Discrete Applied Mathematics 257 (2019) 101–114

Contents lists available at ScienceDirect

Discrete Applied Mathematics

journal homepage: www.elsevier.com/locate/dam

The approximability of maximum rooted triplets consistency with fan triplets and forbidden triplets☆

Katharina Dannenberg [a], Jesper Jansson [b,*], Andrzej Lingas [c], Eva-Marta Lundell [c]

[a] Institute for Theoretical Computer Science and Graduate School for Computing in Medicine and Life Sciences, University of Lübeck, Ratzeburger Allee 160, 23562 Lübeck, Germany
[b] Department of Computing, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong
[c] Department of Computer Science, Lund University, Box 118, 221 00 Lund, Sweden

ARTICLE INFO

Article history:
Received 19 August 2016
Received in revised form 29 August 2018
Accepted 30 August 2018
Available online 3 November 2018

Keywords:
Phylogenetic tree
Rooted triplet
Approximation algorithm
Dynamic programming
Smooth polynomial integer program

ABSTRACT

The NP-hard *maximum rooted resolved triplets consistency problem* (MRTC) takes as input a set S of leaf labels and a set $\mathcal{R}$ of resolved triplets over S and asks for a rooted phylogenetic tree that is consistent with the maximum number of elements in $\mathcal{R}$. This article studies the approximability of a generalization of the problem called the *maximum rooted triplets consistency problem* (MTC) where in addition to resolved triplets, the input may contain fan triplets, forbidden resolved triplets, and forbidden fan triplets. To begin with, we observe that MTC admits a 1/4-approximation in polynomial time. Next, we generalize Wu's exact exponential-time algorithm for MRTC (Wu, 2004) to MTC. Forcing the algorithm to always output a rooted k-ary phylogenetic tree for any specified $k \geq 2$ subsequently leads to an exponential-time approximation scheme (ETAS) for MTC. We then present a polynomial-time approximation scheme (PTAS) for *complete* instances of MTC (meaning that for every $S' \subseteq S$ with $|S'| = 3$, $\mathcal{R}$ contains at least one rooted triplet involving the leaf labels in S'), based on the techniques introduced by Jiang et al. (2001) for a related problem. We also study the computational complexity of MTC restricted to fan triplets and forbidden fan triplets. Finally, extensions to weighted instances are considered.

1. Introduction

Phylogenetic trees are used by scientists to describe treelike evolutionary history for objects such as biological species, natural languages, manuscripts, etc. [7]. Inferring an accurate phylogenetic tree from experimental data can be a difficult task; for example, computationally expensive methods like maximum likelihood that are known to yield good trees may be impractical for large data sets [6]. One potential remedy is the divide-and-conquer approach [6,12,19]: first apply some expensive method to obtain a collection of highly reliable trees for small, overlapping subsets of the leaf labels, and then use a computationally less intensive method to merge these small trees into a larger phylogenetic tree (also known as a *phylogenetic supertree*).

☆ A preliminary version of this article appeared in *Proceedings of the 26th Annual Symposium on Combinatorial Pattern Matching* (CPM 2015), volume 9133 of *Lecture Notes in Computer Science*, pp. 272–283, Springer International Publishing Switzerland, 2015.

* Corresponding author.
E-mail addresses: dannenberg@tcs.uni-luebeck.de (K. Dannenberg), jesper.jansson@polyu.edu.hk (J. Jansson), Andrzej.Lingas@cs.lth.se (A. Lingas), Eva-Marta.Lundell@cs.lth.se (E.-M. Lundell).

https://doi.org/10.1016/j.dam.2018.08.028
0166-218X/

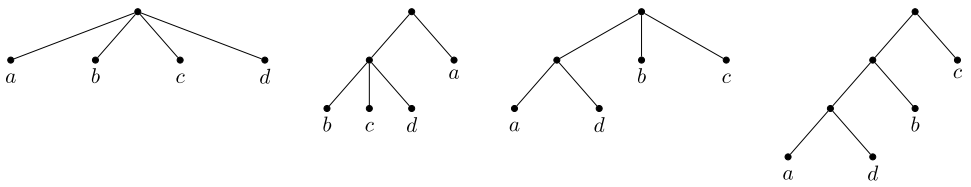

Fig. 1. Four of the optimal solutions to the instance of MTC in the example in Section 1.1. Each of the shown trees satisfies three of the four constraints from $\mathcal{R}$. The three leftmost solutions satisfy the same constraints from $\mathcal{R}$ even though they are non-isomorphic.

A concept that captures the combinatorial aspects of the smallest meaningful building blocks of phylogenetic trees in the rooted case is *rooted triplets consistency*. Given a set $\mathcal{R}$ of possibly contradicting rooted phylogenetic trees with exactly three leaves each (so-called *rooted triplets*), the *maximum rooted triplets consistency problem* asks for a tree that contains as many of the rooted triplets in $\mathcal{R}$ as possible as embedded subtrees. Most previous work on the topic (e.g., [1,4,5,10,21,23,24]) has focused on the case where all the given rooted triplets are *resolved triplets*, meaning that they are binary. This article considers a more general problem variant where $\mathcal{R}$ may also contain non-binary triplets called *fan triplets* that should preferably be included in the output tree as well as *forbidden triplets* that should be avoided. In cases where the raw experimental data is of poor quality and an output tree containing a smaller number of internal nodes would be preferred to a fully resolved one (as the latter might suggest many unsupported groupings of the leaf labels), including some appropriately chosen fan triplets in $\mathcal{R}$ may be helpful. Forbidden triplets can be used to exclude evolutionary relationships that are known for sure to be false.

1.1. Definitions

A (rooted) *phylogenetic tree* is a rooted tree in which every internal node has at least two children and all leaves are distinctly labeled. (We do not consider unrooted phylogenetic trees here.) All phylogenetic trees are assumed to be unordered in the sense that there is no fixed left-to-right ordering of the children of the internal nodes. The *degree* of a node u in a phylogenetic tree is the number of children of u and the *degree* of a phylogenetic tree equals the maximum of all of its nodes' degrees. For any integer $k \geq 2$, a *k-ary phylogenetic tree* is defined as a phylogenetic tree in which every internal node has degree at most k.

The set of all leaf labels in a phylogenetic tree T is denoted by $\Lambda(T)$. To simplify the presentation, we identify each leaf in T with the unique element in $\Lambda(T)$ that labels it. For any $x, y \in \Lambda(T)$, $lca^T(x, y)$ is the lowest common ancestor in T of x and y.

Suppose that T is a phylogenetic tree. For any distinct $x, y, z \in \Lambda(T)$, define the following four types of constraints on T:

1. $xy|z$, specifying that $lca^T(x, y)$ should be a proper descendant of $lca^T(x, z)$ (or equivalently, that $lca^T(x, y)$ should be a proper descendant of $lca^T(y, z)$).
2. $x|y|z$, specifying that $lca^T(x, y) = lca^T(x, z) = lca^T(y, z)$ should hold.
3. $\neg(xy|z)$, specifying that $lca^T(x, y)$ should not be a proper descendant of $lca^T(x, z)$ (or equivalently, that $lca^T(x, y)$ should not be a proper descendant of $lca^T(y, z)$).
4. $\neg(x|y|z)$, specifying that the same node should not be the lowest common ancestor of a and b for all pairs $a, b \in \{x, y, z\}$.

The *maximum rooted triplets consistency problem* (MTC) is: given a set S of leaf labels and a set $\mathcal{R}$ of constraints on S as defined above, output a phylogenetic tree T with $\Lambda(T) = S$ that satisfies as many constraints from $\mathcal{R}$ as possible. In this article, the special case of MTC where all constraints in $\mathcal{R}$ are of type 1 is called the *maximum rooted resolved triplets consistency problem* (MRTC).[1] For any $x, y, z \in S$, the two constraints $xy|z$ and $yx|z$ express the same condition on T, so we assume without loss of generality that the input $\mathcal{R}$ to MTC contains at most one of them, and similarly that $\mathcal{R}$ contains at most one of the six constraints $x|y|z$, $y|x|z$, $\ldots$, $z|y|x$, at most one of $\neg(xy|z)$ and $\neg(yx|z)$, and at most one of $\neg(x|y|z)$, $\neg(y|x|z)$, $\ldots$, $\neg(z|y|x)$. Also observe that T cannot be binary if it satisfies a constraint of type 2.

Example. Let $S = \{a, b, c, d\}$ and let $\mathcal{R}$ consist of the following constraints: $ab|c$ [type 1], $b|c|d$ [type 2], $\neg(ab|d)$ [type 3], $\neg(ac|d)$ [type 3]. Then no phylogenetic tree can satisfy all four constraints in $\mathcal{R}$, so the phylogenetic tree T with $\Lambda(T) = S$ in which all leaves are directly attached to the root is an optimal solution to MTC as it satisfies three constraints. Note that T is not the unique optimal solution for this instance; some other optimal solutions are displayed in Fig. 1. □

To express the size of an instance of MTC, we write $n = |S|$ and $m = |\mathcal{R}|$. An instance $(S, \mathcal{R})$ of MTC is *complete* if, for every $S' \subseteq S$ with $|S'| = 3$, $\mathcal{R}$ contains at least one constraint involving all three elements in S'. An approximation algorithm $\mathcal{A}$ for MTC is said to have *approximation ratio* f (where $0 \leq f \leq 1$), if, for every possible input, the number of input constraints satisfied by $\mathcal{A}$'s output is at least f times the number of input constraints satisfied by an optimal solution.

[1] MRTC is called MAX-LEVEL-0 in [4], MaxRTC in [5], MILCT in [10,15], MaxCL-o in [13], MTC in [21], and MCTT in [23,24].

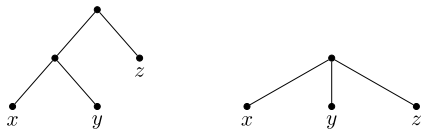

Fig. 2. The type-1 constraint $xy|z$ and the type-2 constraint $x|y|z$ correspond to the resolved triplet on the left and the fan triplet on the right, respectively.

