On the Weighted Quartet Consensus problem

Manuel Lafond, Celine Scornavacca

On the Weighted Quartet Consensus problem*

Manuel Lafond[1,2] and Celine Scornavacca[3]

[1] Department of Computer Science, Université de Sherbrooke, 2500 Boulevard de l'Université, J1K 2R1, Sherbrooke, Canada

[2] Department of Mathematics and Statistics, 545 King Edward, K1N7N5, University of Ottawa, Ottawa, Canada (mlafond2@uottawa.ca)

[3] Institut des Sciences de l'Evolution – Université Montpellier, CNRS, IRD, EPHE, Place Eugène Bataillon, 34095 Montpellier, France (celine.scornavacca@umontpellier.fr).

Abstract. In phylogenetics, the *consensus* problem consists in summarizing a set of phylogenetic trees that all classify the same set of species into a single tree. Several definitions of consensus exist in the literature; in this paper we focus on the Weighted Quartet Consensus problem, whose complexity status has remained open since it was posed in 2008. Here we show that the Weighted Quartet Consensus problem is NP-hard, and provide several results on the approximability of the problem. Namely, we show that Weighted Quartet Consensus admits a polynomial-time approximation scheme (PTAS) based on prior work of Jiang, Kearney and Li. The algorithm is unfortunately impractical, and so we also present an implementable 1/2-factor approximation algorithm. During the process, we propose a derandomization procedure of a previously known *randomized* 1/3-factor approximation, which guarantees that a tree that contains a third of any given multi-set of quartets can be found deterministically. We also investigate the fixed-parameter tractability of this problem.

1 Introduction

Phylogenetics is the branch of biology that studies evolutionary relationships among different species. These relationships are represented via *phylogenetic trees* – acyclic connected graphs with leaves labeled by species – which are reconstructed from molecular and morphological data [13]. One fundamental problem in phylogenetics is to summarize the information contained in a set of unrooted trees $\mathcal{T}$ classifying the same set of species into a single tree T. The set $\mathcal{T}$ can consist of optimal trees for a single data set, of trees issued from a bootstrap analysis of a unique data set, or even of trees issued from the analysis of different data sets. Several consensus methods have been proposed in the past, the probably most known are the strict consensus [24, 19] and the majority-rule consensus [18, 5]. For a survey, see [8].

In this paper we focus on the Weighted Quartet Consensus (WQC) problem [20], also called the Median Tree with Respect to Quartet Distance problem [4] and Quartet Consensus problem in [17]. Roughly speaking, this problem consists in finding a consensus tree maximizing the weights of the 4-leaf trees – *quartets* – it displays, where the weight of a quartet is defined as its frequency in the set of input trees (for a more formal definition, see next section).

More general versions of this problem, where the input trees are allowed to have missing species or where the weight of a quartet is not defined w.r.t. a set of trees, are known to be NP-hard [25] (and in fact, even Max-SNP-Hard), but the complexity of the WQC problem has remained open since the question was first posed in 2008 [3]. This problem has been conjectured to be NP-hard [4, 20], and heuristics have recently been implemented and widely used, for instance ASTRAL [21], which is a practical implementation of Bryant and Steel's work from[9] (in fact, we show that the ASTRAL algorithm is a 2-approximation for the minimization version of WQC). So far, the best known approximation algorithm for the WQC problem consists in choosing a random tree as a solution [17]. This tree is expected to contain a third of the quartets from the input trees, and so it is a randomized factor 1/3 approximation. In [4], the *minimization* version of the problem is studied, where the objective is to find a median tree T minimizing the sum of quartet *distances* between T and the input trees (the quartet distance is the cardinality of the symmetric difference

* An extended abstract of this work appeared in the Proceedings of the 28th Annual Symposium on Combinatorial Pattern Matching (CPM), pages pp. 28:1–28:18, volume 78 of LIPIcs,, 2017.

between the sets of quartets in the two trees). A 2-approximation algorithm is given, based on the fact that the quartet distance is a metric [10, 4].

A related problem that has received more attention is the *Complete Maximum Quartet Compatibility* problem (CMQC) (see [7, 6, 17, 15, 26, 27, 11, 22, 23]). In the CMQC problem, we are given, for each set S of four species, exactly one quartet on S, and the objective is to find a tree containing a maximum number of quartets from the input. This can be seen as a version of WQC in which each set of four species has one quartet of weight 1, and the others have weight 0. It was shown in [17] that the CMQC problem admits a polynomial time approximation scheme (PTAS), but it can only be extended to WQC instances on a constant number of trees. Also, CMQC was shown in [15, 11] to be fixed-parameter tractable w.r.t. the parameter "number of quartets to reject from the input".

Our results. The main result of this paper is to establish the NP-hardness of the WQC problem. In Section 2, we introduce preliminary notions, and in Section 3 we propose a reduction from the NP-hard Cyclic Ordering problem to WQC. In Section 4, we discuss how being in a consensus setting, i.e. having weights based on a set of input trees on the same leaf set rather than arbitrary weights, does not necessarily make the problem easier, as one could expect: We list some structural properties that, surprisingly, are not satisfied by optimal solutions of a WQC instance. One example of this is that even if a quartet belongs to every single input tree, it is possible that none of the optimal solutions contain this quartet. Nevertheless, in Section 5 we show that WQC is amenable to approximation algorithms. We first show that the problem admits a PTAS, essentially by adjusting the weights appropriately and using the work of Jiang, Kearney and Li [17] to find a solution within a factor $(1-\epsilon)$ from optimal, for any $0 < \epsilon < 1$. This algorithm is unfortunately not practical, and so we devise a simpler factor $1/2$ approximation algorithm for WQC running in time $O(t^2n^2+tn^4+n^6)$, where t is the number of trees and n the number of species. During the process, we also show how to derandomize the algorithm that outputs a tree T that, on expectation, contains $1/3$ of the quartets from the input. To achieve this, we design a specific uniform tree sampling procedure that allows us to apply the technique of conditional expectations in the construction of the tree. Finally, in Section 6, we show that the known FPT algorithms for the CMQC problem can be extended to the consensus setting. This yields an FPT algorithm that is efficient on instances in which there is not too much ambiguity, i.e. when few competing quartets on the same 4 species appear with the same frequency. We then conclude with some remarks and open problems related to the quartet consensus problem.

2 Preliminaries

An *unrooted phylogenetic tree* T consists of vertices connected by edges, in which every pair of nodes is connected by exactly one path and no vertex is of degree two. The *leaves* of a tree T, denoted by $L(T)$, consists of the set of vertices of degree one. Each leaf is associated to a label; the set of labels associated to the leaves of a tree T is denoted by $\mathcal{L}(T)$. We suppose that there is a bijection between $L(T)$ and $\mathcal{L}(T)$. Due to this bijectivity, we will refer to leaves and labels interchangeably. We denote $|\mathcal{L}(T)|$ as $|T|$. In the following, we will often omit the word "phylogenetic" and, unless otherwise stated, all trees are leaf-labeled. A *binary* unrooted tree has only vertices with degree three and vertices with degree one. A rooted (binary) phylogenetic tree is defined in the same way, except that it has exactly one node of degree two called the *root*, denoted by $r(T)$. Note that sometimes in the literature, rooted trees are seen as directed and such that all arcs are oriented away from the root. Unless stated otherwise, all trees in this paper are unrooted.

Given an unrooted phylogenetic tree T and a subset $Y \subseteq \mathcal{L}(T)$, we denote by $T|Y$ the tree obtained from the minimal subgraph of T connecting Y when contracting degree-2 vertices. A *quadset* is a set of four labels. For a quadset $\{a, b, c, d\}$, there are three non-isomorphic[1] unrooted binary trees, called *quartets*, which are denoted respectively by $ab|cd$, $ac|bd$, and $ad|bc$, depending on how the central edge splits the four labels. We say that an unrooted tree T *displays* the quartet $ab|cd$ if $T|\{a, b, c, d\}$ is $ab|cd$. We denote the set of quartets that an unrooted tree T displays by $Q(T)$. Note that if T is binary, then $|Q(T)| = \binom{|T|}{4}$. A set of quartets Q on a set $\mathcal{X}$ is said to be

[1] Isomorphism preserving labels.

complete if for each quadset $\{a, b, c, d\} \subseteq \mathcal{X}$, there is in Q *exactly* one quartet among $ab|cd$, $ac|bd$, and $bc|ad$.

We are now ready to state our optimization problem. The Weighted Quartet Consensus problem asks for a tree that has as many quartets as possible in common with a given set of trees on the same set of labels $\mathcal{X}$:

Weighted Quartet Consensus (WQC) problem
Input: a set of unrooted trees $\mathcal{T} = \{T_1, \ldots, T_t\}$ such that $\mathcal{L}(T_1) = \ldots = \mathcal{L}(T_t) = \mathcal{X}$.
Output: a binary unrooted tree M with $\mathcal{L}(M) = \mathcal{X}$ that maximizes $\sum_{T \in \mathcal{T}} |Q(M) \cap Q(T)|$.

The problem is weighted as each quartet on $\mathcal{X}$ is weighted by frequency in $\mathcal{T}$, see below.

A tree M that maximizes the above objective function will be called an *optimal consensus tree*. In this paper we will focus on the particular case where the input trees are all binary. In fact, proving the problem NP-hard for this restricted case implies NP-hardness of the general problem. Note, however, that relaxing the requirement of the output M to be binary leads to a different problem, as one needs to define how unresolved quartets in M are weighted.

We denote by Q_{all} the set of all possible quartets on the set of labels $\mathcal{X}$, that is, $\mathsf{Q}_{all} = \{ab|cd : \{a, b\}, \{c, d\} \in \binom{\mathcal{X}}{2}, \{a, b\} \cap \{c, d\} = \emptyset\}$. In the remainder of the article, we will sometimes consider multi-sets of quartets, that are sets in which the same quartet can appear multiple times. Denote by $f_{\mathcal{Q}}(q)$ the number of times that a quartet q appears in a multi-set $\mathcal{Q}$ (we may write $f(q)$ if $\mathcal{Q}$ is unambiguous). We also denote $|\mathcal{Q}| = \sum_{q \in \mathsf{Q}_{all}} f(q)$. We say that a tree T *contains* k quartets of $\mathcal{Q}$ if there are distinct quartets $q_1, \ldots, q_p \in Q(T)$ such that $\sum_{i=1}^{p} f(q_i) = k$. A set or multi-set $\mathcal{Q}$ of quartets is said *compatible* if there exists a tree that contains each quartet of $\mathcal{Q}$. Given a quadset $\{a, b, c, d\}$, we say that $ab|cd$ is *dominant* (w.r.t. f) if $f(ab|cd) \geq f(ac|bd)$ and $f(ab|cd) \geq f(ad|bc)$. We say that $ab|cd$ is *strictly dominant* if both inequalities are strict.

Observe that the Weighted Quartet Consensus problem can be rephrased as follows: given the trees $T_1, \ldots, T_t$ on leafset $\mathcal{X}$, find a binary tree M on $\mathcal{X}$ that contains a maximum number of quartets from $Q(T_1) \uplus Q(T_2) \uplus \ldots \uplus Q(T_t)$, where $\uplus$ denotes multiset union. We will implicitly work with the decision version of WQC, i.e. for a given integer q, is there a consensus tree M containing at least q quartets from $Q(T_1) \uplus Q(T_2) \uplus \ldots \uplus Q(T_t)$? This latter formulation will sometimes be more convenient.

3 NP-hardness of the Weighted Quartet Consensus problem

In this section, we present a reduction from the Cyclic ordering problem. This problem has been used in phylogenetics before in [16] in the context of inferring rooted binary trees from rooted triplets (trees on 3 leaves).

