Determining the Consistency of Resolved Triplets and Fan Triplets

Jesper Jansson[1,2(✉)], Andrzej Lingas[3], Ramesh Rajaby[4,5], and Wing-Kin Sung[4,6]

[1] Laboratory of Mathematical Bioinformatics, ICR, Kyoto University, Gokasho, Uji, Kyoto 611-0011, Japan
jj@kuicr.kyoto-u.ac.jp
[2] Department of Computing, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong, China
[3] Department of Computer Science, Lund University, 22100 Lund, Sweden
Andrzej.Lingas@cs.lth.se
[4] School of Computing, National University of Singapore, 13 Computing Drive, Singapore 117417, Singapore
ramesh.rajaby@gmail.com, ksung@comp.nus.edu.sg
[5] NUS Graduate School for Integrative Sciences and Engineering, National University of Singapore, 28 Medical Drive, Singapore 117456, Singapore
[6] Genome Institute of Singapore, 60 Biopolis Street, Genome, Singapore 138672, Singapore

Abstract. The $\mathcal{R}^{+-}\mathcal{F}^{+-}$ Consistency problem takes as input two sets R^+ and R^- of resolved triplets and two sets F^+ and F^- of fan triplets, and asks for a distinctly leaf-labeled tree that contains all elements in $R^+ \cup F^+$ and no elements in $R^- \cup F^-$ as embedded subtrees, if such a tree exists. This paper presents a detailed characterization of how the computational complexity of the problem changes under various restrictions. Our main result is an efficient algorithm for dense inputs satisfying $R^- = \emptyset$ whose running time is linear in the size of the input and therefore optimal.

Keywords: Phylogenetic tree · Rooted triplets consistency · Algorithm · Computational complexity

1 Introduction

Phylogenetic trees have been used by biologists for more than 150 years to describe evolutionary history. In the last 50 years, many methods for systematically reconstructing phylogenetic trees from different kinds of data have been proposed [10,23]. In general, inferring a reliable phylogenetic tree is a time-consuming task for large data sets, but the *supertree approach* (see, e.g., [2,3]) may in many cases provide a reasonable compromise between accuracy and computational efficiency by way of divide-and-conquer: first, infer a set of trees for small, overlapping subsets of the species using a computationally expensive method such as maximum likelihood [7,10], and then merge all the small trees

S.C. Sahinalp (Ed.): RECOMB 2017, LNBI 10229, pp. 82–98, 2017.
DOI: 10.1007/978-3-319-56970-3_6

into one big tree with some combinatorial algorithm. In this context, the fundamental problem of determining if a given set of *resolved triplets* (rooted, binary phylogenetic trees with exactly three leaf labels each) can be combined without conflicts, and if so, constructing such a tree can be solved efficiently by Aho *et al.*'s BUILD algorithm from [1]. BUILD has therefore been extended in various ways [8,12,13,16,18–21,24], for example, to also allow *fan triplets* (rooted, non-binary phylogenetic trees with three leaf labels each) or *forbidden* resolved triplets in the input, and to handle related optimization problems where the input may contain errors and the objective is to find a tree that satisfies as much of the input as possible (for details, see [6,9] and the references therein).

Below, we investigate how the computational complexity of the basic decision problem varies according to which types of inputs are allowed and present some new results that expose the boundary between efficiently solvable and intractable versions of the problem.

1.1 Problem Definitions

A *phylogenetic tree* is a rooted, unordered, distinctly leaf-labeled tree in which every internal node has at least two children. (From here on, phylogenetic trees are simply referred to as "trees" and every leaf in a tree is identified with its corresponding leaf label.) For any tree T, the set of all nodes in T is denoted by $V(T)$ and the set of all leaf labels occurring in T is denoted by $\Lambda(T)$. The *degree of a node* $u \in V(T)$ is the number of children of u, and the *degree of* T is the maximum degree of all nodes in $V(T)$. For any $u, v \in V(T)$, $lca^T(u, v)$ denotes the lowest common ancestor in T of u and v.