Remark 1. Phylogenetic trees with exactly three leaves are also referred to as *rooted triplets* in the literature. A rooted triplet t is either a binary or a non-binary tree. In the former case, t is a *resolved triplet* and always satisfies a constraint of type 1, and if this constraint is also satisfied in a phylogenetic tree T then t and T are said to be *consistent*. Similarly, if t is non-binary then t is called a *fan triplet* and always satisfies a constraint of type 2; if it is also satisfied in a phylogenetic tree T then t and T are *consistent*. Thus, an equivalent formulation of MTC is: given a "good" set $\mathcal{G}$ and a "bad" set $\mathcal{B}$ of rooted triplets, output a phylogenetic tree T with $\Lambda(T) = \bigcup_{t\in\mathcal{G}\cup\mathcal{B}} \Lambda(t)$ maximizing $|T(\mathcal{G})| + |\mathcal{B} \setminus T(\mathcal{B})|$, where $T(\mathcal{X})$ for any set $\mathcal{X}$ of rooted triplets is the subset of $\mathcal{X}$ consistent with T. In analogy with this terminology, constraints of type 1, 2, 3, and 4 are also called *resolved triplets*, *fan triplets*, *forbidden resolved triplets*, and *forbidden fan triplets* from here on. See Fig. 2 for an illustration. Non-binary nodes in a phylogenetic tree (such as the root of the fan triplet in Fig. 2) can be used to represent so-called *hard polytomies* in biology.

1.2. Previous results

An $O(mn)$-time algorithm by Aho et al. [1] can determine if there exists a phylogenetic tree consistent with *all* of the resolved triplets (i.e., constraints of type 1) in a given set, and if so, output such a tree. Its time complexity was improved to $\min\{O(n+mn^{1/2}),\ O(m+n^2\log n)\}$ by Henzinger et al. [12]. Ng and Wormald [20] extended Aho et al.'s algorithm to the case where the input also contains fan triplets. Later, He et al. [11] extended it to the case where the input consists of resolved triplets and forbidden resolved triplets, and the resulting running time to determine if there exists a phylogenetic tree that satisfies all the input constraints is $O((m+n)n\log n)$.

In comparison, the optimization versions of rooted triplets consistency are computationally harder. MRTC is NP-hard [3,15,24], even if restricted to complete instances [13]. Furthermore, MRTC in the non-complete case is APX-hard [4]. Obviously, these hardness results carry over to MTC as it is a generalization of MRTC. As for positive results for MRTC, Gąsieniec et al. [10] presented a top-down, polynomial-time 1/3-approximation algorithm (see Section 5.1 in [10]). It was generalized to a polynomial-time 1/3-approximation algorithm for the problem variant where all input constraints are of type 1 or type 3 in [11]. Wu [24] proposed a bottom-up, polynomial-time heuristic for MRTC that was shown experimentally to perform well in practice, and other heuristics for MRTC (with unknown approximation ratios) have been published in Section of [10] and in [14,21,23]. An exact algorithm for MRTC running in $O(3^n\cdot(m+n^2)) = O^*(3^n)$ time, using dynamic programming, was given by Wu in [24]. We also remark that the complementary version of MRTC in which the objective is to remove as few elements as possible from the input $\mathcal{R}$ so that there exists a phylogenetic tree consistent with the remaining elements in $\mathcal{R}$ cannot be approximated within $c\cdot\ln n$ for some constant $c > 0$ in polynomial time, unless P = NP [5]. A variant of MRTC for *ordered* phylogenetic trees was studied in [9].

The unrooted analogue of a resolved triplet, called a *quartet* [22], is an unrooted tree with two internal nodes and four distinctly labeled leaves. The corresponding maximum quartets consistency problem is MAX SNP-hard [18,22], but the complete version of the problem admits a polynomial-time approximation scheme (PTAS) [18]. In an unpublished manuscript [17], we outlined how to obtain a similar PTAS for complete MRTC.

See the survey in Section 2 in [5] for references to other rooted triplets consistency-related results in the literature involving enumeration, ordered trees, phylogenetic networks, multi-labeled phylogenetic trees (MUL-trees), etc. Also, a related minimization problem asking for a tree having as few internal nodes as possible that is consistent with all of the resolved triplets in the input (when such a tree exists) was introduced in [16].

1.3. New results and organization of the article

First, Section 2 shows how any known f-approximation algorithm for MRTC can be applied to obtain an $\frac{f}{1+f}$-approximation algorithm for MTC. Plugging in the polynomial-time approximation algorithm for MRTC from [10] with $f = 1/3$ thus gives a polynomial-time 1/4-approximation algorithm for MTC.

Next, in Section 3, we extend Wu's exact exponential-time algorithm for MRTC [24] to MTC. We also let it take an additional parameter $k \geq 2$ as input and force the output to be a k-ary phylogenetic tree. The resulting algorithm runs in $O((k+1)^{n+1}\cdot(m+n))$ time. This may be $\Omega(n^n)$ if k is unrestricted, but the running time is single-exponential in n when $k = O(1)$ and we use this fact to design an exponential-time approximation scheme (ETAS) for unrestricted MTC. More precisely, for any constant $0 < \epsilon < 1$, the ETAS builds a phylogenetic tree satisfying at least a fraction of $(1-\epsilon)$ of the maximum number of input constraints satisfied by any phylogenetic tree in $O((\lceil\frac{12}{\epsilon}\rceil+1)^{n+1}\cdot(m+n))$ time.

The APX-hardness of MRTC [4] (and hence, MTC) effectively rules out the possibility of finding a polynomial-time approximation scheme (PTAS) for MTC. Nevertheless, in Section 4, we make further progress on the approximation status of

Algorithm 1

Input: A set S of n leaf labels and a set $\mathcal{R}$ of m triplet constraints over S.

Output: A phylogenetic tree whose leaf label set is S.

1. Let T_1 be a tree whose root has n children, each of them a leaf with a distinct label from S.
2. Extract the set $\mathcal{R}'$ of all triplet constraints of type 1 from $\mathcal{R}$ and apply any approximation algorithm for MRTC to $\mathcal{R}'$. While the tree is non-binary, select any internal node u with degree at least three and any two children c_1 and c_2 of u, create a new child v of u and let c_1 and c_2 become children of v instead. Let T_2 be the resulting tree.
3. Among T_1 and T_2, return one that satisfies the largest number of triplet constraints in $\mathcal{R}$.

Fig. 3. An approximation algorithm for MTC.

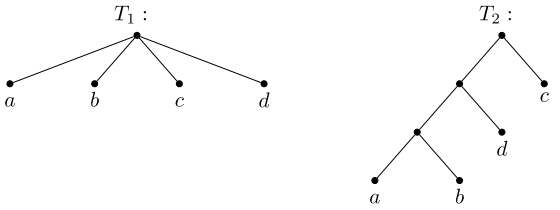

Fig. 4. Applying Algorithm 1 to the example from Section 1.1 with $S = \{a, b, c, d\}$ and $\mathcal{R} = \{ab|c,\ b|c|d,\ \neg(ab|d),\ \neg(ac|d)\}$. Step 1 constructs the tree T_1 on the left and step 2 constructs a binary tree T_2 such as the tree shown on the right. Here, step 3 will return T_1.

MTC by providing a PTAS for MTC restricted to complete instances, based on some key ideas from [18] and generalizing our unpublished work in [17].

As mentioned in Section 1.2, the restriction of MTC to constraints of type 1 (i.e., MRTC) is NP-hard [3,15,24]. On the other hand, MTC is trivially solvable in polynomial time when restricted to constraints of type 2 (just output a phylogenetic tree where all leaves are children of the root) or constraints of type 4 (output an arbitrary binary phylogenetic tree for the given leaf labels). Although MTC is easy for each of these two cases, it becomes NP-hard again when constraints of types 2 and 4 are allowed simultaneously, as proved in Section 5. We also describe a polynomial-time 1/2-approximation algorithm for this problem variant in Section 5.

Section 6 discusses how to adapt our algorithms to the weighted case, where nonnegative weights are assigned to the input triplet constraints and the objective is to construct a phylogenetic tree that maximizes the sum of the weights of the satisfied constraints. For our ETAS and our PTAS, we have to additionally assume that the ratio between the largest and the smallest constraint weights is bounded by a constant.

Finally, Section 7 states some open problems.

2. A polynomial-time 1/4-approximation algorithm for MTC

Suppose that an approximation algorithm for MRTC is available. We can use it as a subroutine to obtain an approximation algorithm for the more general maximum rooted triplets consistency problem (MTC) by applying the MRTC-approximation algorithm to the type-1 constraints in $\mathcal{R}$, arbitrarily refining all non-binary nodes to get a binary tree, and outputting the better solution among the tree thus obtained and the trivial tree consisting of a root node to which all leaf labels are directly attached. The details of the algorithm, from here on referred to as Algorithm 1, are given in Fig. 3 and an example is given in Fig. 4.

Theorem 1. *Algorithm 1 is an $\frac{f}{1+f}$-approximation algorithm for MTC, where f is the approximation ratio of the algorithm for MRTC used in step 2.*

Proof. To analyze the approximation ratio, partition $\mathcal{R}$ into $(\mathcal{R}_1, \mathcal{R}_2, \mathcal{R}_3, \mathcal{R}_4)$, where $\mathcal{R}_i$ for $i \in \{1, 2, 3, 4\}$ is the subset of $\mathcal{R}$ of all triplet constraints of type i. Let OPT denote the number of triplet constraints in $\mathcal{R}$ satisfied by an optimal solution for MTC on input $\mathcal{R}$. Similarly, let OPT' be the number of triplet constraints in $\mathcal{R}_1$ satisfied by an optimal solution for MRTC on input $\mathcal{R}_1$. There are two main cases:

1. $|\mathcal{R}_2 \cup \mathcal{R}_3| \geq \frac{f}{1+f} \cdot OPT$: The tree T_1 satisfies all input constraints of types 2 and 3, so in this case, the output satisfies at least $\frac{f}{1+f} \cdot OPT$ input constraints.