But beforehand, we need some additional notation. A *caterpillar* is an unrooted binary tree obtained by taking a path $P = p_1 p_2 \ldots p_r$, then adding a leaf ℓ_i adjacent to p_i for each $1 \leq i \leq r$, then adding another leaf ℓ'_1 adjacent to p_1 and a leaf ℓ'_r adjacent to p_r. The two leaves ℓ'_1 and ℓ'_r inserted last are called the *ends* of the caterpillar. An *augmented caterpillar* T is a binary tree obtained by taking a caterpillar, then replacing each leaf by a binary rooted tree (replacing a leaf ℓ by a tree T' means replacing ℓ by $r(T')$). If T_1, T_2 are the two trees replacing the ends of the caterpillar, then T is called a (T_1, T_2)-augmented caterpillar. For instance, the 6 trees in Figure 1 are all (W, Z)-augmented caterpillars, with the special case that all the "inner" trees have only one leaf. Note that every binary tree is a (T_1, T_2)-augmented caterpillar for some T_1, T_2. Let T be a caterpillar with leaves $(\ell_1, \ell_2, \ldots, \ell_k)$ taken in the order in which we encounter them on the path between the two ends ℓ_1 and ℓ_k (more precisely, traverse the $\ell_1 - \ell_k$ path, and each time an internal node is encountered, append its adjacent leaves to the sequence), and let $T_1, \ldots, T_k$ be rooted binary trees. We denote by $(T_1|T_2|\ldots|T_k)$ the (T_1, T_k)-augmented caterpillar obtained by replacing ℓ_i by T_i, $1 \leq i \leq k$. This notation gives rise to a natural ordering of the subtrees, and we say that $T_i < T_j$ if $i < j$ (i.e. T_i appears before T_j in the given ordering of the caterpillar subtrees). Note that other orderings of the subtrees can yield the same unrooted tree, e.g. by reversing it, or by exchanging T_1, T_2 or T_{k-1}, T_k. However this will be inconsequential in the proofs that follow. If each T_i consists of a single leaf ℓ_i for $2 \leq i \leq k-1$, then we may write $(T_1|\ell_2|\ldots|\ell_{k-1}|T_k)$, and $\ell_i < \ell_j$ in T to indicate that ℓ_i appears before ℓ_j in the ordering (see e.g. Figure 1).

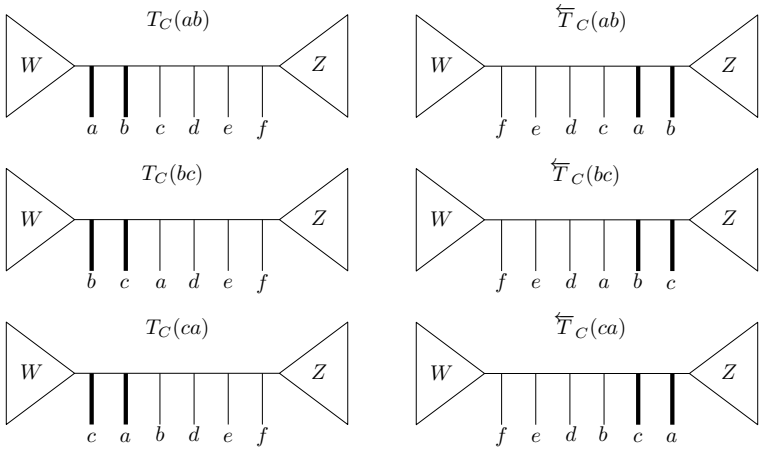

Fig. 1. The 6 trees of $\mathcal{T}(C)$ corresponding to the triple $C = (a, b, c)$. In this example, $S = \{a, b, c, d, e, f\}$. The first, second and third row respectively show the "$a < b$", "$b < c$" and "$c < a$" trees.

We are now ready to describe the Cyclic Ordering problem. Let $L = (s_1, \ldots, s_n)$ be a linear ordering of a set S of at least 3 elements, and let (a, b, c) be an ordered triple, with a, b, c being distinct elements of S. We say that L *satisfies* (a, b, c) if one of the following holds in L: $a < b < c, b < c < a$ or $c < a < b$. If $\mathcal{C}$ is a set of ordered triples, we say that L satisfies $\mathcal{C}$ if it satisfies every element of $\mathcal{C}$. Intuitively speaking, L satisfies (a, b, c) when, by turning L into a directed cyclic order by attaching s_n to s_1, one can go from a to b, then to c and then to a. This leads to the following problem definition:

Cyclic Ordering problem
Input: A set S of n elements and a set $\mathcal{C}$ of m ordered triples (a, b, c) of distinct members of S.
Question: Does there exist a linear ordering $L = (s_1, \ldots, s_n)$ of S that satisfies $\mathcal{C}$?

The Cyclic Ordering problem is NP-hard [14]. In the rest of this section, we let S and $\mathcal{C}$ be the input set and triples, respectively, of a Cyclic Ordering problem instance. We denote $n = |S|$ and $m = |\mathcal{C}|$. We shall use the following simple yet useful characterization.

Lemma 1. *A linear ordering L of S satisfies $\mathcal{C}$ if and only if for each $(a, b, c) \in \mathcal{C}$, exactly two of the following relations hold in L: $a < b, b < c, c < a$.*

Proof. ($\Rightarrow$): let L be a linear ordering satisfying $\mathcal{C}$, and let $(a, b, c) \in \mathcal{C}$. Thus in L, one of $a < b < c, b < c < a$ or $c < a < b$ holds. It is straightforward to verify that in each case, exactly two of $a < b, b < c, c < a$ hold.
($\Leftarrow$): suppose that L does not satisfy $\mathcal{C}$. Then there is some $(a, b, c) \in \mathcal{C}$ such that one of $a < c < b, b < a < c$ or $c < b < a$ does not hold. Again, one can easily verify that each of these cases satisfies only one of $a < b, b < c$ and $c < a$. □

We now start describing the reduction from Cyclic Ordering to WQC. Given S and $\mathcal{C}$, we construct a set of unrooted binary trees $\mathcal{T}$ on a common set of labels $\mathcal{X}$.

First let W and Z be two very large rooted binary trees, each on say $(nm)^{100}$ leaves (the topology is arbitrary). The leaf sets $\mathcal{L}(W)$ and $\mathcal{L}(Z)$ are disjoint. Our trees are defined on the leaf set $\mathcal{X} = S \cup \mathcal{L}(W) \cup \mathcal{L}(Z)$ (where $S, \mathcal{L}(W), \mathcal{L}(Z)$ are disjoint). Let $C \in \mathcal{C}$ with $C = (a, b, c)$. We construct 6 trees from C, that is, 3 pairs of trees. These are illustrated in Figure 1.

- The "$a < b$" trees: let $(s_1, \ldots, s_{n-2})$ be an arbitrary ordering of $S \setminus \{a, b\}$. Then we build the trees $T_C(ab) = (W|a|b|s_1|s_2|\ldots|s_{n-2}|Z)$ and $\overleftarrow{T}_C(ab) = (W|s_{n-2}|s_{n-3}|\ldots|s_1|a|b|Z)$.
- The "$b < c$" trees: let $(\hat{s}_1, \ldots, \hat{s}_{n-2})$ be an arbitrary ordering of $S \setminus \{b, c\}$. Then we build the trees $T_C(bc) = (W|b|c|\hat{s}_1|\hat{s}_2|\ldots|\hat{s}_{n-2}|Z)$ and $\overleftarrow{T}_C(bc) = (W|\hat{s}_{n-2}|\hat{s}_{n-3}|\ldots|\hat{s}_1|b|c|Z)$.
- The "$c < a$" trees: let $(\bar{s}_1, \ldots, \bar{s}_{n-2})$ be an arbitrary ordering of $S \setminus \{c, a\}$. Then we build the trees $T_C(ca) = (W|c|a|\bar{s}_1|\bar{s}_2|\ldots|\bar{s}_{n-2}|Z)$ and $\overleftarrow{T}_C(ca) = (W|\bar{s}_{n-2}|\bar{s}_{n-3}|\ldots|\bar{s}_1|c|a|Z)$.

Denote by $\mathcal{T}(C)$ the set of 6 constructed trees for $C \in \mathcal{C}$. The input for our Weighted Quartet Consensus instance constructed from S and $\mathcal{C}$ is $\mathcal{T} = \bigcup_{C \in \mathcal{C}} \mathcal{T}(C)$. Note that $|\mathcal{T}| = 6m$. Before moving on to the proof of the reduction, it will be useful to state a few remarks and introduce some terminology.

First observe that each tree of $\mathcal{T}(C)$ is a (W, Z)-augmented caterpillar. Moreover, note that the pairs of trees of $\mathcal{T}(C)$ contain "opposite" orderings for all but one pair of elements from S. To state this more precisely, let $a, b \in S$ and $x, y \in S \setminus \{a, b\}$, and let $\{T_C(ab), \overleftarrow{T}_C(ab)\}$ be an "$a < b$" tree-pair. Then we have $x < y$ in $T_C(ab)$ if and only if $y < x$ in $\overleftarrow{T}_C(ab)$. Similarly for any $x \in S \setminus \{a, b\}$, $a < x, b < x$ in $T_C(ab)$ but $x < a, x < b$ in $\overleftarrow{T}_C(ab)$. Only $a < b$ holds in both trees.

Let $T \in \mathcal{T}$, and let $B(T)$ denote the set of quartets of T that have at least two members of $\mathcal{L}(W)$, or at least two members of $\mathcal{L}(Z)$. Thus $B(T)$ consists of all the quartets of the form $w_1w_2|xy$ and $z_1z_2|xy$ of T, where $w_1, w_2 \in \mathcal{L}(W), z_1, z_2 \in \mathcal{L}(Z)$ and $x, y \in \mathcal{X}$ (note that no quartet of $B(T)$ has the form $w_1x|yw_2$ for $x, y \notin \mathcal{L}(W)$, nor the form $z_1x|yz_2$ for $x, y \notin \mathcal{L}(Z)$). Since all the trees from $\mathcal{T}$ are (W, Z)-augmented caterpillars, for any tree $T' \in \mathcal{T}$, we have $B(T) = B(T')$. Let $K_{wz} := 6m|B(T)|$ be the total number of such quartets in $\mathcal{T}$, i.e. K_{wz} is the size of $\biguplus_{T \in \mathcal{T}} B(T)$. We observe the following:

Remark 1. Any (W, Z)-augmented caterpillar on $\mathcal{X}$ contains the K_{wz} quartets $\biguplus_{T \in \mathcal{T}} B(T)$.

Now, denote $K_{out} := 3m|W||Z|\left(\binom{n-2}{2} + 2(n-2)\right)$ (the meaning of this expression will become apparent shortly). Let $T \in \mathcal{T}$ and suppose that T is an "$a < b$" tree, for some $a, b \in S$. For $w \in \mathcal{L}(W), z \in \mathcal{L}(Z)$ and $x, y \in S$, a quartet $wx|yz$ displayed by T is called an *out-quartet* of T if $\{x, y\} \neq \{a, b\}$, and an *in-quartet* of T if $x = a$ and $y = b$ (note that $x = b$ and $y = a$ is not possible, by construction of "$a < b$" trees). Let $out(T)$ and $in(T)$ denote the set of out-quartets and in-quartets, respectively, of T. Note that each tree T has $|W||Z|$ in-quartets and $|W||Z|\left(\binom{n-2}{2} + 2(n-2)\right)$ out-quartets (because there are $\binom{n-2}{2} + 2(n-2)$ ways to choose $\{x, y\} \neq \{a, b\}$). Thus K_{out} is half the total number of out-quartets $\biguplus_{T \in \mathcal{T}} out(T)$. As it turns out, it will be the in-quartets that will be decisive in showing our hardness proof. The following Lemma is a first hint towards the structure of an optimal consensus tree for $\mathcal{T}$.

Lemma 2. *Any tree M on leafset $\mathcal{X}$ contains at most K_{out} quartets from $\biguplus_{T \in \mathcal{T}} out(T)$. Moreover, if M is a (W, Z)-augmented caterpillar $(W|s_1|\dots|s_n|Z)$, where $\{s_1, \dots, s_n\} = S$, then M contains exactly K_{out} quartets from $\biguplus_{T \in \mathcal{T}} out(T)$.*

Proof. Let $w \in \mathcal{L}(W)$ and $z \in \mathcal{L}(Z)$. Let $\{T_C(ab), \overleftarrow{T}_C(ab)\}$ be an "$a < b$" tree-pair of $\mathcal{T}$, for some $a, b \in S$, and let $x, y \in S$ such that $\{x, y\} \neq \{a, b\}$. Because $x < y$ in $T_C(ab)$ if and only if $y < x$ in $\overleftarrow{T}_C(ab)$, we get that the out-quartet $wx|yz$ is in $T_C(ab)$ if and only if $wy|xz$ is in $\overleftarrow{T}_C(ab)$. Since M can only contain one of the two quartets, it follows that M can contain at most half of the quartets from $out(T_C(ab)) \uplus out(\overleftarrow{T}_C(ab))$. Thus M contains at most half the quartets from $\biguplus_{T \in \mathcal{T}} out(T)$, which is $3m|W||Z|\left(\binom{n-2}{2} + 2(n-2)\right) = K_{out}$. As for the second assertion, if $M = (W|s_1|\dots|s_n|Z)$ then M contains one of $wx|yz$ or $wy|xz$ for each $x, y \in S$. Thus if M does not contain the out-quartet $wx|yz$ from $T_C(ab)$, then it contains the out-quartet $wy|xz$ from $\overleftarrow{T}_C(ab)$. We deduce that M contains at least half the quartets from $out(T_C(ab)) \uplus out(\overleftarrow{T}_C(ab))$, and thus half the quartets from $\biguplus_{T \in \mathcal{T}} out(T)$. □

Before going further, we make a useful general observation.