A *rooted triplet* is a tree with precisely three leaves. Let t be any rooted triplet and suppose that $\Lambda(t) = \{x, y, z\}$. If t is binary then t is called a *resolved triplet* and we write $t = xy|z$, where $lca^t(x, y)$ is a proper descendant of $lca^t(x, z) = lca^t(y, z)$. On the other hand, if t is not binary then t is called a *fan triplet* and we write $t = x|y|z$. Note that there are four different rooted triplets leaf-labeled by $\{x, y, z\}$, namely $xy|z$, $xz|y$, $yz|x$, and $x|y|z$.

For any tree T and $\{x, y, z\} \subseteq \Lambda(T)$, the resolved triplet $xy|z$ is *consistent with* T if $lca^T(x, y)$ is a proper descendant of $lca^T(x, z) = lca^T(y, z)$. Similarly, the fan triplet $x|y|z$ is *consistent with* T if $lca^T(x, y) = lca^T(x, z) = lca^T(y, z)$. Finally, for any tree T, let $T||_{\{x,y,z\}}$ be the rooted triplet with leaf label set $\{x, y, z\}$ that is consistent with T, and let $t(T)$ be the set of *all* rooted triplets (resolved triplets as well as fan triplets) consistent with T, i.e., define $t(T) = \{T||_{\{x,y,z\}} : \{x, y, z\} \subseteq \Lambda(T)\}$.

The problem studied in this paper is:

> The $\mathcal{R}^{+-}\mathcal{F}^{+-}$ Consistency problem:
> Given two sets R^+ and R^- of resolved triplets and two sets F^+ and F^- of fan triplets over a leaf label set L, output a tree T with $\Lambda(T) = L$ such that $R^+ \cup F^+ \subseteq t(T)$ and $(R^- \cup F^-) \cap t(T) = \emptyset$, if such a tree exists; otherwise, output *null*.

In other words, R^+ and F^+ specify rooted triplets that are required to be embedded in the output tree, while R^- and F^- are forbidden rooted triplets. See Fig. 1 for two examples. Throughout the paper, we use n to denote the cardinality of the input leaf label set L.

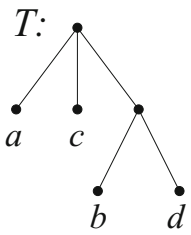

Fig. 1. As an example, consider the following instance of the $\mathcal{R}^{+-}\mathcal{F}^{+-}$ Consistency problem: $L = \{a, b, c, d\}$, $R^+ = \emptyset$, $R^- = \{cd|a\}$, $F^+ = \{a|b|c\}$, and $F^- = \{b|c|d\}$. The shown tree T satisfies $t(T) = \{a|b|c, bd|a, a|c|d, bd|c\}$, so $R^+ \cup F^+ \subseteq t(T)$ and $(R^- \cup F^-) \cap t(T) = \emptyset$ hold. Thus, T is a valid solution. As another example, if L, R^+, R^-, and F^- are the same as above but F^+ is changed to $F^+ = \{a|b|c|, a|b|d\}$ then the answer is *null*.

The special cases of the $\mathcal{R}^{+-}\mathcal{F}^{+-}$ Consistency problem where one or more of the four input sets R^+, R^-, F^+, F^- are empty will also be denoted by removing the corresponding "+" and "−" symbols from the problem name. For example, the $\mathcal{R}^-\mathcal{F}^+$ Consistency problem requires that $R^+ = F^- = \emptyset$. To simplify the notation, if $R^+ = R^- = \emptyset$ then we omit the "$\mathcal{R}$", and analogously for "$\mathcal{F}$"; e.g., $\mathcal{R}^-$ means $R^+ = F^+ = F^- = \emptyset$. Ignoring the trivial case where all of R^+, R^-, F^+, F^- are empty, this yields exactly 14 problem variants in addition to the original problem. Our goal is to establish the computational complexity of all these problem variants as well as some other potentially useful special cases. Because of space constraints, the proofs of Lemmas 5 and 6 have been deferred to the journal version.