2. $|\mathcal{R}_2 \cup \mathcal{R}_3| < \frac{f}{1+f} \cdot OPT$: Since T_2 is an f-approximation for MRTC and T_2 is binary, T_2 satisfies at least $f \cdot OPT' + |\mathcal{R}_4|$ triplet constraints from $\mathcal{R}_1 \cup \mathcal{R}_4$. Now there are two subcases:
   (a) $OPT \geq (1+f) \cdot OPT'$: First observe that $|\mathcal{R}_1 \cup \mathcal{R}_4| > m - \frac{f}{1+f} \cdot OPT$, so $|\mathcal{R}_4| > m - |\mathcal{R}_1| - \frac{f}{1+f} \cdot OPT$. Next, since $OPT' + |\mathcal{R}_2| + |\mathcal{R}_3| + |\mathcal{R}_4| \geq OPT$ and $f \leq 1$, a bound on the number of input triplet constraints satisfied by T_2 is as follows: $f \cdot OPT' + |\mathcal{R}_4| > f \cdot OPT' + m - |\mathcal{R}_1| - \frac{f}{1+f} \cdot OPT = f \cdot OPT' + |\mathcal{R}_2| + |\mathcal{R}_3| + |\mathcal{R}_4| - \frac{f}{1+f} \cdot OPT \geq (f-1) \cdot OPT' + OPT - \frac{f}{1+f} \cdot OPT = \frac{1}{1+f} \cdot OPT - (1-f) \cdot OPT' \geq \frac{1}{1+f} \cdot OPT - \frac{1-f}{1+f} \cdot OPT = \frac{f}{1+f} \cdot OPT$.
   (b) $OPT < (1+f) \cdot OPT'$: Here, $f \cdot OPT' + |\mathcal{R}_4| > \frac{f}{1+f} \cdot OPT + |\mathcal{R}_4| \geq \frac{f}{1+f} \cdot OPT$. □

Employing the known polynomial-time approximation algorithm for MRTC from Section 5.1 of [10] gives $f = 1/3$ in Theorem 1 and thus immediately implies that MTC can be approximated within a ratio of 1/4 in polynomial time. Moreover, if a polynomial-time 1/2-approximation algorithm for MRTC is discovered in the future then Theorem 1 will give a polynomial-time 1/3-approximation algorithm for MTC.

We can actually say something even stronger in the particular case of $f = 1/3$ because Gąsieniec et al. [10] proved that the solution output by their polynomial-time 1/3-approximation algorithm for MRTC satisfies at least $|\mathcal{R}|/3$ triplet constraints in $\mathcal{R}$. Assuming that this algorithm is used in step 2 of Algorithm 1 above then in case $|\mathcal{R}_2 \cup \mathcal{R}_3| \geq \frac{1}{4} \cdot |\mathcal{R}|$, T_1 satisfies at least $|\mathcal{R}|/4$ triplet constraints from $\mathcal{R}$, while in case $|\mathcal{R}_2 \cup \mathcal{R}_3| < \frac{1}{4} \cdot |\mathcal{R}|$, T_2 satisfies at least $\frac{1}{3} \cdot \frac{3}{4} \cdot |\mathcal{R}| = \frac{1}{4} \cdot |\mathcal{R}|$ triplet constraints from $\mathcal{R}$.

Corollary 1. *In polynomial time, one can find an approximate solution to MTC satisfying at least $|\mathcal{R}|/4$ triplet constraints in $\mathcal{R}$.*

Corollary 1 implies that any optimal solution to MTC satisfies at least $|\mathcal{R}|/4$ triplet constraints in $\mathcal{R}$. This fact will be used in the proof of Theorem 5 in Section 4.2. Also observe that the bound in Corollary 1 is worst-case optimal in the sense that if $\mathcal{R}$ consists of the four triplet constraints $xy|z$, $xz|y$, $yz|x$, and $x|y|z$ for every three leaf labels $x, y, z \in S$, then at most $|\mathcal{R}|/4$ triplet constraints in $\mathcal{R}$ can be satisfied by any phylogenetic tree.

3. An ETAS for MTC

In this section, we develop an exponential-time approximation scheme (ETAS) for MTC. Section 3.1 describes a generalization of Wu's exact algorithm for MRTC [24], which is then utilized in Section 3.2 to obtain our ETAS. For this purpose, we introduce some additional notation. Let $(S, \mathcal{R})$ be an instance of MTC and consider any $U \subseteq S$. For any partition P of U with $|P| \geq 2$, let:

- $w_2(P)$ be the number of resolved triplets (constraints of type 1) $ab|c$ in $\mathcal{R}$ such that a and b belong to two different parts in P and $c \notin U$;
- $w_3(P)$ be the number of fan triplets (constraints of type 2) $a|b|c$ in $\mathcal{R}$ such that a, b, c belong to three different parts in P;
- $w_{f2}(P)$ be the number of forbidden resolved triplets (constraints of type 3) $\neg(ab|c)$ in $\mathcal{R}$ such that a and c belong to two different parts in P and $b \notin U$, or b and c belong to two different parts in P and $a \notin U$, or a, b, c belong to three different parts in P; and
- $w_{f3}(P)$ be the number of forbidden fan triplets (constraints of type 4) $\neg(a|b|c)$ in $\mathcal{R}$ such that two elements in $\{a, b, c\}$ belong to two different parts in P and the remaining one does not belong to U.

Suppose that T is a phylogenetic tree. Let $I(T)$ be the set of internal nodes in T. For every $v \in I(T)$, let T_v denote the subtree of T rooted at v, that is, the subtree of T induced by v and all proper descendants of v. Also, let π_v be the partition of $\Lambda(T_v)$ into $(\Lambda(T_{v_1}), \Lambda(T_{v_2}), \ldots, \Lambda(T_{v_\ell}))$, where $v_1, v_2, \ldots, v_\ell$ are the children of v. With this notation, we can write:

Lemma 1. *For any phylogenetic tree T with $\Lambda(T) = S$, the number of constraints in $\mathcal{R}$ satisfied by T is equal to $\sum_{v \in I(T)} (w_2(\pi_v) + w_3(\pi_v) + w_{f2}(\pi_v) + w_{f3}(\pi_v))$.*

Proof. Consider any constraint C in $\mathcal{R}$ involving three leaf labels $a, b, c \in S$. We show the following equivalence: C is satisfied by T if and only if there is a unique node v in T such that (*) the partition π_v of $\Lambda(T_v)$ satisfies the conditions on $\{a, b, c\}$ in the definition of $w_l(\pi_v)$, where l is equal to 2, 3, $f2$, or $f3$ depending on if C is of type 1, 2, 3, or 4, respectively.

First suppose that C is satisfied by T. For $l = 2$, the unique node v in T for which (*) holds is $lca^T(a, b)$. For $l = 3$, the unique node v in T for which (*) holds is $lca^T(a, b) = lca^T(a, c) = lca^T(b, c)$. For $l = f2$, the unique node v in T for which (*) holds is either $lca^T(a, c)$ or $lca^T(b, c)$ or $lca^T(a, b) = lca^T(a, c) = lca^T(b, c)$. For $l = f3$, the unique node v in T for which (*) holds is either $lca^T(a, b)$ or $lca^T(a, c)$ or $lca^T(b, c)$.

For the other direction, it is immediate that if the partition π_v for some node v satisfies the condition in the definition of $w_l(\pi_v)$ with respect to $\{a, b, c\}$ then C is satisfied by T. □

3.1. A generalization of Wu's algorithm

We first generalize Wu's dynamic programming-based algorithm for MRTC [24], which always outputs a *binary* phylogenetic tree, to MTC by allowing it to construct k-ary phylogenetic trees. (Recall that for any integer $k \geq 2$, a k-ary phylogenetic tree is a phylogenetic tree in which every internal node has at most k children). We also extend Wu's algorithm to take into account constraints of types 2, 3, and 4 in the input.

As in Wu's algorithm, all subsets of S are considered in order of nondecreasing cardinality. For each such $U \subseteq S$, the new algorithm computes and stores a value $score(U)$, defined as follows. For any singleton U, define $score(U)$ to be 0. For every non-singleton subset U of S, define $score(U)$ recursively by $score(U) = \max_{\ell=2}^{k}\{score_\ell(U)\}$, where for every integer $\ell \geq 2$:

$$score_\ell(U) = \max_{\ell-\text{partition } (U_1,\ldots,U_\ell) \text{ of } U} \left\{ \sum_{i=1}^{\ell} score(U_i) + \sum_{j=2}^{3} (w_j(U_1,\ldots,U_\ell) + w_{fj}(U_1,\ldots,U_\ell)) \right\}.$$

After computing $score(U)$ for every $U \subseteq S$, the algorithm recovers an optimal k-ary phylogenetic tree in a traceback step, starting at $U = S$ and always picking an ℓ-partition of the current subset U yielding the maximum value of $score(U)$. The corresponding node in the constructed tree gets ℓ children in one-to-one correspondence with the subsets of U that form the selected partition. Note that in the special case where $k = 2$ and $w_3(P) = w_{f2}(P) = w_{f3}(P) = 0$ for every partition P of every $U \subseteq S$, the new algorithm is identical to Wu's algorithm.

The next lemma shows that the expression $w_2(\pi_v) + w_3(\pi_v) + w_{f2}(\pi_v) + w_{f3}(\pi_v)$ in Lemma 1 can be computed efficiently.

Lemma 2. *Given a partition P of $U \subseteq S$, the values of $w_2(P)$, $w_3(P)$, $w_{f2}(P)$, and $w_{f3}(P)$ can be computed in $O(m+n)$ time, where $m = |\mathcal{R}|$ and $n = |S|$.*

Proof. Let $\ell = |P|$. Color the elements in U with ℓ colors according to P, and assign another color to the elements in $S \setminus U$. Then, for each constraint in $\mathcal{R}$, check the colors of its elements to determine whether or not it increases $w_2(P)$, $w_3(P)$, $w_{f2}(P)$, or $w_{f3}(P)$ by one. □

Theorem 2. *Let $k \geq 2$ be a given integer. For any instance of MTC, Wu's generalized algorithm outputs a k-ary phylogenetic tree T with $\Lambda(T) = S$ that maximizes the number of satisfied triplet constraints in $\mathcal{R}$ among all k-ary phylogenetic trees using $O((k+1)^{n+1} \cdot (m+n))$ time.*

Proof. It follows by induction on $|U|$ and Lemma 1 that $score(U)$ equals the maximum number of input triplet constraints that can be satisfied in any k-ary phylogenetic tree leaf-labeled by U. Therefore, the output of the algorithm is correct.