Proposition 1. *Let X, Y be two non-empty sets such that $Y \not\subseteq X$. Then $|X| \cdot |Y \setminus X| \geq |Y| - 1$.*

Proof. First note that for any strictly positive pair of integers x and y, $x \cdot y \geq x + y - 1$ (with equality if and only if $x = 1$ or $y = 1$). Now, suppose to begin with that $X \cap Y = \emptyset$. Then clearly $|X||Y \setminus X| = |X||Y| \geq |Y| - 1$. Suppose otherwise that $|X \cap Y| = k > 0$. Denote $h = |Y \setminus X|$, so that $|Y| = k + h$. Then $|X||Y \setminus X| \geq k \cdot h \geq k + h - 1 = |Y| - 1$. □

What follows is the intuitive statement that since all input trees are (W, Z)-augmented caterpillars, any optimal consensus tree should also have this property.

Lemma 3. *Any optimal consensus tree for $\mathcal{T}$ is a (W, Z)-augmented caterpillar.*

Proof. We first introduce the notion of a rooted subtree of a binary unrooted tree T. By removing an edge $e = \{u, v\}$ of T, we obtain two disjoint rooted subtrees T_1 and T_2, respectively rooted at u and v. Call T' a *rooted subtree* of T if T' is a rooted tree that can be obtained by removing an edge of T. For $X \subset \mathcal{L}(T)$, a *rooted subtree for* X is a rooted subtree T' of T such that $X \subseteq \mathcal{L}(T')$. We denote by $T[X]$ the rooted subtree for X that contains a minimum number of leaves (if there are multiple choices, choose $T[X]$ arbitrarily among the possible choices). Note that $T[X]$ always exists since we can remove an edge incident to a leaf $y \notin X$. However, $T[X]$ may contain leaves other than X.

We now prove that any optimal solution to $\mathcal{T}$ as constructed in our reduction must be a (W, Z)-augmented caterpillar. Suppose that M is an optimal solution for $\mathcal{T}$, and that M is not a (W, Z)-augmented caterpillar. Denote $M_W = M[\mathcal{L}(W)]$ and $M_Z = M[\mathcal{L}(Z)]$. If M is a (W', Z')-augmented caterpillar $(W'|T_1|\dots|T_k|Z')$ for some trees W', Z' with $\mathcal{L}(W') = \mathcal{L}(W)$ and $\mathcal{L}(Z') = \mathcal{L}(Z)$, it is not hard to see that $M' = (W|T_1|\dots|T_k|Z)$ is a better solution than M, a contradiction. Thus, M is not such a caterpillar, and this implies that either $\mathcal{L}(M_W) \neq \mathcal{L}(W)$ or $\mathcal{L}(M_Z) \neq \mathcal{L}(Z)$ (or both). That is, the rooted subtrees containing $\mathcal{L}(W)$ and/or $\mathcal{L}(Z)$ have "outsider" leaves. Suppose first that $\mathcal{L}(M_W) \neq \mathcal{L}(W)$ holds. Then there exists a node x with children x_l and x_r in M_W such that all leaves X_l below x_l are in $\mathcal{L}(W)$ with $\mathcal{L}(W) \neq X_l$ (otherwise $M_W = M[\mathcal{L}(W)]$ would be chosen incorrectly), and no leaf X_r below x_r belongs to $\mathcal{L}(W)$ (this can be seen by observing that the node x of M_W having leaves both in W and not in W that is the farthest from the root has this property). Figure 2 illustrates this.

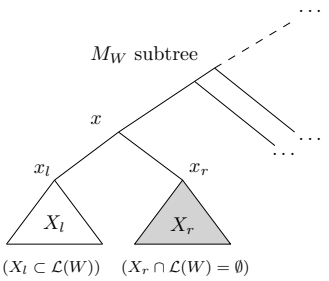

Fig. 2. The situation when $\mathcal{L}(M_W) \neq \mathcal{L}(W)$. X_r is grayed to indicate that it has no element from $L(W)$.

We claim that $\mathcal{L}(Z) \not\subseteq X_r$. Suppose otherwise that $\mathcal{L}(Z) \subseteq X_r$. Then $|X_r| \geq |Z|$ and so $|M_W| \geq |W| + |Z|$. However in M, by removing the xx_r edge we obtain two rooted trees, one of which is a rooted subtree for $\mathcal{L}(W)$. Moreover, this subtree has at most $|W| + |S| < |W| + |Z|$ leaves, which contradicts the minimality of $M_W = M[\mathcal{L}(W)]$. We deduce that $\mathcal{L}(Z)$ is not a subset of X_r.

Now, observe that M contains the quartet $w_1y|w_2z$ for each $w_1 \in X_l, y \in X_r, w_2 \in \mathcal{L}(W) \setminus X_l, z \in \mathcal{L}(Z) \setminus X_r$. There are at least

$$|X_l| \cdot |X_r| \cdot |\mathcal{L}(W) \setminus X_l| \cdot |\mathcal{L}(Z) \setminus X_r| \geq (|W| - 1)(|Z| - 1)$$

such quartets. The inequality is obtained by applying Proposition 1 to $|X_l| \cdot |\mathcal{L}(W) \setminus X_l|$ and $|X_r| \cdot |\mathcal{L}(Z) \setminus X_r|$, which is applicable since we have argued that $\mathcal{L}(W)$ is not a subset of X_l and $\mathcal{L}(Z)$ is not a subset of X_r. Moreover, each input tree of $\mathcal{T}$ contains the quartet $w_1w_2|yz$ instead, and hence in total in $\mathcal{T}$ there are at least $6m(|W| - 1)(|Z| - 1)$ quartets of the form $w_1w_2|yz$ that M does not contain. In the same manner, if the case $\mathcal{L}(M_Z) \neq \mathcal{L}(Z)$ holds, then there are at least $6m(|W| - 1)(|Z| - 1)$ quartets of the form $z_1z_2|yw$ that M does not contain, where here $z_1, z_2 \in \mathcal{L}(Z), y \notin \mathcal{L}(Z), w \in \mathcal{L}(W)$.

Now, let $\rho(M)$ be the number of quartets that M contains from $\biguplus_{T \in \mathcal{T}} Q(T)$ that have the form $wx|yz$, where $w \in \mathcal{L}(W), z \in \mathcal{L}(Z), x, y \in S$. Formally,

$$\rho(M) = \sum_{\substack{wx|yz \in Q(M) \\ x,y \in S \\ w \in \mathcal{L}(W) \\ z \in \mathcal{L}(Z)}} f(wx|yz)$$

recalling that $f(wx|yz)$ denotes the number of trees of $\mathcal{T}$ that contain the $wx|yz$ quartet. For a given $u \in \mathcal{L}(W) \cup \mathcal{L}(Z)$, let $\rho(M, u)$ denote the number of quartets counted in $\rho(M)$ that contain u. Formally, if $w \in \mathcal{L}(W)$, we have

$$\rho(M, w) = \sum_{\substack{wx|yz \in Q(M) \\ x,y \in S \\ z \in \mathcal{L}(Z)}} f(wx|yz)$$

The definition of $\rho(M, z)$ is the same for $z \in \mathcal{L}(Z)$, except that z gets fixed instead of w in the summation. Notice that

$$\rho(M) = \sum_{w \in \mathcal{L}(W)} \rho(M, w) = \sum_{z \in \mathcal{L}(Z)} \rho(M, z)$$

Let $w^* = \arg\max_{w \in \mathcal{L}(W)}\{\rho(M, w)\}$, i.e. the element w^* of $\mathcal{L}(W)$ that is in the maximum number of quartets of the form $w^*x|yz$. We obtain an alternative solution M' from M in the following manner: remove all leaves of $\mathcal{L}(W) \setminus \{w^*\}$ from M, delete the degree 2 nodes, and replace w^* by the W tree. Note that if $w^*x|yz$ is a quartet of M, then $wx|yz$ is a quartet of M' for all $w \in \mathcal{L}(W)$, and so $\rho(M', w) \geq \rho(M, w)$ for all such w by the choice of w^*. Consequently, $\rho(M') \geq \rho(M)$. We repeat the same operation on M' for the Z tree and obtain our final tree M^*. That is, we find $z^* = \arg\max_{z \in \mathcal{L}(Z)}\{\rho(M', z)\}$, and replace z^* by the Z tree. As above, we obtain $\rho(M^*) \geq \rho(M')$. Since M^* has W and Z as rooted subtrees, it follows that M^* is a (W, Z)-augmented caterpillar.

To finish the argument, we show that M^* contains more quartets from the input trees than M. First observe that the quartets on which M and M^* differ must contain a member of $\mathcal{L}(W) \cup \mathcal{L}(Z)$, since only these leaves switched position. We separately consider the following three types of possible quartets: (1) the quartets that contain at least two elements of $\mathcal{L}(W)$ or at least two elements of $\mathcal{L}(Z)$ (or both); (2) the quartets that contain exactly one member of $\mathcal{L}(W)$ and exactly one member of $\mathcal{L}(Z)$; and (3) those quartets that contain exactly one member of $\mathcal{L}(W) \cup \mathcal{L}(Z)$.

For (1), the tree M^* contains every quartet of $\biguplus_{T \in \mathcal{T}} Q(T)$ that have at least two members of $\mathcal{L}(W)$, or two members of $\mathcal{L}(Z)$. This includes the aforementioned (at least) $6m(|W|-1)(|Z|-1)$ quartets of the form $w_1w_2|yz$ or $z_1z_2|yw$ that M does not contain. As for (2), the quartets that contain exactly one member of $\mathcal{L}(W)$ and exactly one member of $\mathcal{L}(Z)$, M^* contains at least as many such quartets as M since in $\biguplus_{T \in \mathcal{T}} Q(T)$, these quartets are all of the form $wx|yz$, and we have $\rho(M^*) \geq \rho(M)$. Finally for (3), each tree of $\mathcal{T}$ has at most $(|W| + |Z|)n^3$ quartets that have exactly one member of $\mathcal{L}(W) \cup \mathcal{L}(Z)$. Thus at most $6m(|W| + |Z|)n^3$ quartets of this type are contained by M and not contained by M^*.

So from M to M^*, we have gained at least $6m(|W|-1)(|Z|-1)$ quartets due to (2) and lost at most $6m(|W| + |Z|)n^3$ due to (3). Since the latter quantity is smaller than $6m(|W|-1)(|Z|-1)$ for our choice of $|W|$ and $|Z|$, M^* contains more quartets from the input than M. □

We finally arrive at our main result.

Theorem 1. *The* Weighted Quartet Consensus *problem is NP-hard.*

Proof. It is straightforward to see that the construction of $\mathcal{T}$ from S and $\mathcal{C}$ can be carried out in polynomial time. We show that there exists a linear ordering of S satisfying $\mathcal{C}$ if and only if there exists a weighted quartet consensus tree M for $\mathcal{T}$ that contains at least $K_{wz} + K_{out} + 4m|W||Z|$ quartets from $\biguplus_{T \in \mathcal{T}} Q(T)$. For the rest of the proof, we let $w \in \mathcal{L}(W)$ and $z \in \mathcal{L}(Z)$ be arbitrary leaves of W and Z, respectively.

($\Rightarrow$): let $L = (s_1, s_2, \ldots, s_n)$ be a linear ordering of S satisfying $\mathcal{C}$. Then we claim that the weighted quartet consensus tree $M = (W|s_1|s_2|\ldots|s_n|Z)$ contains the desired number of quartets. Since M is a (W, Z)-augmented caterpillar, M contains K_{wz} quartets of $\mathcal{T}$ that have two or more elements from $\mathcal{L}(W)$, or two or more elements from $\mathcal{L}(Z)$, see Remark 1. Moreover by Lemma 2, M contains K_{out} quartets from $\biguplus_{T \in \mathcal{T}} out(T)$. As for the in-quartets, let $(a, b, c) \in \mathcal{C}$ and let $\mathcal{T}((a, b, c))$ be the set of 6 trees corresponding to (a, b, c). By Lemma 1, L satisfies two of the relations $a < b, b < c, c < a$. This implies that M has exactly two of the following quartets: $wa|bz, wb|cz, wc|az$. Since, for every $w \in \mathcal{L}(W)$ and $z \in \mathcal{L}(Z)$, each of these three quartets appears

as an in-quartet in exactly two trees of $\mathcal{T}((a,b,c))$ (e.g. $wa|bz$ is an in-quartet of $T_{(a,b,c)}(ab)$ and $\overleftarrow{T}_{(a,b,c)}(ab)$), it follows that M contains $4|W||Z|$ quartets of $\biguplus_{T\in\mathcal{T}((a,b,c))} in(T)$. As this holds for every $(a,b,c) \in C$, M contains $4m|W||Z|$ quartets of $\biguplus_{T\in\mathcal{T}} in(T)$. Summing up, we get that M has at least $K_{wz} + K_{out} + 4m|W||Z|$ quartets from $\mathcal{T}$.