1.2 Overview of Old and New Results

Aho *et al.* [1] presented a polynomial-time algorithm named BUILD that solves the $\mathcal{R}^+$ Consistency problem, and Ng and Wormald [18] extended BUILD to solve $\mathcal{R}^+\mathcal{F}^+$ Consistency in polynomial time. Using a similar approach, He *et al.* [13] showed how to solve $\mathcal{R}^{+-}$ Consistency in polynomial time. As for negative results, Bryant [4, Theorem 2.20] proved that $\mathcal{R}^-$ Consistency is NP-hard under the additional constraint that the output tree is binary. Three direct consequences of these known results are given in Sect. 2 (Lemmas 1, 2, and 3). In Sect. 3, we shall prove that the $\mathcal{F}^{+-}$ Consistency problem is NP-hard (Theorem 1). Significantly, Lemmas 1, 2, and 3 together with Theorem 1 then provide a complete characterization of the polynomial-time solvability of

all 15 variants of the $\mathcal{R}^{+-}\mathcal{F}^{+-}$ Consistency problem defined in Sect. 1.1 since each of the remaining problem variants is either a special case of a polynomial-time solvable problem variant or a generalization of an NP-hard one. See Table 1.

Table 1. Overview of the computational complexity of the 15 different variants of the $\mathcal{R}^{+-}\mathcal{F}^{+-}$ Consistency problem. "P" means solvable in polynomial time. The results written in bold text are due to [1,13,18].

Consistency	$\emptyset$	$\mathcal{F}^{+}$	$\mathcal{F}^{-}$	$\mathcal{F}^{+-}$
$\emptyset$	$\times$	**P**	P	NP-hard (Theorem 1)
$\mathcal{R}^{+}$	**P**	**P**	P (Lemma 2)	NP-hard
$\mathcal{R}^{-}$	**P**	P	NP-hard (Lemma 3)	NP-hard
$\mathcal{R}^{+-}$	**P**	P (Lemma 1)	NP-hard	NP-hard

Motivated by these observations, we then try to identify some way of restricting the $\mathcal{R}^{+-}\mathcal{F}^{+-}$ Consistency problem that leads to more efficiently solvable problem variants. One natural restriction is to require the degree of the output tree to be at most D for some integer $D \geq 2$; unfortunately, Sect. 4 demonstrates that this generally makes the problems *harder*. See Table 2 for a summary. In particular, Theorem 2 proves that even $\mathcal{F}^{+}$ Consistency is NP-hard when restricted to degree-D trees for every fixed $D \geq 4$. Furthermore, by Corollary 2, D-bounded degree $\mathcal{R}^{-}$ Consistency becomes NP-hard for every fixed $D \geq 2$. The only efficiently solvable problem variants that we know of are covered by Corollary 1, stating that D-bounded degree $\mathcal{R}^{+}\mathcal{F}^{-}$ Consistency remains polynomial-time solvable for every $D \geq 2$.

Table 2. The complexity of $\mathcal{R}^{+-}\mathcal{F}^{+-}$ Consistency when the output tree is required to have degree at most D. "NP-hard*" (with an asterisk) means NP-hard for every fixed $D \geq 4$ and trivially polynomial-time solvable for $D = 2$ while the complexity for $D = 3$ is still open.