There are $\binom{n}{q}$ subsets U of S with q elements, and the number of ℓ-partitions of a subset U with $|U| = q$ is less than ℓ^q. Thus, the total number of partitions processed by our algorithm is at most $\sum_{q=1}^{n}\binom{n}{q} \cdot \sum_{\ell=2}^{k} \ell^q \leq \sum_{\ell=2}^{k}(\ell+1)^n \leq (k+1)^{n+1}$ by binomial expansion. According to Lemma 2, for any given partition P of $U \subseteq S$, the values of $w_2(P)$, $w_3(P)$, $w_{f2}(P)$, and $w_{f3}(P)$ can be computed in $O(m+n)$ time. We conclude that the algorithm runs in $O((k+1)^{n+1} \cdot (m+n))$ time. □

Remark 2. When $|P| = 2$ in Lemma 2, $w_2(P)$ is the same as $w(V_1, V_2)$ in Wu's exact algorithm for MRTC [24]. Theorem 2 in [24] computes $w(V_1, V_2)$ in $O(m+n^2)$ time, so using our Lemma 2 instead slightly improves the running time of Wu's original algorithm from $O(3^n \cdot (m+n^2))$ to $O(3^n \cdot (m+n))$.

3.2. The ETAS

We now analyze how much is lost by forcing the solution to an instance of MTC to be a k-ary phylogenetic tree, where k is any integer such that $k \geq 13$. See Fig. 5 for a simplified example.

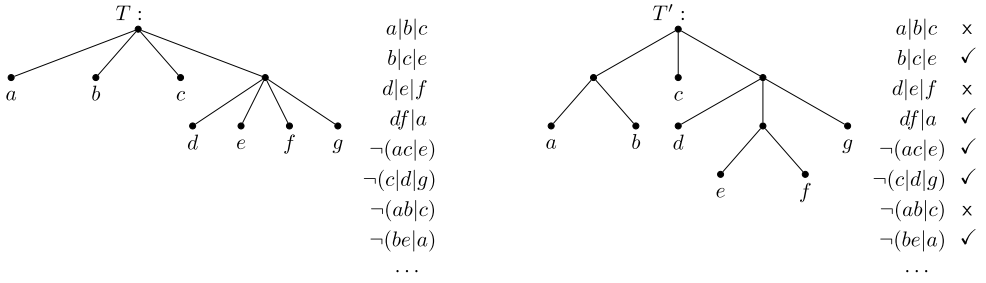

Fig. 5. Illustrating Theorem 3. The phylogenetic tree T on the left can be approximated by the k-ary phylogenetic tree T' on the right, where $k = 3$, while still satisfying many of the specified constraints. Note that the theorem concerns the case $k \geq 13$, but the idea is the same as shown here.

Theorem 3. *For any integer $k \geq 13$ and any phylogenetic tree T, there exists a k-ary phylogenetic tree T' on the same set of leaf labels, i.e., with $\Lambda(T') = \Lambda(T)$, that satisfies at least a fraction of $(1 - 12/k)$ of the input triplet constraints satisfied by T.*

Proof. For the sake of the proof, assign to each forbidden resolved triplet $\neg(ab|c)$ in $\mathcal{R}$ contributing to $w_{f2}(\pi_v)$ for some node v in T, either the resolved triplet $ac|b$ or the resolved triplet $bc|a$ or the fan triplet $a|b|c$, depending on which of these three constraints is satisfied in T. Similarly, assign to each forbidden fan triplet $\neg(a|b|c)$ in $\mathcal{R}$ contributing to $w_{f3}(\pi_v)$, either the resolved triplet $ab|c$ or the resolved triplet $ac|b$ or the resolved triplet $bc|a$, depending on the structure of T. Let $h_2(\pi_v)$ be the cardinality of the multiset of assigned resolved triplets and let $h_3(\pi_v)$ be the cardinality of the multiset of assigned fan triplets. Then $w_{f2}(\pi_v) + w_{f3}(\pi_v) = h_2(\pi_v) + h_3(\pi_v)$, so the formula in Lemma 1 becomes $\sum_{v \in I(T)}(w_2(\pi_v) + w_3(\pi_v) + h_2(\pi_v) + h_3(\pi_v))$.

We give a probabilistic argument to show that there exists a T' satisfying the theorem. In bottom-up order, consider every node v in T having degree larger than k. Let $v_1, v_2, \ldots, v_\ell$ be the children of v, where $\ell > k$. Partition $\{v_1, v_2, \ldots, v_\ell\}$ into k subsets uniformly at random. For every fan triplet $a|b|c$ contributing to $w_3(\pi_v)$ (i.e., having each of its elements in a distinct T_{v_i}) or to $h_3(\pi_v)$ (i.e., being assigned to a forbidden resolved triplet), define an indicator random variable $X_{a,b,c}$ whose value is 1 if and only if $\{a, b, c\}$ fall into three different subsets. The probability that any two elements in $\{a, b, c\}$ fall into the same subset is upper bounded by $1/k + 2/k = 3/k$, so the expected value of $X_{a,b,c}$ is at least $1 - 3/k$. By linearity of expectation, the expected number of fan triplets contributing to $w_3(\pi_v) + h_3(\pi_v)$ and having their elements in three different subsets is at least a fraction of $(1 - 3/k)$ of all such fan triplets. Hence, there exists a partition P of $\{v_1, v_2, \ldots, v_\ell\}$ into k subsets such that a fraction of at least $(1 - 3/k)$ of the fan triplets contributing to $w_3(\pi_v) + h_3(\pi_v)$ have their elements in three different subsets. For each $p \in P$, define an arbitrary rooted binary tree F_p whose leaves are labeled by the children of v belonging to p and then replace each leaf v_i in F_p by the subtree T_{v_i}. Next, delete the edges in T connecting v to its children and instead attach the root of F_p for every $p \in P$ as a child of v. Observe that a fan triplet that contributes to $w_3(\pi_v)$ may also contribute up to three times to $h_3(\pi_v)$ because it may have been assigned to up to three forbidden resolved triplets. It follows that the sum of the new values of $w_3(\pi_v) + h_3(\pi_v)$ is at least $(1 - 4 \cdot 3/k)$ of the sum of its previous values. Let T' be the tree obtained after treating all nodes of T in bottom-up order.

In turn, consider any resolved triplet $ab|c$ that contributes to $w_2(\pi_v)$ or $h_2(\pi_v)$ for some node v in T. Then $v = lca^T(a, b)$ is a proper descendant of $lca^T(a, c)$, and c is not a descendant of v. By the definition of T', $lca^{T'}(a, b)$ is still a proper descendant of $lca^{T'}(a, c)$, which means that $ab|c$ is also satisfied in T'. Thus, the sum of $w_2(\pi_v) + h_2(\pi_v)$ over all internal nodes v in T is the same as the sum of $w_2(\pi_v) + h_2(\pi_v)$ over all internal nodes v in T'. The theorem follows from Lemma 1. □

Motivated by Theorem 3, one can try to find a good approximate solution for MTC by constructing a k-ary phylogenetic tree satisfying the maximum number of input triplet constraints over all k-ary phylogenetic trees, for some suitable value of k. Indeed, combining Theorems 2 and 3 gives an ETAS for MTC:

Theorem 4. *For any instance of MTC and any specified constant $0 < \epsilon < 1$, one can build a phylogenetic tree satisfying at least a fraction of $(1-\epsilon)$ of the maximum number of triplet constraints in $\mathcal{R}$ satisfied by any phylogenetic tree in $O((\lceil \frac{12}{\epsilon} \rceil + 1)^{n+1} \cdot (m+n))$ time.*

Proof. Select $k = \lceil \frac{12}{\epsilon} \rceil$ (≥ 13) and apply Wu's generalized algorithm from Section 3.1. According to Theorem 2, this gives an optimal k-ary phylogenetic tree. By Theorem 3, this tree satisfies at least a fraction of $(1 - 12/k) \geq (1 - \epsilon)$ of the number of input triplet constraints satisfied by an optimal, degree-unrestricted phylogenetic tree. □

4. A PTAS for complete MTC

This section presents a polynomial-time approximation scheme (PTAS) for MTC restricted to complete instances, obtained by modifying the PTAS of Jiang et al. in [18] for the unrooted analogue of complete MRTC. In short, their PTAS works by constructing many instances of the *label-to-bin assignment problem* that are solved approximately by applying a PTAS of Arora et al. [2] for smooth polynomial integer programs. For complete MTC, because of the type-2 constraints in the input, the approach of [18] has to be generalized to also consider *non-binary* trees, which makes it a little more complicated to prove the existence of a so-called decomposition tree that provides a sufficiently good approximation. Furthermore, a more intricate smooth polynomial integer program is needed to be able to represent input constraints of types 2, 3, and 4. The details are given below. For completeness, we repeat certain key arguments from [18] in adapted form.

4.1. Preliminaries

Let $k \geq 3$ be a positive integer. Similarly to [18], we first show that any rooted phylogenetic tree T has a *decomposition tree* T_k, here defined as a rooted, unordered tree whose leaves are distinctly labeled by $\Lambda(T)$ and with the following properties:

- For each internal node u in T_k, either all children of u are leaves (in this case, u is called a *bin root*) or all children of u are internal nodes.
- The number of bin roots in T_k is at most k and the number of leaves attached to each bin root is at most $8n/k$.

Algorithm 2

Input: A phylogenetic tree T and a positive integer $k \geq 3$.

Output: A decomposition tree T_k for T.