($\Leftarrow$): suppose that no linear ordering of S satisfies $\mathcal{C}$. Let M be an optimal consensus tree for $\mathcal{T}$. By Lemma 3, we may assume that M is a (W,Z)-augmented caterpillar. We bound the number of quartets of $\mathcal{T}$ that can be contained in M.

First, by Lemma 3, M contains K_{wz} quartets of $\mathcal{T}$ that have at least two elements of $\mathcal{L}(W)$ or at least two elements of $\mathcal{L}(Z)$. As for the quartets with one or zero elements from $\mathcal{L}(W) \cup \mathcal{L}(Z)$, in any tree $T \in \mathcal{T}$ there are at most $(|W| + |Z|)n^3$ quartets of the form $wa|bc$ or $za|bc$ with $a,b,c \in S$, and at most n^4 quartets of the form $ab|cd$ with $a,b,c,d \in S$. Thus M contains at most $6m((|W| + |Z|)n^3 + n^4) < (|W| + |Z|)mn^5$ quartets of $\mathcal{T}$ that are of the form $wa|bc$, $za|bc$ or $ab|cd$ with $a,b,c,d \in S$ (the inequality holds because $n \geq 3$ and $|W| = |Z| = (nm)^{100}$). Also, by Lemma 2, M contains at most K_{out} quartets from $\biguplus_{T\in\mathcal{T}} out(T)$. It remains to count the in-quartets.

Let $(a,b,c) \in \mathcal{C}$. The following in-quartets appear, each twice, in $\mathcal{T}((a,b,c))$: $wa|bz$, $wb|cz$, $wc|az$. It is easy to check that these three quartets are incompatible, i.e. no tree can contain all three of them, and hence M can have at most two of them. We deduce that there must be at least two trees $T, \overleftarrow{T}$ of $\mathcal{T}((a,b,c))$ such that M contains no quartet from $in(T) \uplus in(\overleftarrow{T})$. Therefore M contains at most $4|W||Z|$ quartets from $\biguplus_{T\in\mathcal{T}((a,b,c))} in(T)$, and thus at most $4m|W||Z|$ quartets from $\biguplus_{T\in\mathcal{T}} in(T)$. We will however show that there must be some $(a,b,c) \in \mathcal{C}$ such that M contains only $2|W||Z|$ of the quartets from $\biguplus_{T\in\mathcal{T}((a,b,c))} in(T)$.

Since M is a (W,Z)-augmented caterpillar, we may write $M = (W|T_1|T_2|\dots|T_k|Z)$. For $a \in S$, let $T(a)$ be the tree of $\{T_1,\dots,T_k\}$ that contains a as a leaf. Then a quartet $wa|bz$ is in $Q(M)$ if and only if $T(a) < T(b)$. Let L be a linear ordering of S such that $T(a) < T(b) \Rightarrow a < b$ in L (L is easily seen to exist). Since no linear ordering of S can satisfy $\mathcal{C}$, by Lemma 1 there must be some $(a,b,c) \in \mathcal{C}$ such that only one of $a < b, b < c, c < a$ holds in L. This also means that at most one of $T(a) < T(b), T(b) < T(c), T(c) < T(a)$ holds. Thus M has at most one of the $wa|bz, wb|cz, wc|az$ quartets. It follows that M contains at most $2|W||Z|$ quartets from $\biguplus_{T\in\mathcal{T}((a,b,c))} in(T)$. Therefore M contains at most $4m|W||Z| - 2|W||Z|$ quartets of $\biguplus_{T\in\mathcal{T}} in(T)$.

In total, the number of quartets that M contains from the input is bounded by $K_{wz} + K_{out} + (|W| + |Z|)mn^5 + (4m-2)|W||Z| < K_{wz} + K_{out} + 4m|W||Z|$, by our choice of $|W|$ and $|Z|$. □

4 The (non)-structure of WQC

In the rest of this paper, we aim at designing algorithms building on the fact that the weight of each quartet is not arbitrary, and is rather based on a set of input trees on the same leaf set. When designing optimized algorithms for a problem, understanding the relationship between the input and the optimal solution(s) can be of great help. In phylogenetics, several problems are harder in the supertree setting, i.e. when the input trees do not all contain the same species, than in the consensus setting as in the WQC problem. An example is the problem of finding an unrooted phylogenetic tree containing as minors a set of unrooted phylogenetic trees – the compatibility problem – which is NP-hard in the supertree setting [25] and polynomially solvable in the consensus setting [1]. Despite the NP-hardness of WQC, there may exist some properties inherent to the consensus setting that are useful for devising efficient FPT algorithm, or for establishing lower bounds on the value of an optimal solution in order to develop approximation algorithms.

In an attempt to establish useful properties of the weights of quartets in the consensus setting, we initially conjectured that the following relationships between the input trees and the solution(s) hold. Despite being reasonable in appearance, we prove all these conjectures false.

1. let D be the set of strictly dominant quartets of the input multiset $\mathcal{Q}$, i.e. the quartets $ab|cd$ such that $f(ab|cd) > f(ac|bd)$ and $f(ab|cd) > f(ad|bc)$. Then there is a constant $\alpha > 0$ such that there exists an optimal solution containing at least $\alpha|D|$ quartets of D;
2. if a quartet $ab|cd$ has a higher weight than the sum of the other quartets on the same quadset, i.e. $f(ab|cd) > f(ac|bd) + f(ad|bc)$, then some optimal solution contains $ab|cd$;

3. more generally, there exists $\beta > 0$ such that if a quartet $ab|cd$ is in a fraction β of the input trees, then $ab|cd$ must be in some optimal solution. In particular, if $ab|cd$ is in *every* input tree, then there is some optimal solution that contains $ab|cd$;
4. if a quartet $ab|cd$ is in no input tree, then no optimal solution contains $ab|cd$.
5. call $ab|cd$ a *strictly least-frequent quartet* if $f(ab|cd) < f(ac|bd)$ and $f(ab|cd) < f(ad|bc)$. Suppose that there exists a tree T^* on leaf set $\mathcal{X}$ that contains no strictly least-frequent quartet, and choose such a T^* that contains a maximum number of quartets from the input. Then T^* is an optimal solution for WQC.

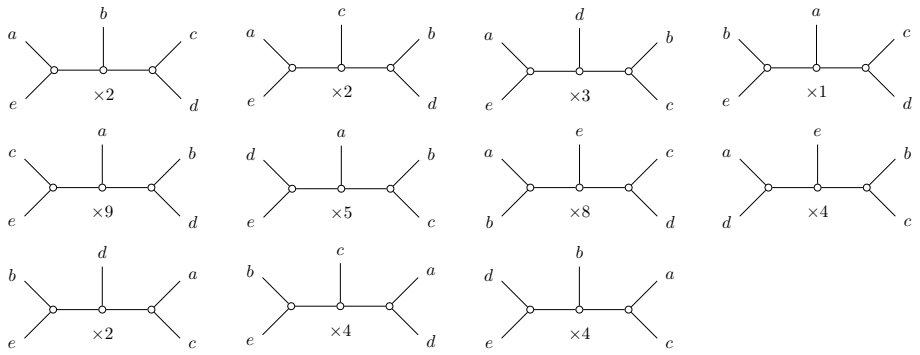

Fig. 3. An instance of WQC such that the optimal solution (the third tree from the left on the first row) contains no strictly dominant quartet. The numbers correspond to the number of times that each tree appears in the input.

Conjecture 1 is disproved by Theorem 2, and Conjecture 3 by Theorem 3, which implies that Conjectures 2 and 4 are also false; finally Conjecture 5 is disproved by Theorem 4.

Theorem 2. *There exists an instance of WQC such that every optimal solution contains none of the strictly dominant quartets.*

Proof. Figure 3 shows an instance of WQC demonstrating Theorem 2. Table 1 lists the frequency of each quartet in this instance.

Table 1. Frequency of each quartet (left column contains the quadsets)

abcd	$ab\|cd$ 11	$ac\|bd$ 17	$ad\|bc$ 16
abce	$ab\|ce$ 17	$ac\|cb$ 11	$ae\|bc$ 16
abde	$ab\|de$ 17	$ad\|be$ 11	$ae\|bd$ 16
acde	$ac\|de$ 11	$ad\|ce$ 17	$ae\|cd$ 16
bcde	$bc\|de$ 16	$bd\|ce$ 11	$be\|cd$ 17

Notice that for every possible quadset S, there is a strictly dominant quartet appearing 17 times, whereas the second-most and third-most quartets appear in 16 and 11 trees, respectively. Consider the third tree on the top row of Figure 3 (the $ae|bc$ quartet with d grafted on the middle branch). Call this tree T^*. For every quadset S, T^* contains the second-most frequent quartet on S, and so it contains 80 quartets from the input. Now, one can check that any tree T that contains a strictly dominant quartet for some quadset S must also contain a least frequent quartet on some other quadset S'. Hence, as there are 5 quadsets, T contains at most $4 \cdot 17 + 11 = 79$ quartets from the input. To see this, consider a tree that contains the strictly dominant $ac|bd$ quartet. Wherever we graft the e leaf on this quartet, we either have $bd|cd$ or $ac|de$, which are the least frequent.

Similarly, $ab|ce$ enforces either $ab|cd$ or $bd|cd$; $ab|de$ enforces either $ab|cd$ or $ac|de$; $ad|ce$ enforces either $ad|be$ or $bd|ce$; $be|cd$ enforces either $ad|be$ or $ab|cd$. □

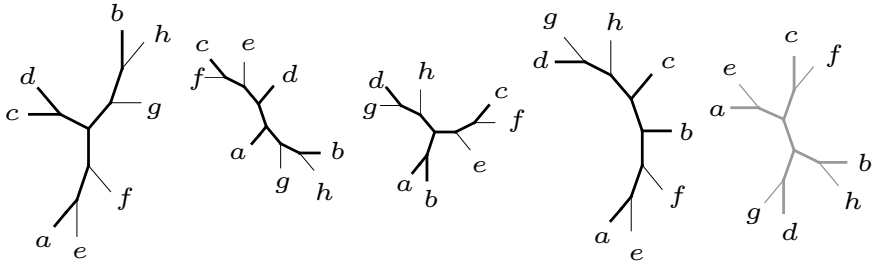

Fig. 4. The first four trees form an instance of WQC in which every tree contains $ab|cd$. The rightmost tree is the unique optimal solution to the WQC instance (every possible solution was verified computationally).

Note that the instance used to prove Theorem 2 consists of trees on only 5 leaves. We do not know if such instances exist for any $n > 5$ leaves.

Theorem 3. *There exists an instance of WQC such that there is a quartet q that appears in every input tree, but q is not a quartet of any optimal solution.*

Figure 4 shows an instance of WQC proving Theorem 3. Each input tree contains the $ab|cd$ quartet, whereas the optimal solution, which is unique, does not. Indeed using brute-force checking of every possible tree computionally, we found that the rightmost tree contains 180 quartets from the input multiset $\mathcal{Q}$, whereas any other tree has at most 176.

Finally, we note that the main interest behind Conjecture 5 is the following: if it holds, in cases where the set F of strictly least-frequent quartets is complete we could tell in polynomial time – using results of [9] – whether there is a tree T^* that contains no quartet from F. Conjecture 5 could then lead to interesting approximations or FPT algorithms. However, least-frequent quartets cannot be excluded automatically.

Theorem 4. *There exists an instance of WQC such that every optimal solution contains a strictly least-frequent quartet, even if there exists a tree T^* with no such quartet.*

Proof. The instance corresponding to Theorem 4 is obtained from the instance shown in Figure 3, by removing all occurrences of the third tree on the top row (i.e. this tree now appears 0 times instead of 3 times). Since this tree contained each second-most frequent quartet on every quadset, each such quartet now appears 13 times in the input (i.e. in Table 1, replace all occurrences of 16 by 13). The tree T^* that contains all these second-most frequent quartets has a total weight of $5 \cdot 13 = 65$. Notice that there are trees that contain 75 quartets from the input, for instance, the tree of cardinality 9 in the figure. It follows that any optimal tree must have a strictly dominant quartet, since T^* is not optimal. But as we argued in the proof of Theorem 2, these optimal trees must also contain a strictly least-frequent quartet. □

5 Approximatibility of WQC

In this section, we first show that WQC admits a polynomial time approximation scheme (PTAS). The algorithm is however impractical both from the point of view of time complexity and implementability. Therefore, we also give a factor 1/2 approximation algorithm that achieves practical running times and is easy to implement. Note that this is better than the (randomized) factor 1/3 approximation for WQC, the best known so far. We also derandomize this approximaiton algorithm.

