *Systematic Zoology*, vol. 22, pp. 50-54, 1976.
- [32] D.L. Swofford, G.J. Olsen, P.J. Waddell, and D.M. Hillis, "Phylogenetic Inference," *Molecular Systematics*, D.M. Hillis, C. Moritz, and B.K. Mable, eds., pp. 407-509. Sunderland, Mass.: Sinauer Assoc., 1996.
- [33] M.R. Garey and D.S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. New York: W.H. Freeman, 1979.
- [34] M. Yannakakis, "Computing the Minimum Fill-In Is NP-Complete," *SIAM J. Algebraic and Discrete Methods*, vol. 2, no. 1, pp. 77-79, 1981.
- [35] M. Yannakakis, "Node-Deletion Problems on Bipartite Graphs," *SIAM J. Computing*, vol. 10, pp. 310-326, 1981.
- [36] A. Natanzon, R. Shamir, and R. Sharan, "Complexity Classification of Some Edge Modification Problems," *Discrete Applied Math.*, vol. 113, no. 1, pp. 109-128, 2001.
- [37] R.G. Downey and M.R. Fellows, *Parameterized Complexity*. Springer, 1997.
- [38] L. Cai, "Fixed-Parameter Tractability of Graph Modification Problems for Hereditary Properties," *Information Processing Letters*, vol. 58, pp. 171-176, 1996.
- [39] D. Gusfield, "Efficient Algorithms for Inferring Evolutionary History," *Networks*, vol. 21, pp. 19-28, 1992.
- [40] J.G. Burleigh, D. Chen, O. Eulenstein, D. Fernández-Baca, and M.J. Sanderson, "Minimum Flip Supertrees," *Phylogenetic Supertrees: The Book*, Kluwer, to appear.
- [41] D. Chen, L. Diao, O. Eulenstein, D. Fernández-Baca, and M.J. Sanderson, "Flipping: A Supertree Construction Method," *Bioconsensus, DIMACS: Series in Discrete Math. and Theoretic Computer Science*, vol. 61, pp. 135-160, Providence, R.I.: Am. Math. Soc., 2003.
- [42] D. Chen, O. Eulenstein, and D. Fernández-Baca, "Rainbow: A Toolbox for Phylogenetic Supertree Construction and Analysis," *Bioinformatics Advance Access*, 2004.



Duhong Chen received the MS degree in biophysics from China Agricultural University, Beijing, in August 1998 and the MS in computer science from Iowa State University, Ames, in May 2002. He is currently a PhD student in computer science at Iowa State University. His research interests lie in algorithmic phylogenetic tree and supertree construction, graph partition, tree decomposition, and data mining.



Oliver Eulenstein received the PhD degree in computer science from the Rheinische Friedrich-Wilhelms-Universität Bonn, Germany, in October 1998. He is an assistant professor in computer science at Iowa State University. Prior to joining Iowa State University in August 2000, he was a postdoctoral fellow in the Computer Science Department at the University of California, Davis. His research focuses on the development of algorithms for problems in molecular biology.



David Fernández-Baca received the BS degree from the National Autonomous University of Mexico (UNAM) in 1980 and the MS and PhD degrees from the University of California, Davis, in 1983 and 1986, respectively. He is a professor of computer science at Iowa State University. His work experience includes a research associate position at the Instituto de Investigaciones Eléctricas in Cuernavaca, Mexico, a visiting position at DIMACS, and a stint as department chair. His research focuses on computational phylogenetics and on sensitivity analysis in combinatorial optimization.



Michael Sanderson is a professor in evolution and ecology at the University of California, Davis. His research focuses on reconstructing phylogenetic trees and studying the evolutionary process in the context of those trees. In addition to computational biology, he is interested in the biological diversity of *Astragalus*, the largest genus of flowering plants, with well over 2,500 species.