1. Traverse T, and for every node u visited, check if the subtree $T(u)$ of T rooted at u contains at most $8n/k$ leaves. If so, u is designated as a bin root (unless u is a leaf), and all internal edges of $T(u)$ except for those incident to a leaf are contracted so that $T(u)$ becomes a tree of height 1. If $T(u)$ has more than $8n/k$ leaves, continue traversing T at a child of u.
2. For any single leaf ℓ that is not yet attached to a bin root, subdivide the edge between ℓ and its parent to create a new bin root associated with ℓ.
3. For each small bin root b, do the following while b is small and b has a sibling b' that is also a small bin root: make all children of b' become children of b and delete b'.
4. For any node u in T, let $p(u)$ denote the parent of u. For each small bin root b, do the following while b is small, $p(b)$ has at most two children, $p(b)$ has exactly one sibling p', and p' is a small bin root: make all children of p' become children of b, delete p', and contract the edge between $p(b)$ and $p(p(b))$.
5. Return T.

Fig. 6. An algorithm for constructing a decomposition tree T_k for a phylogenetic tree T.

- For every $\{a, b, c\} \subseteq \Lambda(T)$ attached to three different bin roots in T_k, it holds that any triplet constraint of type 1, 2, 3, or 4 over $\{a, b, c\}$ is satisfied by T if and only if it is satisfied by T_k.

In particular, note that the last property guarantees that if one replaces T by T_k then every triplet constraint over three leaf labels attached to three different bin roots is preserved. For any phylogenetic tree T, let $\gamma(T)$ be the set of all constraints of type 1, 2, 3, or 4 over $\Lambda(T)$ that are satisfied by T. The next lemma, corresponding to Lemma 2.4 in [18], shows that a decomposition tree for T satisfies many of the constraints in $\gamma(T)$ and therefore motivates the definition above.

Lemma 3. *Suppose that T is an optimal solution to an instance of MTC and that T_k is a decomposition tree for T. The number of triplet constraints from $\mathcal{R}$ satisfied by T_k is $|\gamma(T_k) \cap \mathcal{R}| \geq |\gamma(T) \cap \mathcal{R}| - \frac{122}{k} \cdot n^3$.*

Proof. Every triplet constraint in $\gamma(T) \setminus \gamma(T_k)$ has at least two of its three leaves attached to the same bin root in T_k (otherwise, it belongs to both $\gamma(T)$ and $\gamma(T_k)$ by the definition of T_k). The number of such triplet constraints is at most $k \cdot (8n/k)^2 \cdot \frac{1}{2} \cdot n + k \cdot (8n/k)^3 \cdot \frac{1}{6} < \frac{61}{k} \cdot n^3$ since $k \geq 3$, and for each one, the input $\mathcal{R}$ contains at most two triplet constraints that are satisfied by T but not by T_k. Hence, $|\gamma(T_k) \cap \mathcal{R}| \geq |\gamma(T) \cap \mathcal{R}| - \frac{122}{k} \cdot n^3$. □

A bin root is called *small* if $\leq 4n/k$ leaves are attached to it, and *large* otherwise. To prove that T_k always exists, consider the algorithm in Fig. 6 which constructs a decomposition tree for any given phylogenetic tree. It is a variant of Algorithm k-Bin Decomposition in [18]. The difference between the old algorithm and our new one is that the rule for how to merge small bin roots has been expanded to work for non-binary trees. Below, the new algorithm is named Algorithm 2.

Lemma 4. *Algorithm 2 outputs a decomposition tree for T having at most k bin roots.*

Proof. To derive an upper bound on the number of bin roots, let s and l denote the number of small and large bin roots in the output tree, respectively. For any node u in the output tree, denote the number of small and large bin roots in the subtree rooted at u by s_u and l_u, and let $Child(u)$ be the set of children of u. Due to step 3, $Child(u)$ contains at most one small bin root for every node u.

We use induction on the internal nodes' heights to show that $s \leq 3l - 2$ (cf. the proof of Lemma 2.3 in [18]). First of all, the inequality is true for the subtree rooted at any internal node u at height 2 (the base case) since $Child(u)$ consists of at most one small bin root and at least one large bin root, giving $s \leq 1$ and $l \geq 1$ and thus $s \leq 3l - 2$. Next, in the general case, u is an internal node at height ≥ 3 and one of the following situations holds, where $C = Child(u) \setminus \{v : v \text{ is a small bin root}\}$.

- $|C| \geq 2$: Then the inequality follows directly from the induction hypothesis after using the fact that at most one child of u is a small bin root and writing $s_u \leq (\sum_{v \in C} s_v) + 1 \leq \sum_{v \in C} (3l_v) - 2|C| + 1 < 3l_u - 2$.
- $|C| = 1$: Then $Child(u)$ consists of a small bin root a and a node v of height at least 2. See Fig. 7. The set $Child(v)$ contains at least two elements but it is not possible for it to consist of a small bin root b and exactly one other node because step 4 would have merged a and b in this case. Thus, $|D| \geq 2$, where $D = Child(v) \setminus \{w : w \text{ is a small bin root}\}$. By the induction hypothesis, $s_u \leq (\sum_{w \in D} s_w) + 1 + 1 \leq \sum_{w \in D} (3l_w) - 2|D| + 2 \leq 3l_u - 2$.

By the principle of mathematical induction, $s \leq 3l - 2$. Finally, by definition, each large bin root has at least $4n/k$ leaves, so $l \leq k/4$. This means that the total number of bin roots is $s + l < 3l + l \leq k$.

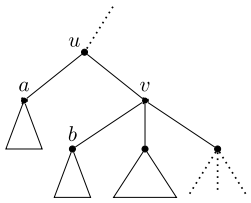

Fig. 7. In the case $|C| = 1$ in the proof of Lemma 4, $Child(u) = \{a, v\}$, where a is a small bin root and v is a node of height at least 2. If $Child(v)$ contains a small bin root b then $|Child(v)| \geq 3$ because of step 4 in Algorithm 2. The siblings of b may be large bin roots or non-bin roots.

It is straightforward to verify that the output of the algorithm has all of the other properties required to be a decomposition tree. □

4.2. The PTAS

By Lemma 4, an optimal solution T^{opt} to any given instance $(S, \mathcal{R})$ of MTC has a decomposition tree T_k^{opt} for any positive integer $k \geq 3$. Moreover, by Lemma 3, if one could find T_k^{opt} then it would be an approximation, whose quality depends on k, of T^{opt}. Importantly, removing all leaves from T_k^{opt} yields an unlabeled tree with k leaves corresponding to the k bin roots, and this tree can be found in polynomial time by enumeration when k is constant. Based on this observation, the PTAS of Jiang et al. [18] modified to MTC for complete instances works as follows.

Let $(S, \mathcal{R})$ be a complete instance of MTC and $0 < \epsilon < 1$ a specified constant. Generate all unlabeled, unordered rooted trees having k leaves and no degree-1 nodes, where $k \geq 3$ is a positive integer that depends on $1/\epsilon$ (how to set k will be addressed later). For each such tree K, henceforth called a *kernel tree*, interpret the leaves of K as initially empty bin roots and try to attach the leaf labels in S to bin roots so that as many triplet constraints as possible from $\mathcal{R}$ are satisfied. The latter problem is formalized as the *label-to-bin assignment problem* (LBA): given a set S of leaf labels, a set $\mathcal{R}$ of triplet constraints on S, and a kernel tree K with k bin roots, attach the elements of S to the bin roots of K so that each bin root gets at most $8n/k$ elements and the maximum number of triplet constraints in $\mathcal{R}$ are satisfied. Solve each instance of LBA approximately by applying the PTAS of Arora et al. [2] for smooth polynomial integer programs. Finally, among all the found approximate LBA-solutions, output one that is a best solution to MTC.

Observe that there are two separate approximations involved here: approximating an optimal solution to the given instance of MTC by a decomposition tree for the given k (Lemma 3) and approximating an optimal decomposition tree, represented as an instance of LBA, by using Arora et al.'s PTAS (Lemma 5 below).

We now explain how to solve LBA approximately. Jiang et al. [18] demonstrated that although LBA is NP-hard even when $\mathcal{R}$ consists of type-1 constraints only, it admits a PTAS in this setting, obtained by formulating the problem as a smooth polynomial integer program and applying a PTAS by Arora et al. [2]. We need to handle more types of constraints, so we modify their formulation of LBA accordingly. Let $\gamma(K)$ be the set of all constraints of type 1, 2, 3, or 4 over the bin roots that are satisfied by K. Introduce a 0/1-variable x_{sb} for every $s \in S$ and every bin root b in K; in the final solution, $x_{sb} = 1$ if and only if s is attached to b. First, for every resolved triplet $ab|c$ in $\mathcal{R}$, define the polynomial:

$$p_{ab|c}(x) = \sum_{ij|k \in \gamma(K)} x_{ai}x_{bj}x_{ck} + x_{bi}x_{aj}x_{ck}$$

Next, for every fan triplet $a|b|c$ in $\mathcal{R}$, define the following polynomial, where $Permute(a, b, c)$ stands for the set of all bijections from $\{a, b, c\}$ to $\{a, b, c\}$:

$$p_{a|b|c}(x) = \sum_{i|j|k \in \gamma(K)} \sum_{\delta \in Permute(a,b,c)} x_{\delta(a)i}x_{\delta(b)j}x_{\delta(c)k}$$

For every forbidden resolved triplet $\neg(ab|c)$ in $\mathcal{R}$, define the polynomial:

$$p_{\neg(ab|c)}(x) = p_{ac|b}(x) + p_{bc|a}(x) + p_{a|b|c}(x)$$

In the same way, for every forbidden fan triplet $\neg(a|b|c)$ in $\mathcal{R}$, define the polynomial:

$$p_{\neg(a|b|c)}(x) = p_{ab|c}(x) + p_{ac|b}(x) + p_{bc|a}(x)$$

Finally, define:

$$p(x) = \sum_{ab|c \in \mathcal{R}} p_{ab|c}(x) + \sum_{a|b|c \in \mathcal{R}} p_{a|b|c}(x) + \sum_{\neg(ab|c) \in \mathcal{R}} p_{\neg(ab|c)}(x) + \sum_{\neg(a|b|c) \in \mathcal{R}} p_{\neg(a|b|c)}(x)$$

This way, $p(x)$ counts the number of input triplet constraints satisfied by a 0/1-assignment to the x_{sb}-variables. To ensure that each leaf label is attached to exactly one bin root and that no bin root gets too many leaf labels, the statement of LBA becomes:

Maximize $p(x)$ subject to

$x_{sb} \in \{0, 1\}$ for each $s \in S$ and bin root b,

$\sum_{b=1}^{k} x_{sb} = 1$ for each $s \in S$, and

$\sum_{s=1}^{n} x_{sb} \leq 8n/k$ for each bin root b.