We say that an algorithm A for a maximization (resp. minimization) problem P is an α-*approximation* of P, where $\alpha < 1$ (resp. $\alpha > 1$), if for every instance I of P, A runs in polynomial time and outputs a solution of value $APP(I)$ such that $APP(I) \geq \alpha OPT(I)$ (resp. $APP(I) \leq \alpha OPT(I)$), where $OPT(I)$ is the optimal value of I. For a maximization problem P, a PTAS is an algorithm (or a family of algorithms) that, given some $0 < \epsilon < 1$, runs in time $O(n^{g(\epsilon)})$ for some function g not depending on n, and always outputs a solution of value at least $(1-\epsilon)OPT(I)$.

We will make use of an important feature of quartets throughout this section. Let $\mathcal{Q}$ be a multi-set of quartets over the set $\mathcal{X}$, not necessarily originating from a set of trees. If we sample a binary tree T on leafset $\mathcal{X}$ uniformly at random, each quartet of $\mathcal{Q}$ has probability $1/3$ of being contained in T, since there are 3 quartets for each quadset and each is equally likely. This gives rise to a trivial factor $1/3$ randomized approximation algorithm: output a random tree [2]. Later in this section, we will show how to derandomize this algorithm, which will be useful to devise a *deterministic* $1/2$ approximation for WQC. This property is also useful in designing a PTAS when $\mathcal{Q}$ originates from input trees. The fact that a random tree, on expectation, contains $1/3$ of the quartets from $\mathcal{Q}$ yields the following.

Proposition 2. *Let $T_1, \ldots, T_t$ be a set of binary trees on leafset $\mathcal{X}$, and let $\mathcal{Q} = Q(T_1) \uplus \ldots \uplus Q(T_t)$. Then there exists a tree T on leaf set $\mathcal{X}$ such that $\sum_{q \in Q(T)} f_{\mathcal{Q}}(q) \geq \frac{1}{3} \sum_{i=1}^{t} |Q(T_i)| = \frac{t}{3}\binom{n}{4}$.*

5.1 A PTAS for WQC

Recall that the Complete Maximum Quartet Compatibility (CMQC) problem asks, given a complete set Q of quartets on $\mathcal{X}$, for a tree T that contains a maximum number of quartets from Q. As mentioned before, CMQC admits a PTAS [17], the idea of the algorithm being to enumerate every possible binary tree T with $h(\epsilon)$ leaves for some function h, then "plug-in" the elements of $\mathcal{X}$ into the leaves of T whilst maximizing the number of quartets in common with the input. This latter step is performed using smooth polynomial integer programming (see [2]).

The authors of [17] also indicate how this PTAS can be extended to the WQC problem, but only when the number of input trees is constant. In fact, this is a consequence of a more general result on the existence of a PTAS when the pool of possible weights is constant. More specifically, the authors establish that, to paraphrase, "the CMQC PTAS can be extended to solve this weighted variation (of CMQC) as long as the weights are drawn from some interval of positive integers of constant range to preserve the smoothness of the polynomial integer programs". This means that if there is a constant $\lambda \in \mathbb{N}$ such that a given quartet weighing function f satisfies the following conditions:

1. $f(q) \in \mathbb{N}$ for all $q \in \mathsf{Q}_{all}$
2. for every quadset S, $f(ab|cd) + f(ac|bd) + f(ad|bc) = \lambda$

then the PTAS can find a solution within a factor $(1-\epsilon)$ from optimal in time $O(n^{g(\epsilon,\lambda)})$ where here g is a function depending on ϵ and λ but not on n. The difficulty in the WQC problem is that this constant λ is not guaranteed to exist.

Nevertheless, we show that the PTAS can still be adapted to WQC. Suppose that $(1-\epsilon)$ is the desired approximation factor. The idea is simply to fix λ appropriately, then divide and round each weight of the WQC instance to the nearest weight in $\{0, \ldots, \lambda\}$. Here λ can depend on ϵ, but not on n. We will assume that λ is known, and determine its exact value later.

Recall that t is the number of trees in the input (hence, $f(q) \leq t$ for every quartet q). For each quartet $q \in \mathsf{Q}_{all}$, define a temporary weight $f'(q) = \lfloor f(q)/t \cdot \lambda \rfloor$. We have

$$f(q)/t \cdot \lambda - 1 \leq f'(q) \leq f(q)/t \cdot \lambda$$

Hence, for a quadset $\{a, b, c, d\}$, because $f(ab|cd) + f(ac|bd) + f(ad|bc) = t$, we have

$$\lambda - 3 \leq f'(ab|cd) + f'(ac|bd) + f'(ad|bd) \leq \lambda$$

[2] Here, the approximation is "randomized" in the sense that the output tree contains an *expected* number of quartets from the input that is at least a third.

To obtain the weights $\hat{f}$ that we will actually use, we adjust f' slightly to ensure that the quadset weights sum to λ, since this allows us to use the PTAS. We know that $f'(ab|cd)+f'(ac|bd)+f'(ad|bd) = \lambda - x$, where $x \in \{0,1,2,3\}$ since the f' weights are integral. To define $\hat{f}$ on the quartets of $\{a,b,c,d\}$, choose an arbitrary subset $S \subseteq \{ab|cd, ac|bd, ad|bc\}$ with $|S| = x$, set $\hat{f}(q) = f'(q)+1$ to each $q \in S$, and set $\hat{f}(q) = f'(q)$ to the quartets on $\{a,b,c,d\}$ not in S. This guarantees that $\hat{f}(ab|cd) + \hat{f}(ac|bd) + \hat{f}(ad|bd) = \lambda$.

Moreover, for any quartet q, we now have

$$f(q)/t \cdot \lambda - 1 \leq \hat{f}(q) \leq f(q)/t \cdot \lambda + 1$$

We now compare an optimal solution for $\hat{f}$ with an optimal solution for f. Let T' be a tree that maximizes $\sum_{q \in Q(T')} \hat{f}(q)$ among all possible trees. Let us denote $Q' := Q(T')$. Moreover, let T be a tree that maximizes $\sum_{q \in Q(T)} f(q)$, and denote $Q := Q(T)$. Write $OPT(\hat{f}) := \sum_{q \in Q'} \hat{f}(q)$ and $OPT(f) := \sum_{q \in Q} f(q)$. Since Q is a possible solution for the weights $\hat{f}$, we have

$$OPT(\hat{f}) \geq \sum_{q \in Q} \hat{f}(q) \geq \sum_{q \in Q} (f(q)/t \cdot \lambda - 1) = \frac{\lambda}{t} \sum_{q \in Q} f(q) - \binom{n}{4} = \frac{\lambda}{t} OPT(f) - \binom{n}{4}$$

Now, suppose that we run the PTAS on the weights $\hat{f}$ with approximation ratio $(1-\epsilon')$, where ϵ' is yet to be determined. We obtain a tree T'' with quartet set $Q'' := Q(T'')$ satisfying $\sum_{q \in Q''} \hat{f}(q) \geq (1-\epsilon') OPT(\hat{f})$. We argue that taking Q'' as a solution for the WQC instance defined by f (instead of $\hat{f}$) is close to $OPT(f)$. We have

$$\begin{aligned}
\sum_{q \in Q''} f(q) &\geq \sum_{q \in Q''} \frac{t}{\lambda} (\hat{f}(q) - 1) = \frac{t}{\lambda} \left(\sum_{q \in Q''} \hat{f}(q) - \binom{n}{4} \right) \\
&\geq \frac{t}{\lambda} \left((1-\epsilon') OPT(\hat{f}) - \binom{n}{4} \right) \\
&\geq \frac{t}{\lambda} \left((1-\epsilon') \left[\frac{\lambda}{t} OPT(f) - \binom{n}{4} \right] - \binom{n}{4} \right) \\
&= (1-\epsilon') OPT(f) - \frac{t(2-\epsilon')}{\lambda} \binom{n}{4}
\end{aligned}$$

By Proposition 2, $OPT(f) \geq \frac{t}{3}\binom{n}{4}$, i.e. $\binom{n}{4} \leq \frac{3}{t} OPT(f)$. Therefore,

$$\begin{aligned}
(1-\epsilon') OPT(f) - \frac{t(2-\epsilon')}{\lambda} \binom{n}{4} &\geq (1-\epsilon') OPT(f) - \frac{t(2-\epsilon')}{\lambda} \frac{3}{t} OPT(f) \\
&= \left(1 - \epsilon' - \frac{3(2-\epsilon')}{\lambda} \right) OPT(f)
\end{aligned}$$

If $\left(1 - \epsilon' - \frac{3(2-\epsilon')}{\lambda}\right) \geq (1-\epsilon)$, then we will have achieved our desired approximation factor. As long as $\epsilon > \epsilon'$, by choosing $\lambda \geq \frac{6-3\epsilon'}{\epsilon-\epsilon'}$, this inequality holds. For instance, we can set $\epsilon' = \epsilon/2$ and set $\lambda = 12/\epsilon - 3$ to get the $(1-\epsilon)$ approximation. Therefore, a PTAS for WQC can be obtained by setting λ appropriately, adjusting the weights and running the PTAS from [17].

5.2 Preserving 1/3 of the input quartets deterministically

We now show how, given a multiset of quartets $\mathcal{Q}$, one can deterministically output a tree T that contains at least 1/3 of the quartets from $\mathcal{Q}$. Instead of working with multi-sets, in this section we will assume that we are given a weight function $f : \mathsf{Q}_{all} \to \mathbb{N}$ on each possible quartet (here the weights do not have to originate from the quartets of a set of input trees). For the remainder of this section, let $\mathcal{X}$ be the set of species and let $n = |\mathcal{X}|$.

Theorem 5. *There exists a deterministic algorithm that outputs a tree T satisfying $\sum_{q \in Q(T)} f(q) \geq \frac{1}{3} \sum_{q' \in \mathcal{Q}_{all}} f(q')$. Moreover, this algorithm runs in time $O(n^6)$.*

We use the method of conditional expectations to prove Theorem 5. The idea is to start from a fully unresolved tree on leafset $\mathcal{X}$, then refine it into a binary tree in a top-down, step-by-step fashion, ensuring at each step that we maintain the expectation of containing 1/3 of the input quartets. To simplify the description of our procedure, we shall work with rooted trees in this section.

It is worth mentioning that an analogous bottom-up approach called *quartet joining* has been published multiple times (e.g. [12, 28]). The algorithm consists in starting from an unresolved tree, making the two "best" leaves of this tree a cherry, contracting these two leaves with their parent and repeating (see references for a definition of *best*). However, it can be shown that this method does not guarantee preserving 1/3 of the quartets[3].

For an integer m, let $t(m)$ denote the number of rooted binary trees whose set of leaves are labeled by $\{1, \ldots, m\}$. It is well-known that $t(m) = (2m-3)!!$, and we will assume that in a preprocessing step, we have computed and stored $t(m)$ for all $1 \leq m \leq n$. This takes initial time $O(n^2)$ and allows constant-time access to the value of $t(m)$ in our algorithm. Note that alternatively $t(m) = \frac{1}{2} \sum_{j=1}^{m-1} \binom{n}{j} t(j)t(m-j)$ also holds, since a rooted binary tree can be obtained by giving j leaves to its left child and the other $m-j$ leaves to its right child, and resolving the left and right trees (and we divide by 2 due to the symmetry).

Call a rooted tree T *internally binary* if the only nodes of T that have more than two children have only leaves as children. We say a rooted binary tree T' *resolves* an internally binary tree T if T can be obtained from T' by some sequence of edge contractions. Given an internally binary tree T, we describe a procedure to select a tree that resolves T uniformly at random.

Repeat the following steps until T is binary:

1. Choose a node v of T that has more than one child. Let $v_1, \ldots, v_m$ be the children of v, which are all leaves.
2. Choose an integer $s \in \{1, \ldots, m-1\}$ at random using the following distribution: for $j \in \{1, \ldots, m-1\}$,
$$Pr[s = j] = \frac{\binom{m}{j} t(j)t(m-j)}{2t(m)}$$
3. Remove all the children of v from T and add two leaf children to v labeled by ρ_s and ρ_{m-s}.
4. Reinsert the $v_1, \ldots, v_m$ leaves by first choosing a subset $S \subset \{v_1, \ldots, v_m\}$ with $|S| = s$ at random, each subset with equal probability $1/\binom{m}{s}$, then inserting each element of S as a leaf child of ρ_s. Then, insert all the elements of $\{v_1, \ldots, v_m\} \setminus S$ as leaf children of ρ_{m-s}.