Bounded degree Consistency	$\emptyset$	$\mathcal{F}^{+}$	$\mathcal{F}^{-}$	$\mathcal{F}^{+-}$
$\emptyset$	$\times$	NP-hard* (Theorem 2)	P	NP-hard*
$\mathcal{R}^{+}$	P	NP-hard*	P (Corollary 1)	NP-hard*
$\mathcal{R}^{-}$	NP-hard (Corollary 2)	NP-hard	NP-hard	NP-hard
$\mathcal{R}^{+-}$	NP-hard	NP-hard	NP-hard	NP-hard

Therefore, we need to find another way to restrict the problem. For this purpose, Sect. 5 considers inputs that are *dense* in the sense that for each $L' \subseteq L$

with $|L'| = 3$, at least one rooted triplet t with $\Lambda(t) = L'$ is specified in R^+, R^-, F^+, or F^-. As shown in [9], the maximization version of $\mathcal{R}^{+-}\mathcal{F}^{+-}$ Consistency (whose objective is to output a tree T with $\Lambda(T) = L$ maximizing the value of $|T(R^+ \cup F^+)| + |(R^- \cup F^-) \setminus T(R^- \cup F^-)|$, where $T(X)$ for any set X of rooted triplets denotes the subset of X consistent with T) admits a polynomial-time approximation scheme (PTAS) when restricted to dense inputs, whereas no such PTAS is known for the non-dense case; in fact, the non-dense case of the maximization problem is APX-complete [5, Proposition 2]. This gives us some hope that $\mathcal{R}^{+-}\mathcal{F}^{+-}$ Consistency may be easier for dense inputs. Although $\mathcal{R}^{-}\mathcal{F}^{-}$ Consistency turns out to be NP-hard in the dense case by Lemma 3, $\mathcal{R}^{+}\mathcal{F}^{+-}$ Consistency restricted to dense inputs indeed admits a polynomial-time algorithm (Theorem 4), and moreover, its time complexity is $O(n^3)$ which is optimal because the size of a dense input is $\Omega(n^3)$. The situation for dense inputs is summarized in Table 3.

Table 3. The complexity of $\mathcal{R}^{+-}\mathcal{F}^{+-}$ Consistency restricted to dense inputs. The results written in bold text are due to [1,13,18].

Dense Consistency	$\emptyset$	$\mathcal{F}^+$	$\mathcal{F}^-$	$\mathcal{F}^{+-}$
$\emptyset$	$\times$	**P**	P	P
$\mathcal{R}^+$	**P**	**P**	P	P (Theorem 4)
$\mathcal{R}^-$	**P**	P	NP-hard (Lemma 3)	NP-hard
$\mathcal{R}^{+-}$	**P**	P (Lemma 1)	NP-hard	NP-hard

2 Preliminaries

This section lists some simple results that follow immediately from previous work.

Lemma 1. *The $\mathcal{R}^{+-}\mathcal{F}^{+}$ Consistency problem is solvable in polynomial time.*

Proof. For any instance of $\mathcal{R}^{+-}\mathcal{F}^{+}$ Consistency, by removing each fan triplet of the form $x|y|z$ from F^+ and inserting the three resolved triplets $xy|z$, $xz|y$, $yz|x$ into R^-, one obtains an equivalent instance of $\mathcal{R}^{+-}$ Consistency to which the *MTT* algorithm in [13] can be applied. By [13], the running time becomes $O(|R^+| \cdot n + (|R^-| + |F^+|) \cdot n \log n + n^2 \log n)$. □

Lemma 2. *The $\mathcal{R}^{+}\mathcal{F}^{-}$ Consistency problem is solvable in polynomial time.*

Proof. For any instance of $\mathcal{R}^{+}\mathcal{F}^{-}$ Consistency, run the BUILD algorithm [1] with input R^+ and let T be its output. If T is not *null* then, as long as T is non-binary, select any internal node u with degree larger than two and any two children c_1 and c_2 of u, remove the edges $\{u, c_1\}$ and $\{u, c_2\}$, create a new child v of u, and insert the edges $\{v, c_1\}$ and $\{v, c_2\}$. Finally, output T. Using a

fast implementation of BUILD from [14] along with an improved data structure for supporting dynamic graph connectivity queries [15] (see [17] for details), the $\mathcal{R}^+\mathcal{F}^-$ Consistency problem becomes solvable in $\min\{O(|R^+| \cdot \log^2 n + |F^-| + n), O(|R^+| + |F^-| + n^2 \log n)\}$ time. □