The resulting polynomial integer program is an $O(1)$-*smooth degree*-3 *polynomial integer program* according to the following definition from [2]: *An $O(1)$-smooth degree-d polynomial integer program is to maximize $p(x_1, \ldots, x_n)$ subject to $x_i \in \{0, 1\}$ for all $1 \leq i \leq n$, where $p(x_1, \ldots, x_n)$ is a degree-d polynomial in which the absolute value of the coefficient of each degree-i term is $O(n^{d-i})$.* Arora et al.'s PTAS for smooth polynomial integer programs [2] can thus be applied, after the minor modification described next. The original PTAS solves the fractional version of the problem and then rounds the obtained fractional value for each variable individually in order to obtain an integer solution. However, this does not work here because of the condition $\sum_{b=1}^{k} x_{sb} = 1$ for each $s \in S$. Following [18], we instead set $x_{sb} = 1$ and $x_{sa} = 0$ for all $a \neq b$ with probability equal to the fractional value for x_{sb} so that exactly one of $x_{s1}, \ldots, x_{sk}$ is 1 and the rest 0. The method can be derandomized as in [2]. In analogy to Theorem 2.6 in [18], we have:

Lemma 5. *For every constant $0 < \epsilon' < 1$, there is a polynomial-time algorithm that for any instance of LBA specified by a set S of leaf labels, a set $\mathcal{R}$ of triplet constraints for MTC over S, and a kernel tree K, produces a phylogenetic tree T' with $|\gamma(T') \cap \mathcal{R}| \geq |\gamma(T) \cap \mathcal{R}| - \epsilon' \cdot n^3$, where T is an optimal LBA solution.*

In summary, we obtain the next theorem, which states that the method outputs a phylogenetic tree satisfying at least a fraction of $(1 - \epsilon)$ of the maximum possible number of input constraints and that it runs in polynomial time as long as $|\mathcal{R}| = \Omega(n^3)$.

Theorem 5. *For any specified constants $0 < \epsilon < 1$ and $c > 0$, there is a polynomial-time algorithm for MTC restricted to instances with $|\mathcal{R}| \geq c \cdot n^3$ that produces a tree T'' approximating an optimal solution T^{opt} in such a way that $|\gamma(T'') \cap \mathcal{R}| \geq (1 - \epsilon) \cdot |\gamma(T^{opt}) \cap \mathcal{R}|$.*

Proof. Choose any constant integer $k \geq \frac{976}{\epsilon \cdot c}$ and any constant $0 < \epsilon' \leq \frac{\epsilon \cdot c}{8}$, and apply the method just described in this subsection to obtain T''. That is, for each possible kernel tree K with at most k bin roots, find an approximate LBA-solution by using the modified PTAS of Arora et al., and then, among all the approximate solutions found, let T'' be one giving a best solution to MTC. Since k is constant, the running time is polynomial in the input size.

First note that by Corollary 1, $|\gamma(T^{opt}) \cap \mathcal{R}| \geq \frac{1}{4} \cdot |\mathcal{R}|$. Thus, $n^3 \leq \frac{4}{c} \cdot |\gamma(T^{opt}) \cap \mathcal{R}|$.

Next, let T_k^{opt} be a decomposition tree for T^{opt}. By Lemma 5, $|\gamma(T'') \cap \mathcal{R}| \geq |\gamma(T_k^{opt}) \cap \mathcal{R}| - \epsilon' \cdot n^3$, and by Lemma 3, $|\gamma(T_k^{opt}) \cap \mathcal{R}| \geq |\gamma(T^{opt}) \cap \mathcal{R}| - \frac{122}{k} \cdot n^3$. This yields $|\gamma(T'') \cap \mathcal{R}| \geq |\gamma(T^{opt}) \cap \mathcal{R}| - \frac{122}{k} \cdot n^3 - \epsilon' \cdot n^3 \geq |\gamma(T^{opt}) \cap \mathcal{R}| \cdot (1 - \frac{122}{k} \cdot \frac{4}{c} - \epsilon' \cdot \frac{4}{c}) \geq |\gamma(T^{opt}) \cap \mathcal{R}| \cdot (1 - \frac{488 \cdot \epsilon \cdot c}{976 \cdot c} - \frac{\epsilon \cdot c \cdot 4}{8 \cdot c}) = |\gamma(T^{opt}) \cap \mathcal{R}| \cdot (1 - \epsilon)$. □

In particular, Theorem 5 implies a PTAS for complete MTC:

Corollary 2. *For any specified constant $0 < \epsilon < 1$, there is a polynomial-time algorithm for MTC restricted to complete instances that produces a tree T'' approximating an optimal solution T^{opt} in such a way that $|\gamma(T'') \cap \mathcal{R}| \geq (1 - \epsilon) \cdot |\gamma(T^{opt}) \cap \mathcal{R}|$.*

Proof. Since the instance is complete, $|\mathcal{R}| = \binom{n}{3} \geq \frac{1}{7} \cdot n^3$, assuming without loss of generality that $n \geq 21$. Set $c = \frac{1}{7}$ and apply Theorem 5. □

5. The complexity of MTC restricted to constraints of types 2 and 4

Here, we study the computational complexity of another special case of MTC, namely the restriction of MTC to triplet constraints of type 2 and type 4.

Recall from the literature that the NP-hard MAX CUT problem (see, e.g., [8]) takes as input an undirected graph $G = (V, E)$ and asks for a partition of V into two disjoint subsets (V_1, V_2) (any such partition is called a *cut*) that maximizes the number of edges in E having one endpoint in V_1 and one endpoint in V_2 (this is called the *size* of the cut).

Theorem 6. *MTC is NP-hard, even if restricted to constraints of types 2 and 4.*

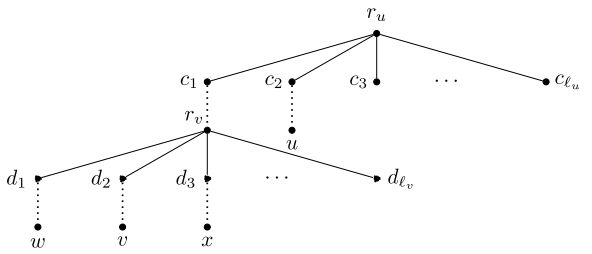

Fig. 8. The construction in point 2. in the proof of Theorem 6. Dotted lines represent zero or more edges.

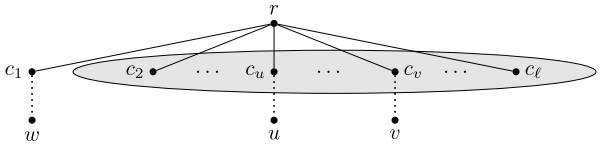

Fig. 9. The construction in point 3 in the proof of Theorem 6. The nodes c_u and c_v belong to the set $\{c_2, \ldots, c_\ell\}$.

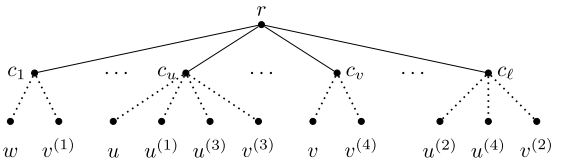

Fig. 10. Illustrating the definition of Q_i. In this example, $Q_u = \{u^{(2)}, u^{(4)}\}$ and $Q_v = \{v^{(2)}, v^{(3)}\}$.

Proof. We give a polynomial-time reduction from MAX CUT. Let $G = (V, E)$ be any given instance of MAX CUT and denote $V = \{v_1, v_2, \ldots, v_p\}$ and $q = |E|$. Define a set of $p(2q + 1) + 1$ leaf labels by $S = V \cup \{w\} \cup \{v^{(1)}, v^{(2)}, \ldots, v^{(2q)} : v \in V\}$; henceforth, leaf labels of the form $v^{(i)}$ are referred to as *copies of* v. Let $\mathcal{R} = \mathcal{R}_A \cup \mathcal{R}_B \cup \mathcal{R}_C$, where:

- $\mathcal{R}_A = \{u|v|w : \{u, v\} \in E\}$ [type 2],
- $\mathcal{R}_B = \{v|v^{(i)}|w : v \in V \text{ and } 1 \leq i \leq 2q\}$ [type 2],
- $\mathcal{R}_C = \{\neg(x|y|z) : x, y, z \in S \setminus \{w\}\}$ [type 4].

For convenience, write $\mathcal{R}' = \mathcal{R}_B \cup \mathcal{R}_C$ and $f = |\mathcal{R}'|$.

Suppose T is a phylogenetic tree with $\Lambda(T) = S$ that satisfies at least f constraints in $\mathcal{R}$. Since $|\mathcal{R}_A| = q$, T satisfies at least $f - q$ constraints in $\mathcal{R}'$. For every $v \in V$, define $r_v = lca^T(v, w)$. We first show that T must have a specific branching structure.