This concludes the procedure. It is easy to see that a binary tree will eventually be obtained, and that the order in which the v nodes are processed does not matter, as each non-binary node is processed independently. Note that in the second step, $\sum_{j=1}^{m-1} Pr[s = j] = 1$ (this follows from $t(m) = \frac{1}{2} \sum_{j=1}^{m-1} \binom{m}{j} t(j)t(m-j)$). We then show that this procedure can be used to sample trees uniformly.

Lemma 4. *Given an internally binary tree T, the above procedure chooses one of the trees that resolve T uniformly at random.*

Proof. We prove the statement by induction over the number of leaves n. As a base case, the statement is easy to verify by hand for small values of n, say $n = 2$ or $n = 3$. For larger n, let T be an internally binary tree and let $B = \{b_1, \ldots, b_l\}$ be its set of nodes that have more than two children. For each $b_i \in B$, denote by $B_i \subseteq \mathcal{X}$ the set of leaf labels that occur as the children of b_i. Observe that one can obtain a tree that resolves T uniformly at random by choosing, for each $b_i \in B$, a tree on leafset B_i chosen uniformly at random, then replacing the b_i node by the root of this tree. If T has more than one internal node, then $|B_i| < n$ for each $1 \leq i \leq l$. By our

[3] For instance, it fails to do so on input $f(xx'|zz') = 100, f(xa|x'b) = f(xc|x'd) = (xe|x'f) = 99$, as it will output a tree in which x, x' form a cherry, thereby preventing all the other quartets.

induction hypothesis, the procedure replaces each b_i node by a tree on leafset B_i chosen uniformly at random, as desired. If T has only one internal node, then T is a star tree and the space of trees that resolve T consists of all the trees on leafset $\mathcal{X}$. Let T' be any rooted binary tree on leafset $\mathcal{X}$. Let u, v be the children of the root of T', let T'_u and T'_v be the subtrees of T' rooted at u and v, respectively, and let $n_u = |\mathcal{L}(T'_u)|$ and $n_v = |\mathcal{L}(T'_v)|$. For the above procedure to yield T', the following events must occur:

E_1: the integer selected s is equal to n_u or n_v;
E_2: the set $\mathcal{L}(T'_u)$ gets selected to be the children of ρ_s or ρ_{m-s};
E_3: the nodes ρ_s and ρ_{m-s} resolve to T'_u and T'_v.

Noting that $n_v = n - n_u$ and that $\binom{n}{n_u} = \binom{n}{n_v}$, the probability that all three events occur is

$$Pr[E_1] \cdot Pr[E_2|E_1] \cdot Pr[E_3|E_1E_2] = \left[2\frac{\binom{n}{n_u}t(n_u)t(n-n_u)}{2t(m)}\right] \cdot \left[\frac{1}{\binom{n}{n_u}}\right] \cdot \left[\frac{1}{t(n_u)}\frac{1}{t(n-n_u)}\right]$$

where the three brackets corresponds to the three probabilities to multiply, and where we have used the induction hypothesis for $Pr[E_3|E_1, E_2]$. After canceling the terms, this probability is equal to $\frac{1}{t(n)}$, as desired. □

Observe that our procedure deals with temporary trees having two special nodes ρ_s and ρ_{m-s} and some leaves of $\mathcal{X}$ missing. We will need to complete and resolve such trees uniformly at random. We thus needs to define this notion before proceeding to our algorithm. We shall call an internally binary tree T a *partial s-tree* if $\mathcal{L}(T) \subseteq \mathcal{X}$ and T has two nodes ρ_s and ρ_{m-s} that have the same parent. Moreover, ρ_s has at most $c \leq s$ children (possibly $c = 0$ or $c = 1$), all of which are leaves, and ρ_{m-s} has at most $m - s$ children, also all leaves (and possibly 0 or 1 children). We say that an internally binary tree T' with $\mathcal{L}(T') = \mathcal{X}$ *completes* T if T' can be obtained from T by adding $s - c$ leaves as children of ρ_s and all the other remaining leaves of $\mathcal{X}$ as children of ρ_{m-s}. Finally, a rooted binary tree T_r *resolves* T if there exists a tree T' that completes T such that T_r resolves T'. It is not hard to see that a tree T_r that resolves T can be sampled uniformly by choosing a subset S of $s - c$ elements of $\mathcal{X} \setminus \mathcal{L}(T)$ uniformly at random, inserting S as leaf children of ρ_s, inserting all other remaining leaves under ρ_{m-s} and executing the above procedure on the resulting internally binary tree. Also note that $\mathcal{L}(T) = \mathcal{X}$ is possible in the definition of a partial s-tree, in which case no completion is needed.

If T is an internally binary tree or a partial s-tree, we denote by $ab|cd \in T$ the random event that a tree T' that resolves T chosen uniformly at random contains the $ab|cd$ quartet. Thus $Pr[ab|cd \in T]$ is the probability that our procedure for internally binary trees or partial s-trees results in a tree that contains $ab|cd$. Given the weight function $f : \mathsf{Q}_{all} \to \mathbb{N}$, denote by $\mathbb{E}[T] = \sum_{q \in \mathsf{Q}_{all}} Pr[q \in T]f(q)$ the expected number of quartets that T contains.

Algorithm 1 describes the derandomization procedure, which is essentially a deterministic version of our random tree sampling procedure. We start with a fully unresolved star tree T that contains, on expectation, 1/3 of the quartets from the input, and each decision in the algorithm maintains this expectation. When resolving a non-binary node v with children $v_1, \ldots, v_m$, instead of choosing a random size s to split its children into two sets of size s and $m - s$, we try every possible size s and keep the size that yields the best expectation. When evaluating a size s, we create the two nodes ρ_s and ρ_{m-s} and insert $v_1, \ldots v_m$ one at a time under either ρ_s or ρ_{m-s}, whichever yields the better expectation. We stop when ρ_s or ρ_{m-s} is full. Note that the output tree T may contain nodes that have only one child - we simply delete these nodes from T.

Notice that the algorithm computes the probability $Pr[q \in T_s]$ that a quartet belongs to a partial s-tree. We will show later how to compute this probability, and for now we show that the 1/3 expectation is always maintained as claimed.

Lemma 5. *Let T^* be the (possibly non-binary) tree obtained at the end of any iteration of the* while *loop of Algorithm 1, i.e. T^* is the tree T after executing line 17. Then $\mathbb{E}[T^*] \geq \frac{1}{3}\sum_{q' \in \mathsf{Q}_{all}} f(q')$.*

Proof. For the remainder of the proof, denote $F := \sum_{q' \in \mathsf{Q}_{all}} f(q')$. Let T be the tree at the start of the iteration of the *while* loop, before executing line 3. We may inductively assume that $\mathbb{E}[T] \geq F/3$. Indeed, this holds for the initial star tree T constructed before entering the first iteration of the

Algorithm 1: saveOneThirdQuartets($\mathcal{X}, f$)

Data: A set $\mathcal{X}$ of n taxa and a weight $f : \mathsf{Q}_{all} \to \mathbb{N}$ for every possible quartet.
Result: A tree T such that $\sum_{q \in Q(T)} f(q) \geq \frac{1}{3} \sum_{q' \in \mathsf{Q}_{all}} f(q')$

1 Let T be a rooted star tree with leaf set $\mathcal{X}$ (i.e. a tree in which each leaf is a child of the root).
2 **while** T *is not binary* **do**
3 Let v be a node of T with more than 2 children $v_1, \ldots, v_m$.
4 **for** $s = 1, 2, \ldots, m-1$ **do**
5 Let T_s be the partial s-tree obtained from T by removing $v_1, \ldots, v_m$, and adding two children labeled ρ_s and ρ_{m-s} to v.
6 **for** $i = 1, 2, 3, \ldots, m$ **do**
7 Let $T_{s,1}$ be the partial s-tree obtained by adding v_i as a child of ρ_s.
8 Let $T_{s,2}$ be the partial s-tree obtained by adding v_i as a child of ρ_{m-s}.
9 Let $score\_x = \mathbb{E}[T_{s,1}] = \sum_{q \in \mathsf{Q}_{all}} Pr[q \in T_{s,1}] f(q)$.
10 Let $score\_y = \mathbb{E}[T_{s,2}] = \sum_{q \in \mathsf{Q}_{all}} Pr[q \in T_{s,2}] f(q)$.
11 **if** $score\_x > score\_y$ **then**
12 Insert v_i as a child of ρ_s in T_s.
13 **else**
14 Insert v_i as a child of ρ_{m-s} in T_s.
15 **if** *ρ_s has s children or ρ_{m-s} has $m-s$ children* **then**
16 Complete T_s in the only possible way, and break out of the current for loop.
17 Put T as the tree T_i of $T_1, \ldots, T_{m-1}$ that maximizes $\mathbb{E}[T_i]$.

loop. Moreover, in the subsequent iterations, this tree T is the tree obtained at the end of the previous iteration - hence if we prove the statement for an arbitrary iteration, our $\mathbb{E}[T] \geq F/3$ assumption holds.

Let v be the non-binary node chosen on line 3 with children $v_1, \ldots, v_m$. By our assumption on T, by choosing a tree T_r that resolves T uniformly at random, T_r has at least $F/3$ quartets from the input. Let T_s be the partial s-tree defined as on line 5, $1 \leq s \leq m-1$. We claim that there must be some value of s such that $\mathbb{E}[T_s] \geq F/3$. By Lemma 4, a tree that resolves T can be chosen uniformly at random by picking $1 \leq s \leq m-1$ with some probability p_s, and then completing and resolving the underlying partial s-tree. For a given quartet q, we may thus write $Pr[q \in T] = \sum_{s=1}^{m-1} p_s \cdot Pr[q \in T_s]$. Therefore,

$$\mathbb{E}[T] = \sum_{q \in \mathsf{Q}_{all}} Pr[q \in T] f(q) = \sum_{q \in \mathsf{Q}_{all}} \sum_{s=1}^{m-1} p_s \cdot Pr[q \in T_s] f(q) = \sum_{s=1}^{m-1} p_s \mathbb{E}[T_s]$$

Since $\mathbb{E}[T] \geq F/3$ and $\sum_{s=1}^{m-1} p_s = 1$, it follows that some T_s must have $\mathbb{E}[T_s] \geq F/3$.

Since the algorithm tries every T_s and retains the best one, it suffices to show that our way of completing the T_s with $\mathbb{E}[T_s] \geq F/3$ retains the desired expectation. So assume that $\mathbb{E}[T_s] \geq F/3$. The algorithm inserts $v_1, \ldots, v_m$ under either ρ_s or ρ_{m-s} one leaf after another. For $0 \leq i \leq m$, denote by $T_s^{(i)}$ the tree obtained in the algorithm after inserting v_i, with $T_s^{(0)} = T_s$. We prove by induction that $\mathbb{E}[T_s^{(i)}] \geq F/3$ for each $0 \leq i \leq m$. As a base case, this holds for $T_s^{(0)}$ as we have argued just above. Now consider the iteration in which v_i is inserted, $1 \leq i \leq m$. By induction, $\mathbb{E}[T_s^{(i-1)}] \geq F/3$. Suppose that in $T_s^{(i-1)}$, ρ_s has k_1 children and ρ_{m-s} has k_2 children, where $k_1 < s$ and $k_2 < m - s$ (as otherwise one of the ρ_s and ρ_{m-s} nodes would be full and we would be done completing T_s). In a resolution of $T_s^{(i-1)}$, v_i is inserted under ρ_s with probability $p_{v_i} = \binom{m-k_1-k_2-1}{s-k_1-1} / \binom{m-k_1-k_2}{s-k_1}$ and under ρ_{m-s} with probability $1 - p_{v_i}$ (the actual probabilities do not matter here though). Thus

$$\mathbb{E}[T_s^{(i-1)}] = p_{v_i} \mathbb{E}\left[T_s^{(i-1)} | v_i \text{ is a child of } \rho_s\right] + (1 - p_{v_i}) \mathbb{E}\left[T_s^{(i-1)} | v_i \text{ is a child of } \rho_{m-s}\right]$$

Since $\mathbb{E}[T_s^{(i-1)}] \geq F/3$ and $p_{v_i} + (1 - p_{v_i}) = 1$, it follows that one of the trees obtained from $T_s^{(i-1)}$ by inserting v_i under ρ_s or ρ_{m-s} attains an expected number of quartets in common with

the input of at least $F/3$. This shows that after completion, T_s has $F/3$ quartets from the input on expectation, which concludes the proof since the *while* loop will finish with T_s being chosen as T^*. □

Since Algorithm 1 terminates with a binary tree T with $\mathbb{E}[T] \geq F/3$ and that all quartets of T are fully determined, there is no randomness involved in $\mathbb{E}[T]$ and it follows that Algorithm 1 outputs a binary tree T that contains a third of the input quartets.