Lemma 3. *The $\mathcal{R}^-\mathcal{F}^-$ Consistency problem is NP-hard, even if restricted to dense inputs.*

Proof. According to Bryant [4, Theorem 2.20], the $\mathcal{R}^-$ Consistency problem is NP-hard when the output tree is constrained to be binary. Given any instance of Bryant's version of the problem consisting of a set R of (forbidden) resolved triplets, construct an equivalent instance of the $\mathcal{R}^-\mathcal{F}^-$ Consistency problem by letting $R^- = R$ and letting F^- be the set of all $\binom{|L|}{3}$ fan triplets over the leaf label set $L = \bigcup_{t \in R} \Lambda(t)$ (note that F^- is dense). Since the reduction is a polynomial-time reduction, the latter problem is also NP-hard. □

3 $\mathcal{F}^{+-}$ Consistency is NP-Hard

Here, we prove that the $\mathcal{F}^{+-}$ Consistency problem is NP-hard by giving a polynomial-time reduction from the NP-hard problem Set Splitting (see, e.g., [11]):

> Set Splitting:
> Given a set $S = \{s_1, s_2, \ldots, s_n\}$ and a collection $\mathcal{C} = \{C_1, C_2, \ldots, C_m\}$ of subsets of S where $|C_j| = 3$ for every $C_j \in \mathcal{C}$, does $(S, \mathcal{C})$ have a set splitting, i.e., can S be partitioned into two disjoint subsets S' and S'' such that for every $C_j \in \mathcal{C}$ it holds that C_j is not a subset of S' and C_j is not a subset of S''?

We now describe the reduction. Given an instance $(S, \mathcal{C})$ of Set Splitting, where we assume w.l.o.g. that $\bigcup_{C_j \in \mathcal{C}} C_j = S$, construct an instance of $\mathcal{F}^{+-}$ Consistency as follows:

- Let $L = S \cup \{x, y, z', z''\} \cup \{\alpha_j, \beta_j, \gamma_j : 1 \leq j \leq m\}$ be the leaf label set.
- For $1 \leq j \leq m$, denote $C_j = \{c_j^1, c_j^2, c_j^3\}$, where $c_j^1, c_j^2, c_j^3 \in S$. Define $F^+ = \{x|y|z',\ x|y|z'',\ x|z'|z''\} \cup \{x|y|s_i : s_i \in S\} \cup \{x|c_j^1|\alpha_j,\ c_j^2|c_j^3|\alpha_j,\ x|c_j^2|\beta_j,\ c_j^1|c_j^3|\beta_j,\ x|c_j^3|\gamma_j,\ c_j^1|c_j^2|\gamma_j : 1 \leq j \leq m\}$.
- Define $F^- = \{s_i|z'|z'' : s_i \in S\}$.

The next lemma ensures the correctness of the reduction:

Lemma 4. *$(S, \mathcal{C})$ has a set splitting if and only if there exists a tree T with $\Lambda(T) = L$ such that $F^+ \subseteq t(T)$ and $F^- \cap t(T) = \emptyset$.*

Proof. $\Rightarrow$) Suppose that (S', S'') is a set splitting of $(S, \mathcal{C})$. Create a tree T with $\Lambda(T) = L$ whose root has $4 + 2m$ children in the following way. First, let two leaves labeled by x and y as well as two internal nodes u' and u'' be children of the root of T, and attach $1 + |S'|$ leaves labeled by $\{z'\} \cup S'$ and $1 + |S''|$ leaves labeled by $\{z''\} \cup S''$ as children of u' and u'', respectively. Next, for each $C_j \in \mathcal{C}$, exactly two of the three elements c_j^1, c_j^2, c_j^3 have the same parent in T because (S', S'') is a set splitting; let u_j be this common parent. By definition, $u_j \in \{u', u''\}$. The three leaves $\alpha_j, \beta_j, \gamma_j$ are inserted into T according to which one of these cases holds:

- c_j^1 and c_j^2 have the same parent u_j: Attach a leaf labeled by γ_j as a child of u_j and two leaves labeled by α_j, β_j as children of the root of T.
- c_j^1 and c_j^3 have the same parent u_j: Attach a leaf labeled by β_j as a child of u_j and two leaves labeled by α_j, γ_j as children of the root of T.
- c_j^2 and c_j^3 have the same parent u_j: Attach a leaf labeled by α_j as a child of u_j and two leaves labeled by β_j, γ_j as children of the root of T.

It is straightforward to verify that $F^+ \subseteq t(T)$ and $F^- \cap t(T) = \emptyset$.

$\Leftarrow$) Suppose that T is a tree with $\Lambda(T) = L$ such that $F^+ \subseteq t(T)$ and $F^- \cap t(T) = \emptyset$. Let $r = lca^T(x, y)$. The node r must be the root of T because (1) $x|y|q \in t(T)$ for all $q \in \{z', z''\} \cup S$ and (2) for each $\delta_j \in \{\alpha_j, \beta_j, \gamma_j\}$, $1 \leq j \leq m$, there exists an $s_i \in S$ such that $x|s_i|\delta_j \in t(T)$. Let T' (resp. T'') be the subtree of T rooted at a child of r which contains z' (resp. z''); then, $T' \neq T''$ since $x|z'|z'' \in t(T)$ and x cannot belong to T' due to $x|y|z' \in t(T)$. Furthermore, each $s_i \in S$ belongs to either T' or T'' since $s_i|z'|z'' \notin t(T)$.

Next, we show by contradiction that for every $C_j \in \mathcal{C}$, exactly one or two of the three elements c_j^1, c_j^2, c_j^3 belong to T' (and hence, that exactly one or two of the three elements belong to T''). Suppose that all three elements belong to T'. The condition $c_j^1|c_j^2|\gamma_j$, $c_j^1|c_j^3|\beta_j$, $c_j^2|c_j^3|\alpha_j \in t(T)$ implies that $\alpha_j, \beta_j, \gamma_j$ also belong to T'. But then $x|c_j^1|\alpha_j$, $x|c_j^2|\beta_j$, $x|c_j^3|\gamma_j$ cannot be consistent with T, which is impossible. In the same way, all three elements cannot belong to T''.

In summary, selecting $S' = \Lambda(T') \cap S$ and $S'' = \Lambda(T'') \cap S$ yields a set splitting of $(S, \mathcal{C})$. □

Since the reduction can be carried out in polynomial time, Lemma 4 gives:

Theorem 1. *The $\mathcal{F}^{+-}$* Consistency *problem is NP-hard.*

4 *D*-Bounded Degree $\mathcal{R}^{+-}\mathcal{F}^{+-}$ Consistency

We now consider the computational complexity of D-bounded degree $\mathcal{R}^{+-}\mathcal{F}^{+-}$ Consistency, i.e., where the degree of the output tree is constrained to be at most D for some integer $D \geq 2$. First, by noting that the method in the proof of Lemma 2 always outputs a binary tree, we have:

Corollary 1. *For every fixed $D \geq 2$, the D-bounded degree $\mathcal{R}^{+}\mathcal{F}^{-}$* Consistency *problem is solvable in polynomial time.*

In contrast, many other variants become NP-hard, as shown in the rest of this section.