1. The degree of r_v, where $v \in V$, cannot be equal to 2 because if so, $2q$ constraints in $\mathcal{R}_B$ of the form $v|v^{(i)}|w$ cannot be satisfied, contradicting that T satisfies at least $f - q$ constraints in $\mathcal{R}'$. Thus, the degree of r_v is at least 3 for every $v \in V$.
2. Suppose that $r_u \neq r_v$ for some $u, v \in V$ and that r_v is a proper descendant of r_u. Denote the children of r_u by $c_1, c_2, \ldots, c_{\ell_u}$ and the children of r_v by $d_1, d_2, \ldots, d_{\ell_v}$. By point 1., $\ell_u, \ell_v \geq 3$. Assume without loss of generality that c_1 and d_1 are ancestors of w and that c_2 and d_2 are ancestors of u and v, respectively. Let x be any leaf that is a descendant of d_3. For an illustration, see Fig. 8. At least q of the constraints of the form $u|u^{(i)}|w$ in $\mathcal{R}_B$ have to be satisfied, so at least q of the copies of u are leaves of one or more of the subtrees of T rooted at $c_3, \ldots, c_{\ell_u}$. This implies that at least q constraints of the form $\neg(u|u^{(i)}|v)$ as well as at least q constraints of the form $\neg(u|u^{(i)}|x)$ from $\mathcal{R}_C$ are not satisfied, which is impossible. We conclude that $r_u = r_v$ for every $u, v \in V$; in the rest of the proof, this node is denoted by r.
3. The degree of r (defined in point 2.) is 3. To prove the claim, let $c_1, c_2, \ldots, c_\ell$ be the children of r, where c_1 is the ancestor of w, and suppose that $\ell \geq 4$. Consider any two $u, v \in V$. According to point 2., $lca^T(u, w) = lca^T(v, w) = r$, so c_1 cannot be an ancestor of u or v. Let $c_u \in \{c_2, \ldots, c_\ell\}$ be the ancestor of u and $c_v \in \{c_2, \ldots, c_\ell\}$ the ancestor of v, possibly with $c_u = c_v$. For $i \in \{u, v\}$, denote the set of all copies of i that belong to the subtrees rooted at $\{c_2, \ldots, c_\ell\} \setminus \{c_i\}$ by Q_i. See Figs. 9 and 10. As in point 2., $|Q_u| \geq q$ and $|Q_v| \geq q$ hold. Now, for any allocation of at least $2q + 2$ balls (corresponding to $\{u, v\} \cup Q_u \cup Q_v$) to at least 3 bins (corresponding to $c_2, \ldots, c_\ell$) so that no bin is left empty, there are at least $q + 1$ ways of selecting a triple of balls, one from each bin. Each such triple corresponds to a

constraint in $\mathcal{R}_C$ that will not be satisfied, contradicting that T satisfies $f - q$ constraints in $\mathcal{R}'$. Thus, the degree of r is 3.

Note that r may or may not be the root of T. Also note that it does not matter where in T the (at most q) copies of u not belonging to Q_u are located. The crucial point is that w belongs to one of the subtrees rooted at the three children of r and that each leaf label in V belongs to one of the other two such subtrees.

Next, we shall show that G has a cut of size at least k if and only there exists a phylogenetic tree T with $\Lambda(T) = S$ that satisfies at least $k + f$ constraints in $\mathcal{R}$.

($\Rightarrow$) Let (V_1, V_2) be a cut of G of size at least k. Construct two arbitrary binary phylogenetic trees T_1 and T_2 with $\Lambda(T_1) = V_1 \cup \{v^{(1)}, v^{(2)}, \ldots, v^{(2q)} : v \in V_2\}$ and $\Lambda(T_2) = V_2 \cup \{v^{(1)}, v^{(2)}, \ldots, v^{(2q)} : v \in V_1\}$. Let T be the phylogenetic tree consisting of a root node with three children: a leaf labeled w, the root of T_1, and the root of T_2. Clearly, $\Lambda(T) = S$ and T satisfies all f constraints in $\mathcal{R}'$. In addition, for every $\{u, v\} \in E$ that contributes to the size of the cut (V_1, V_2), T satisfies the constraint $u|v|w$ in $\mathcal{R}_A$. In total, T satisfies at least $k + f$ constraints in $\mathcal{R}$.

($\Leftarrow$) Let T be a phylogenetic tree with $\Lambda(T) = S$ that satisfies at least $k + f$ constraints in $\mathcal{R}$. Since at least f constraints are satisfied, T has the branching structure described in points 1., 2., and 3. above. Denote the three subtrees of T rooted at the children of r by T_1, T_2, T_3, where $w \in \Lambda(T_1)$ and $v \in \Lambda(T_2) \cup \Lambda(T_3)$ for every $v \in V$. Furthermore, $|\mathcal{R}_B \cup \mathcal{R}_C| = f$ by definition, so T satisfies at least k constraints in $\mathcal{R}_A$. Then there are at least k pairs $\{u, v\}$ of leaf labels from V forming an edge in G such that u and v are not both in the same subtree T_2 or T_3, which means $(V \cap \Lambda(T_2),\ V \cap \Lambda(T_3))$ is a cut of G of size at least k. □

Finally, we note that this problem variant has a very simple approximation algorithm:

Theorem 7. *MTC restricted to constraints of types 2 and 4 admits a polynomial-time* 1/2*-approximation algorithm.*

Proof. For $i \in \{2, 4\}$, let $\mathcal{R}_i$ be the subset of $\mathcal{R}$ of all triplet constraints of type i. If $|\mathcal{R}_2| \geq |\mathcal{R}_4|$ then output a phylogenetic tree consisting of a root node to which every element in S is directly attached; otherwise, output any binary phylogenetic tree T with $\Lambda(T) = S$. In both cases, at least half of the triplet constraints are satisfied. □

6. Extensions to weighted instances of MTC

Having input triplet constraints in the form of rooted triplets and forbidden rooted triplets, it is natural to assign nonnegative real weights to them. These weights can reflect the importance or certainty of the constraints. Consequently, MTC generalizes to the *maximum weighted rooted triplets consistency problem* (MWTC), where the input is a set S of leaf labels and $\mathcal{R}$ is a set of nonnegatively weighted triplet constraints, and the objective is to construct a phylogenetic tree T with $\Lambda(T) = S$ that maximizes the total weight of the satisfied triplet constraints from $\mathcal{R}$.

In the proof of Theorem 1 in Section 2, if we take the total weight of the constraints in each considered subset of triplet constraints instead of its cardinality (e.g., redefine m as the total weight of all constraints in $\mathcal{R}$) and use an approximation algorithm for the weighted version of MRTC in step 2 of Algorithm 1, then the approximation ratio analysis still holds. As shown in [10], the polynomial-time approximation algorithm for MRTC in [10, Section 5.1] with $f = 1/3$ also works for the weighted case, and we directly obtain:

Corollary 1′. *In polynomial time, one can find an approximate solution to MWTC whose total weight of satisfied input triplet constraints is at least* $m/4$*.*

Similarly, the exact algorithm for MTC in Section 3 where the output phylogenetic tree has to be k-ary generalizes to MWTC by using the sums of the weights of respective triplet constraints instead of their cardinalities in the definitions of $w_2(P)$, $w_3(P)$, $w_{f2}(P)$, and $w_{f3}(P)$. This gives the following generalization of Theorem 2.

Theorem 2′. *Let* $k \geq 2$ *be a given integer. For any instance of MWTC, one can find a* k*-ary phylogenetic tree* T *with* $\Lambda(T) = S$ *that maximizes the total weight of the satisfied triplet constraints in* $\mathcal{R}$ *among all* k*-ary phylogenetic trees using* $O((k+1)^{n+1} \cdot (m+n))$ *time.*

For the ETAS for MTC in Section 3 and the PTAS for MTC restricted to complete instances in Section 4, the situation requires an additional adjustment. Let W_{min} and W_{max} denote the minimum and the maximum weights among all triplet constraints in $\mathcal{R}$. In Theorem 3 and Lemma 3, some of the input triplet constraints in T and T_{opt} may be lost in T' and T_k, respectively, and we have to consider the worst case where each such lost constraint has weight W_{max}. In order to generalize our approximation schemes to the weighted case, we shall therefore require that W_{max} is at most $O(1)$ times larger than W_{min}. To extend the ETAS, we need the following generalization of Theorem 3.

Theorem 3′. *For any integer* $k \geq 13 \cdot \frac{W_{max}}{W_{min}}$ *and any phylogenetic tree* T*, there exists a* k*-ary phylogenetic tree* T' *with* $\Lambda(T') = \Lambda(T)$ *that satisfies a subset of the input triplet constraints whose total weight is at least a fraction of* $(1 - \frac{12 \cdot W_{max}}{k \cdot W_{min}})$ *of the total weight of all input triplet constraints satisfied by* T*.*

The proof of Theorem 3′ can be obtained from that of Theorem 3 by multiplying each contribution of a satisfied constraint by its weight. The only difference is that when estimating the fraction of the total weight of constraints lost in the approximation by a k-ary tree, we need to assume the worst-case situation where the lost constraints are of weight W_{max} while those preserved of weight W_{min}. This increases the fraction from the unweighted case by the factor $\frac{W_{max}}{W_{min}}$.

By combining Theorems 2′ and 3′, we obtain an ETAS for MWTC:

Theorem 4′. *For any instance of MWTC with $W_{max} = O(W_{min})$ and any specified constant $0 < \epsilon < 1$, one can build a phylogenetic tree satisfying a subset of the input triplet constraints whose total weight is at least a fraction of $(1 - \epsilon)$ of the optimum in $O((\lceil \frac{12}{\epsilon} \rceil + 1)^{n+1} \cdot (m + n))$ time.*

Finally, we explain how to extend the PTAS in Section 4. For any $\mathcal{R}' \subseteq \mathcal{R}$, let $W(\mathcal{R}')$ be the total weight of all elements in $\mathcal{R}'$. To start with, we need the following generalization of Lemma 3.

Lemma 3′. *Suppose that T is an optimal solution to an instance of MWTC and that T_k is a decomposition tree for T. The total weight of the triplet constraints from $\mathcal{R}$ satisfied by T_k is $W(\gamma(T_k) \cap \mathcal{R}) \geq W(\gamma(T) \cap \mathcal{R}) - \frac{122 \cdot W_{max}}{k} \cdot n^3$.*

The proof of Lemma 3′ is the same as that of Lemma 3, except that in the last step, we need to multiply the upper bound on the number of constraints lost (i.e., those in $(\gamma(T) \setminus \gamma(T_k)) \cap \mathcal{R}$) by W_{max} in order to obtain a worst-case upper bound on the total weight of constraints in $(\gamma(T) \setminus \gamma(T_k)) \cap \mathcal{R}$.