Computing the probability of a quartet. It remains to show how, when inserting some child v_i of v under either ρ_s or ρ_{m-s}, to compute $Pr[q \in T_{s,1}]$ and $Pr[q \in T_{s,2}]$ on lines 9 and 10 for a given quartet q. Assume that in $T_{s,1}$ or $T_{s,2}$, the leaves $v_1, \ldots, v_{i-1}$ have all been reinserted under either ρ_s or ρ_{m-s} under the same parent, with v_i being the leaf currently being evaluated for insertion.

Observe that if $\Pr[q \in T_{s,1}] = \Pr[q \in T_{s,2}]$, then this probability does not contribute to determining which scenario maximizes expectation, and in this case we do not need to consider q. Denote $q = ab|cd$. If none of a, b, c, d is a child of v, then $\Pr[q \in T_{s,1}] = \Pr[q \in T_{s,2}]$, since the choice of inserting v_i under ρ_s or ρ_{m-s} does not affect the quartet into which a, b, c, d resolve. In fact, the same holds if exactly one or exactly two of a, b, c, d are children of v. Thus we may assume that at least three of a, b, c, d are children of v.

There are still multiple cases depending on which of a, b, c and d are children of v, and which have been reinserted or have not, but this probability can be found algorithmically. We shall focus on determining $Pr[ab|cd \in T_{s,1}]$, the calculcation for $Pr[ab|cd \in T_{s,2}]$ being identical. To ease notation, denote $T' = T_{s,1}$. For $x \in \{a, b, c, d\}$, write $x < \rho_s$ (resp. $x < \rho_{m-s}$) to denote the event that in a random completion of T', x is a child of ρ_s (resp. ρ_{m-s}). Note that $Pr[x < \rho_s] \in \{0, 1\}$ is possible if the fate of x is already determined, e.g. $Pr[x < \rho_s] = 1$ if $x \in \{v_1, \ldots, v_{i-1}\}$ and is already a child of ρ_s in T'. Let $U = \{a, b, c, d\} \cap \{v_{i+1}, \ldots, v_m\}$, i.e.the leaves in $\{a, b, c, d\}$ that have not been reinserted yet in T'. Observe that $Pr[ab|cd \in T']$ depends only on where the elements of U get reinserted in T'. For instance, suppose that $U = \{a, b, c, d\}$, and suppose that a, b, c become children of ρ_s and d a child of ρ_{m-s}, as in Figure 5. Then $ab|cd$ is a quartet of this tree if and only if the ρ_s node gets resolved into a tree that contains the $ab|c$ rooted triplet[4], which occurs with probability 1/3.

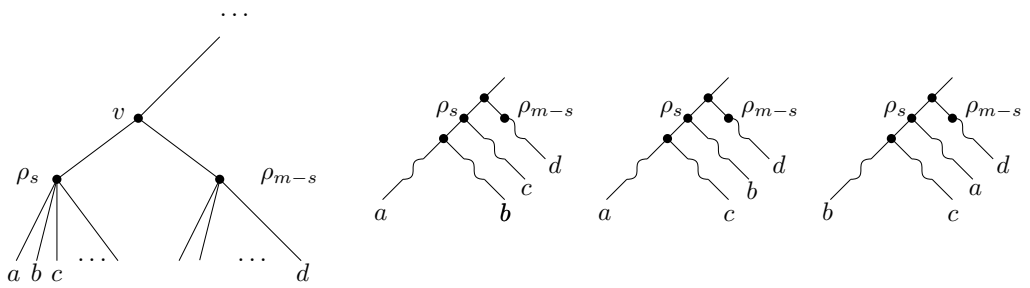

Fig. 5. The first tree on the left is the situation reached after completing ρ_s and ρ_{m-s}. The three trees to its right are the possible ways into which a, b, c and d can resolve, each occurring with equal probabililty (squiggly edges means that there could be other nodes on the path). Only the first resolution yields the $ab|cd$ quartet.

There are 16 possible ways that the U subset can be formed, and for each U there are at most $2^{4-|U|}$ ways that a, b, c, d could already inserted in T' and at most $2^{|U|}$ ways to reinsert the elements of U. The point is that there are a constant number of possible cases. There seems to be no general formula to handle all of these at once, but each case can be calculated by hand. We show how to handle one representative case, and the other ones are similar. Suppose that $U = \{b, c\}$ and that we have already that $a < \rho_s$ and $d < \rho_{m-s}$ already in T' because they were inserted. Then

[4] We say $ab|c$ is a rooted triplet of a rooted tree T if the $a - b$ path in T does not contain all ancestors of a on the $a - c$ path in T

$$\begin{aligned} Pr[ab|cd \in T'] = & \frac{1}{3} \cdot Pr[a,b,c < \rho_s \text{ and } d < \rho_{m-s}] + \\ & 1 \cdot Pr[a,b < \rho_s \text{ and } c,d < \rho_{m-s}] + \\ & \frac{1}{3} Pr[a < \rho_s \text{ and } b,c,d < \rho_{m-s}] + \\ & 0 \cdot Pr[a,c < \rho_s \text{ and } b,c < \rho_{m-s}] \end{aligned}$$

where for $X \subset \{a,b,c,d\}$ and $Y = \{a,b,c,d\} \setminus X$, $Pr[X < \rho_s \text{ and } Y < \rho_{m-s}]$ depends exclusively on m and the number of children k_1 and k_2 of ρ_s and ρ_{m-s}. Moreover this probability can be computed in time $O(1)$. For instance, $Pr[a,b,c < \rho_s \text{ and } d < \rho_{m-s}] = \binom{m-k_1-k_2-2}{s-k_1-2} / \binom{m-k_1-k_2}{s-k_1} = ((k-s)(1+k-s))/((k+l-m)(1+k+l-m))$, which takes constant time to compute. Note that in this manner, $Pr[ab|cd]$ is always the sum of at most 16 terms, whether 3 or 4 or a,b,c,d are children of v. We omit further details, and proceed with the proof of Theorem 5.

Proof of Theorem 5. We have shown that Algorithm 1 achieves the desired number of quartets. It only remains to argue the time complexity of the algorithm. The main while loop creates one new internal node at each iteration, so there must be $O(n)$ iterations. Each iteration runs two inner loops that take $m^2 = O(n^2)$ times, so the overall complexity is n^2 times the time taken to compute $Pr[q \in T_{s,1}]$ and $Pr[q \in T_{s,2}]$ for each of the $O(n^4)$ quartets. To compute such a probability, it suffices to identify in which possible case $q = ab|cd$ falls into (which only requires checking which of a,b,c,d are children of v, and which have or have not been inserted yet) and calculate this probability as above. This can be implemented to take constant time. The total time complexity is therefore $O(n^6)$.

5.3 A more practical 1/2 approximation algorithm

As we mentioned before, the PTAS for WQC is by no means practical. Here, we aim to develop a more implementable algorithm that achieves reasonable approximation guarantees. We borrow ideas from [10] to show that a 1/2 approximation can be achieved by taking the best solution from either the 1/3 approximation to WQC, or a factor 2 approximation to WMQI, the minimization version of WQC (see below). For two unrooted binary trees T_1, T_2 on leaf set $\mathcal{X}$, denote $d_Q(T_1,T_2) = |Q(T_1) \setminus Q(T_2)|$. The WMQI problem is defined as follows:

Weighted Minimum Quartet Inconsistency (WMQI) problem
Input: a set of unrooted trees $\mathcal{T} = \{T_1, \ldots, T_t\}$ such that $\mathcal{L}(T_1) = \ldots = \mathcal{L}(T_t) = \mathcal{X}$.
Output: a binary tree M with $\mathcal{L}(M) = \mathcal{X}$ that minimizes $\sum_{T \in \mathcal{T}} d_Q(M,T)$.

Note that the WMQI problem is equivalent to finding a minimum (in the multiset sense) number of quartets to discard from $\mathcal{Q}$ so that it is compatible, meaning that there exists a tree containing the resulting set of quartets.

It is not hard to show that d_Q is a metric. In particular, d_Q satisfies the triangle inequality, i.e. for any 3 trees T_1, T_2, T_3 on the same leaf set, $d_Q(T_1,T_3) \leq d_Q(T_1,T_2) + d_Q(T_2,T_3)$. This leads to a factor 2 approximation algorithm for WMQI obtained by simply returning the best tree from the input. Intuitively, the input tree that is the closest to the others cannot be too far from the best solution, which is a median tree in the metric space. See [4] for details.

Theorem 6 ([4]). *The following is a factor* 2 *approximation algorithm for WMQI: output the tree* $T \in \mathcal{T}$ *that minimizes* $\sum_{T_i \in \mathcal{T}} d_Q(T,T_i)$.

In [4], the authors explain how to compute $d_Q(T_1,T_2)$ in time $O(n^2)$. Therefore the factor 2 approximation can be implemented to run in time $O(t^2n^2)$, by simply computing d_Q between every pair of trees.

Theorem 6 has a direct implication on the approximation guarantees of the ASTRAL algorithm in [21], an implementation of the work from Bryant and Steel [9]. This algorithm finds, in polynomial time, an optimal solution M for a restricted version of WMQI where every bipartition [5] of M is

[5] A bipartition of a tree T consists in the two leafsets of the two trees obtained by removing an edge of T.

also a bipartition in at least one of the input trees. The solution T returned by the algorithm of Theorem 6 above trivially satisfies this condition. Thus, M is at least as good as T, implying the following.

Corollary 1. *The ASTRAL algorithm is a factor* 2 *approximation for WMQI.*

We do not know whether the factor 2 is tight for the ASTRAL algorithm - we conjecture that ASTRAL can actually achieve a better approximation ratio. As shown in the rest of this section, this would have interesting applications for the approximability of WQC. Do note however that the ASTRAL algorithm cannot be a PTAS. In the example of Figure 4, the unique optimal solution does not have all its bipartitions in the input trees. This solution contains 180 quartets from the input, whereas ASTRAL returns a solution with 176 such quartets. It follows that ASTRAL cannot achieve a better approximation ratio than 176/180. It is worth noting that in practice, heuristic optimizations in ASTRAL allow it to find the optimal solution on this instance.

Now, although WQC and WMQI are not necessarily identical in terms of approximability, we show that WMQI can be used to approximate WQC - the proof is similar to that of [10, Theorem 2].

Theorem 7. *If WMQI can be approximated within a factor α, then WQC can be approximated within a factor $\beta = \alpha/(3\alpha - 2)$.*

Proof. Let $N := t\binom{n}{4}$, i.e. the total number of quartets in $\mathcal{Q}$, let p be the maximum number of quartets that a tree T can contain from $\mathcal{Q}$, and let d be the minimum number of quartets to discard from $\mathcal{Q}$ in order to attain compatibility (here p and d refer to multiset cardinalities). Here p stands for "preserve" and d for "discard". Note that $d = N - p$. We show that taking the best tree between the one obtained from the factor α algorithm for WMQI and the one obtained from the derandomized 1/3 algorithm' achieves a factor β for WQC. Suppose first that $p \leq N/(3\beta)$. By Theorem 5, we can obtain a tree containing at least $N/3$ quartets from $\mathcal{Q}$, and since $N/3 = \beta N/(3\beta) \geq \beta p$, it yields a solution to WQC within a factor β from optimal. Thus we may assume that $p > N/(3\beta) = N(3\alpha-2)/(3\alpha)$. Since we have an α approximation for WMQI, we may obtain a solution discarding at most $\alpha d = \alpha(N-p)$ quartets. This solution preserves at least $N-(\alpha(N-p)) = \alpha p+(1-\alpha)N$ quartets from $\mathcal{Q}$. We claim that this attains a factor β approximation. Suppose instead that $\alpha p + (1-\alpha)N < \beta p$. Then $p < (\alpha-1)N/(\alpha-\beta)$ which, with a little work, yields $p < N(3\alpha-2)/(3\alpha)$, contradicting our assumption on p. Thus, the WMQI approximation preserves at least βp quartets. □

Combined with Theorem 6 and letting $\alpha = 2$ in Theorem 7 one can compute $d_Q(T_1, T_2)$ for every pair of trees, compute all the $O(tn^4)$ quartet weights, send these weights to the derandomization, and compare the two obtained solutions. This yields the following.