Next, we modify the smooth polynomial integer program for the LBA problem to include constraint weights by multiplying the polynomials corresponding to the constraints by the weights assigned to the constraints in the program. In analogy to Lemma 5, we obtain the following generalization.

Lemma 5′. *For every constant $0 < \epsilon' < 1$, there is a polynomial-time algorithm that for any instance of LBA specified by a set S of leaf labels, a set $\mathcal{R}$ of weighted triplet constraints for MWTC over S, and a kernel tree K, produces a phylogenetic tree T' with $W(\gamma(T') \cap \mathcal{R}) \geq W(\gamma(T) \cap \mathcal{R}) - \epsilon' \cdot n^3$, where T is an optimal LBA solution.*

By using Lemma 3′ and 5′, we thus obtain a PTAS for the complete version of MWTC:

Corollary 2′. *For any specified constant $0 < \epsilon < 1$, there is a polynomial-time algorithm for MWTC restricted to complete instances with $W_{min} \geq 1$ and $W_{max} = O(1)$ that produces a tree T'' approximating an optimal solution T^{opt} in such a way that $W(\gamma(T'') \cap \mathcal{R}) \geq (1 - \epsilon) \cdot W(\gamma(T^{opt}) \cap \mathcal{R})$.*

Proof. Choose any constant integer $k \geq \frac{56 \cdot 122 \cdot W_{max}}{\epsilon}$ and any constant $\epsilon' \leq \frac{\epsilon}{56}$. As in the proof of Theorem 5, apply the generalized approximate LBA-method to obtain T''. Next, follow the proofs of Theorem 5 and Corollary 2 to infer that $n^3 \leq 28 \cdot |\gamma(T^{opt}) \cap \mathcal{R}|$. By our assumptions on W_{min}, this yields $n^3 \leq 28 \cdot W(\gamma(T^{opt}) \cap \mathcal{R})$.

Let T_k^{opt} be a decomposition tree for T^{opt}. By Lemma 5′, $W(\gamma(T'') \cap \mathcal{R}) \geq W(\gamma(T_k^{opt}) \cap \mathcal{R}) - \epsilon' \cdot n^3$, and by Lemma 3′, $W(\gamma(T_k^{opt}) \cap \mathcal{R}) \geq W(\gamma(T^{opt}) \cap \mathcal{R}) - \frac{122 \cdot W_{max}}{k} \cdot n^3$. This yields $W(\gamma(T'') \cap \mathcal{R}) \geq W(\gamma(T^{opt}) \cap \mathcal{R}) - \frac{122 \cdot W_{max}}{k} \cdot n^3 - \epsilon' \cdot n^3 \geq W(\gamma(T^{opt}) \cap \mathcal{R}) \cdot (1 - \frac{28 \cdot 122 \cdot W_{max}}{k} - 28\epsilon') \geq W(\gamma(T^{opt}) \cap \mathcal{R}) \cdot (1 - \frac{28 \cdot 122 \cdot W_{max} \cdot \epsilon}{56 \cdot 122 \cdot W_{max}} - 28 \cdot \frac{\epsilon}{56}) = W(\gamma(T^{opt}) \cap \mathcal{R}) \cdot (1 - \epsilon)$. □

7. Open problems

MTC is APX-hard by the APX-hardness of MRTC [4] and Corollary 1. An open problem is to improve the polynomial-time approximation ratios 1/3 and 1/4 for MRTC and MTC. According to Theorem 1, an f-approximation for the former would give an $\frac{f}{1+f}$-approximation for the latter.

Algorithm 1 can be viewed as a derandomization of an algorithm that randomly assigns the leaf labels to the leaves of a rooted star tree (where the permutation of the leaf labels does not matter) and to an arbitrary binary tree (where the permutation of the leaf labels is significant), and then returns the better of the two obtained solutions. An interesting question is whether there exists a single tree shape for which randomly assigning the leaf labels to the leaves always yields an approximation ratio of 1/4 for MTC.

Another open problem is to design an exact algorithm for MRTC that is faster than $O^*(3^n)$. Is $O^*(2^n)$ time complexity achievable, and if so, can that algorithm be extended to MTC?

On the positive side, it was shown in [9] that MRTC can be solved in polynomial time if the input also contains a specified left-to-right ordering $\mathcal{O}$ of S and the output phylogenetic tree is required to be an *ordered* tree in which the left-to-right sequence of leaf labels equals $\mathcal{O}$. Does MTC become polynomial-time solvable in this setting as well?

Acknowledgments

J.J. was partially funded by The Hakubi Project at Kyoto University, Japan, and KAKENHI Grant Number 26330014. K.D. would like to thank Maciej Liśkiewicz for some valuable discussions related to the problems studied in this paper.

Addendum

Theorem 6 was recently strengthened in [J. Jansson, A. Lingas, R. Rajaby, W.-K. Sung, Determining the consistency of resolved triplets and fan triplets, Journal of Computational Biology 25 (7) (2018) 740-754.].

References

[1] A.V. Aho, Y. Sagiv, T.G. Szymanski, J.D. Ullman, Inferring a tree from lowest common ancestors with an application to the optimization of relational expressions, SIAM J. Comput. 10 (3) (1981) 405–421.
[2] S. Arora, D. Karger, M. Karpinski, Polynomial time approximation schemes for dense instances of *NP*-hard problems, J. Comput. System Sci. 58 (1) (1999) 193–210.
[3] D. Bryant, Building Trees, Hunting for Trees, and Comparing Trees: Theory and Methods in Phylogenetic Analysis (Ph.D. thesis), University of Canterbury, Christchurch, New Zealand, 1997.
[4] J. Byrka, P. Gawrychowski, K.T. Huber, S. Kelk, Worst-case optimal approximation algorithms for maximizing triplet consistency within phylogenetic networks, J. Discrete Algorithms 8 (1) (2010) 65–75.
[5] J. Byrka, S. Guillemot, J. Jansson, New results on optimizing rooted triplets consistency, Discrete Appl. Math. 158 (11) (2010) 1136–1147.
[6] B. Chor, M. Hendy, D. Penny, Analytic solutions for three taxon ML trees with variable rates across sites, Discrete Appl. Math. 155 (6–7) (2007) 750–758.
[7] J. Felsenstein, Inferring Phylogenies, Sinauer Associates, Inc., Sunderland, Massachusetts, 2004.
[8] M. Garey, D. Johnson, Computers and Intractability – A Guide to the Theory of NP-Completeness, W. H. Freeman and Company, New York, 1979.
[9] L. Gąsieniec, J. Jansson, A. Lingas, A. Östlin, Inferring ordered trees from local constraints, in: Proceedings of Computing: The 4th Australasian Theory Symposium (CATS'98), in: Australian Computer Science Communications, vol. 20(3), Springer-Verlag, 1998, pp. 67–76.
[10] L. Gąsieniec, J. Jansson, A. Lingas, A. Östlin, On the complexity of constructing evolutionary trees, J. Comb. Optim. 3 (2–3) (1999) 183–197.
[11] Y.J. He, T.N.D. Huynh, J. Jansson, W.-K. Sung, Inferring phylogenetic relationships avoiding forbidden rooted triplets, J. Bioinform. Comput. Biol. 4 (1) (2006) 59–74.
[12] M.R. Henzinger, V. King, T. Warnow, Constructing a tree from homeomorphic subtrees, with applications to computational evolutionary biology, Algorithmica 24 (1) (1999) 1–13.
[13] L. van Iersel, S. Kelk, M. Mnich, Uniqueness, intractability and exact algorithms: Reflections on level-*k* phylogenetic networks, J. Bioinform. Comput. Biol. 7 (4) (2009) 597–623.
[14] S. Jahangiri, S.N. Hashemi, H. Poormohammadi, New heuristics for rooted triplet consistency, Algorithms 6 (3) (2013) 396–406.
[15] J. Jansson, On the complexity of inferring rooted evolutionary trees, in: Proceedings of the Brazilian Symposium on Graphs, Algorithms, and Combinatorics (GRACO 2001), in: Electronic Notes in Discrete Mathematics, vol. 7, Elsevier, 2001, pp. 50–53.
[16] J. Jansson, R.S. Lemence, A. Lingas, The complexity of inferring a minimally resolved phylogenetic supertree, SIAM J. Comput. 41 (1) (2012) 272–291.
[17] J. Jansson, A. Lingas, E.-M. Lundell, A triplet approach to approximations of evolutionary trees. Poster H15 presented at RECOMB 2004, 2004.
[18] T. Jiang, P. Kearney, M. Li, A polynomial time approximation scheme for inferring evolutionary trees from quartet topologies and its application, SIAM J. Comput. 30 (6) (2001) 1942–1961.
[19] P. Kearney, Phylogenetics and the quartet method, in: T. Jiang, Y. Xu, M.Q. Zhang (Eds.), Current Topics in Computational Molecular Biology, The MIT Press, Massachusetts, 2002, pp. 111–133.
[20] M.P. Ng, N.C. Wormald, Reconstruction of rooted trees from subtrees, Discrete Appl. Math. 69 (1–2) (1996) 19–31.
[21] S. Snir, S. Rao, Using Max Cut to enhance rooted trees consistency, IEEE/ACM Trans. Comput. Biol. Bioinform. 3 (4) (2006) 323–333.
[22] M. Steel, The complexity of reconstructing trees from qualitative characters and subtrees, J. Classification 9 (1) (1992) 91–116.
[23] B.Y. Wu, Constructing evolutionary trees from rooted triplets, J. Inf. Sci. Eng. 20 (2004) 181–190.
[24] B.Y. Wu, Constructing the maximum consensus tree from rooted triples, J. Comb. Optim. 8 (1) (2004) 29–39.