Corollary 2. *WQC can be approximated within a factor* 1/2 *in time $O(t^2n^2 + tn^4 + n^6)$.*

6 Fixed-parameter tractability of WQC

To our knowledge, there are no FPT algorithms that are known for WQC. One difficulty lies in finding a relevant parameter for the study of the problem. For instance, it is not too difficult to show that WQC is FPT in q, the number of quartets to remove from the input set $\mathcal{Q}$ to attain compatibiltiy, but this parameter is expected to be high in practice. In this section, we present some basic alternatives, with the goal of initiating a discussion on the fixed-parameter tractability of WQC.

We describe how, based on previous results on the minimum quartet incompatibility problem on complete sets, WQC can be solved in time $O(4^{d+k_2+k_3}n+n^4)$. Here k_2 and k_3 are the number of quadsets that have 2 and 3 dominant quartets (i.e. that occur with equal frequency), respectively, and d is the number of *strictly* dominant quartets that we are allowed to reject (i.e. that are not in an optimal solution). The algorithm makes direct use of the Gramm-Niedermeyer algorithm [15], henceforth called the GN algorithm.

The GN algorithm solves the following problem: given a *complete set* of quartets Q, find, if it exists, a complete and compatible set of quartets Q' such that at most d' quartets of Q' are not in the input set Q (i.e. $|Q' \setminus Q| \leq d'$). This is accomplished by repeatedly applying the following theorem:

Theorem 8 ([15]). *Let Q be a complete set of quartets. Then Q is compatible if and only if for each set of five taxa $\{a, b, c, d, e\} \subseteq \mathcal{X}$, $ab|cd \in Q$ implies $ab|ce \in Q$ or $ae|cd \in Q$.*

The idea behind the GN algorithm is as follows: find a set of five taxa $\{a, b, c, d, e\}$ that does not satisfy the condition of Theorem 8, then correct the situation by branching into the four possible choices:

1. remove $ab|cd$ from Q and add $ac|bd$ to Q;
2. remove $ab|cd$ from Q and add $ad|bc$ to Q;
3. remove $\{ac|be, ae|bc\} \cap Q$ from Q and add $ab|ce$ to Q;
4. remove $\{ac|de, ad|ce\} \cap Q$ from Q and add $ae|cd$ to Q.

The quartets added to Q will not be questioned in the following branchings. With some optimization, this leads to a $O(4^{d'}n + n^4)$ FPT algorithm.

In [15], the authors also note that this algorithm can be extended to sets of quartets Q that contain ambiguous quadsets, i.e. sets $\{a, b, c, d\}$ for which 2 or 3 of the possible quartets on $\{a, b, c, d\}$ are in Q. Suppose there are k'_2 and k'_3 quadsets that have 2 and 3 quartets in Q, respectively. The modified algorithm then, in a first phase, branches into the $2^{k'_2}3^{k'_3}$ ways of choosing one quartet per such quadset, thereby obtaining a complete set of quartets for each possibility. The GN algorithm is thus applied to the so-obtained complete sets. This yields a $O(2^{k'_2} \cdot 3^{k'_3} \cdot 4^{d'}n + n^4)$ algorithm.

It is not hard to see that this gives an FPT algorithm for WQC, where the parameter k'_2 (resp. k'_3) is the number of quadsets such that 2 (resp. 3) possible quartets appear in the input trees, and d' is the number of quartets $ab|cd$ that appear in every input tree, and that we are allowed to discharge. Note however that, in the consensus setting, there is no reason to believe that k'_2 and k'_3 are low - in we fact we believe that $k'_2 + k_3$,$'$ typically takes values in $\Theta(n^4)$. One reason is that even the slightest amount of noise on a quadset makes it included in the count of either k'_2 or k'_3 (e.g. if $t - 1$ trees agree on $ab|cd$ and only one contains $ac|bd$).

The GN algorithm can, however, be used on a more interesting set of parameters. Define k_2 (resp. k_3) as the number of quadsets that have exactly 2 (resp. 3) dominant quartets, and let d be the number of strictly dominant quartets that we are allowed to discharge. It is reasonable to believe that, if each tree of the input is close to the true tree T^*, most "true" quartets will appear as strictly dominant in the input, and there should not be too many ambiguous quadsets. There is a simple algorithm achieving time $O(4^{d+k_2+k_3}n + n^4)$. Construct a complete set of quartets Q as follows: for each quadset $\{a, b, c, d\}$, choose a dominant quartet on $\{a, b, c, d\}$ and add it to Q (if multiple choices are possible, choose arbitrarily). Then, run the GN algorithm on Q with the following modification: each time a quartet q is removed from Q and replaced by another quartet q', decrement either d, k_2 or k_3, depending on whether q belongs to a quadset with $1, 2$ or 3 dominant quartets. We know that the latter two values will never be decremented more than $k_2 + k_3$ times. It follows that if there exists a complete and compatible set of quartets Q' such that at most d strictly dominant quartets are rejected, then the modified algorithm will find it. It should be noted however that finding such a set Q' does not guarantee that the corresponding tree is an optimal solution. Indeed, since quartets are weighted, two solutions Q' and Q'' may both reject only d strictly dominant quartets, yet one has higher weight than the other. However, the correctness of the algorithm follows from the fact that the GN algorithm finds the set of *every* solution discarding at most d dominant quartets - and thus it suffices to pick the solution from this set that has optimal weight. We summary the above discussion formally as a corollary of [15].

Corollary 3. *The WQC problem can be solved in time $O(4^{d+k_2+k_3}n + n^4)$, where d is the number of strictly dominant quartets not in an optimal solution, and k_2, k_3 are the number of quadsets that have 2 and 3 dominant quartets, respectively.*

We finally mention that the FPT algorithms published in [11] are improved versions of the GN algorithm, can also return every solution and thus can be modified in the same manner. These yield FPT algorithms that can solve WQC in time $O(3.0446^{d+k_2+k_3}n+n^4)$ and $O(2.0162^{d+k_2+k_3}n^3+n^5)$.

7 Conclusion

In this paper, we have shown that the WQC problem is NP-hard, answering a question of [20] and [4]. In the latter, the authors also propose a variant of the problem in which the output tree T

is not required to be binary. In this case, one needs to assign a cost p for the unresolved quartets in the output. Our reduction can be extended to show that hardness holds for high enough p (e.g. we get WQC when $p = \infty$), but the complexity of the general case remains open. We have also shown that WQC admits a PTAS and can be approximated within a factor 1/2. As we have shown, improvements in the WMQI problem can immediately lead to better approximation algorithms for WQC, and so it remains to investigate the approximability of WMQI. The fixed-parameter tractability aspects of WQC also deserve further investigation. This would require identifying some structural properties that are present in the consensus setting and that can be used for designing practical exact algorithms. But as we have shown, this might not be an easy task, as many properties which seem reasonable for the consensus setting do not hold.

Acknowledgements

This work was partially supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) and the Mitacs Globalink Campus France program.

References

1. A. V. Aho, Y. Sagiv, T. G. Szymanski, and J. D. Ullman, *Inferring a tree from lowest common ancestors with an application to the optimization of relational expressions*, SIAM Journal on Computing, 10 (1981), pp. 405–421.
2. S. Arora, D. Karger, and M. Karpinski, *Polynomial time approximation schemes for dense instances of np-hard problems*, Journal of computer and system sciences, 58 (1999), pp. 193–210.
3. M. S. Bansal, J. Dong, and D. Fernández-Baca, *Comparing and aggregating partially resolved trees*, in Latin American Symposium on Theoretical Informatics, Springer, 2008, pp. 72–83.
4. M. S. Bansal, J. Dong, and D. Fernández-Baca, *Comparing and aggregating partially resolved trees*, Theoretical Computer Science, 412 (2011), pp. 6634–6652.
5. J.-P. Barthélemy and F. R. McMorris, *The median procedure for n-trees*, Journal of Classification, 3 (1986), pp. 329–334.
6. V. Berry, D. Bryant, T. Jiang, P. Kearney, M. Li, T. Wareham, and H. Zhang, *A practical algorithm for recovering the best supported edges of an evolutionary tree (extended abstract)*, in Proceedings of the Eleventh Annual ACM-SIAM Symposium on Discrete Algorithms, Society for Industrial and Applied Mathematics, 2000, pp. 287–296.
7. V. Berry, T. Jiang, P. Kearney, M. Li, and T. Wareham, *Quartet cleaning: Improved algorithms and simulations*, in Proceedings of the 7th Annual European Symposium on Algorithms, J. Nešetřil, ed., Springer Berlin Heidelberg, 1999, pp. 313–324.
8. D. Bryant, *A classification of consensus methods for phylogenetics*, DIMACS Series in Discrete Mathematics and Theoretical Computer Science, 61 (2003), pp. 163–184.
9. D. Bryant and M. Steel, *Constructing optimal trees from quartets*, Journal of Algorithms, 38 (2001), pp. 237–259.
10. J. Byrka, S. Guillemot, and J. Jansson, *New results on optimizing rooted triplets consistency*, Discrete Applied Mathematics, 158 (2010), pp. 1136–1147.
11. M.-S. Chang, C.-C. Lin, and P. Rossmanith, *New fixed-parameter algorithms for the minimum quartet inconsistency problem*, Theory of Computing Systems, 47 (2010), pp. 342–367.
12. A. Dress, K. T. Huber, and J. Koolen, *Basic phylogenetic combinatorics*, Cambridge University Press, 2012.
13. J. Felsenstein, *Inferring phylogenies*, Sinauer Associates Sunderland, 2004.
14. Z. Galil and N. Megiddo, *Cyclic ordering is NP-complete*, Theoretical Computer Science, 5 (1977), pp. 179–182.
15. J. Gramm and R. Niedermeier, *Minimum quartet inconsistency is fixed parameter tractable*, in Proceedings of the 12th Annual Symposium of Combinatorial Pattern Matching, A. Amir, ed., Springer Berlin Heidelberg, 2001, pp. 241–256.
16. J. Jansson, *On the complexity of inferring rooted evolutionary trees*, Electronic Notes in Discrete Mathematics, 7 (2001), pp. 50–53.
17. T. Jiang, P. Kearney, and M. Li, *A polynomial time approximation scheme for inferring evolutionary trees from quartet topologies and its application*, SIAM Journal on Computing, 30 (2001), pp. 1942–1961.
18. T. Margush and F. R. McMorris, *Consensusn-trees*, Bulletin of Mathematical Biology, 43 (1981), pp. 239–244.

19. F. R. McMorris, D. B. Meronk, and D. Neumann, *A view of some consensus methods for trees*, in Numerical Taxonomy, Springer, 1983, pp. 122–126.
20. S. Mirarab, *Novel scalable approaches for multiple sequence alignment and phylogenomic reconstruction*, PhD thesis, University of Texas at Austin, 2015.
21. S. Mirarab, R. Reaz, M. S. Bayzid, T. Zimmermann, M. S. Swenson, and T. Warnow, *ASTRAL: genome-scale coalescent-based species tree estimation*, Bioinformatics, 30 (2014), pp. i541–i548.
22. A. Morgado and J. Marques-Silva, *A pseudo-boolean solution to the maximum quartet consistency problem*, arXiv preprint arXiv:0805.0202, (2008).
23. A. Morgado and J. Marques-Silva, *Combinatorial optimization solutions for the maximum quartet consistency problem*, Fundamenta Informaticae, 102 (2010), pp. 363–389.
24. R. R. Sokal and F. J. Rohlf, *Taxonomic congruence in the leptopodomorpha re-examined*, Systematic Zoology, 30 (1981), pp. 309–325.
25. M. Steel, *The complexity of reconstructing trees from qualitative characters and subtrees*, Journal of Classification, 9 (1992), pp. 91–116.
26. G. Wu, J.-H. You, and G. Lin, *A lookahead branch-and-bound algorithm for the maximum quartet consistency problem*, in International Workshop on Algorithms in Bioinformatics, Springer, 2005, pp. 65–76.
27. G. Wu, J.-H. You, and G. Lin, *Quartet-based phylogeny reconstruction with answer set programming*, IEEE/ACM Transactions on Computational Biology and Bioinformatics, 4 (2007), pp. 139–152.
28. L. Xin, B. Ma, and K. Zhang, *A new quartet approach for reconstructing phylogenetic trees: Quartet joining method*, in International Computing and Combinatorics Conference, Springer, 2007, pp. 40–50.