4.1 D-Bounded Degree $\mathcal{F}^+$ Consistency is NP-Hard

This subsection proves that for every fixed integer $D \geq 4$, the D-bounded degree $\mathcal{F}^+$ Consistency problem is NP-hard. The proof relies on a simple polynomial-time reduction from the K-Coloring problem, which is NP-hard for every fixed $K \geq 3$ (see [11]):

> K-Coloring:
> Given an undirected, connected graph $G = (V, E)$ and a positive integer K, does G have a K-coloring, i.e., can V be partitioned into K (possibly empty) disjoint subsets $V_1, V_2, \ldots, V_K$ such that for every $\{u, v\} \in E$ it holds that $i \neq j$ where $u \in V_i$ and $v \in V_j$?

The reduction is as follows. Given an instance of $(D-1)$-Coloring, create an instance of D-bounded degree $\mathcal{F}^+$ Consistency by setting $L = V \cup \{x\}$ and $F^+ = \big\{x|u|v \,:\, \{u, v\} \in E\big\}$.

Lemma 5. *G has a $(D-1)$-coloring if and only if there exists a tree T with degree at most D and $\Lambda(T) = L$ such that $F^+ \subseteq t(T)$.*

Theorem 2. *For every fixed $D \geq 4$, the D-bounded degree $\mathcal{F}^+$ Consistency problem is NP-hard.*

4.2 D-Bounded Degree $\mathcal{R}^-$ Consistency is NP-Hard

Bryant [4, Theorem 2.20] proved that the D-bounded degree $\mathcal{R}^-$ Consistency problem is NP-hard for $D = 2$ by reducing from the following NP-hard problem (see, e.g., [11]):

> 3SAT:
> Given a set U of Boolean variables and a collection $C = \{C_1, C_2, \ldots, C_m\}$ of disjunctive clauses over U, each containing exactly 3 literals, is there a truth assignment for U that makes every clause in C true?

The main idea in Bryant's reduction is to represent every literal by a leaf label and define the forbidden resolved triplets so that in any valid tree, assigning true to all literals contained in one particular subtree rooted at a child of the root (and assigning false to the rest) results in a valid truth assignment. In this subsection, we adapt Bryant's proof to obtain an analogous result for the case $D = 3$ by introducing an additional leaf label x and defining a slightly more involved set of forbidden resolved triplets. More precisely, given an instance of 3SAT, we construct an instance of 3-bounded degree $\mathcal{R}^-$ Consistency with $L = U \cup \overline{U} \cup C \cup C' \cup \{x, t, f\}$ and $R^- = R_1 \cup R_2 \cup R_3 \cup R_4$, where $\overline{U} = \{\overline{u} \,:\, u \in U\}$, $C' = \{C'_j \,:\, C_j \in C\}$, and:

- $R_1 = \{tf|x,\ tx|f,\ fx|t\}$,
- $R_2 = \{u\overline{u}|x,\ ux|\overline{u},\ \overline{u}x|u,\ u\overline{u}|t,\ u\overline{u}|f \,:\, u \in U\}$,

- $R_3 = \{C_jC'_j|x,\ C_jx|C'_j,\ C'_jx|C_j,\ C_jC'_j|t,\ C_jC'_j|f\ :\ C_j \in C\}$, and
- $R_4 = \{u_jv_j|C_j,\ w_jC_j|t\ :\ C_j \in C\}$, where we write $C_j = (u_j \vee v_j \vee w_j)$ with $u_j, v_j, w_j \in U \cup \overline{U}$.

Note that R_4 is defined asymmetrically.

Lemma 6. *There is a truth assignment for U making every clause in C true if and only if there exists a tree T with degree at most 3 and $\Lambda(T) = L$ such that $R^- \cap t(T) = \emptyset$.*

Theorem 3. *For $D = 3$, the D-bounded degree $\mathcal{R}^-$ Consistency problem is NP-hard.*

Corollary 2. *For every fixed $D \geq 2$, the D-bounded degree $\mathcal{R}^-$ Consistency problem is NP-hard.*

Proof. For $D \in \{2, 3\}$, see above. For $D \geq 4$, the NP-hardness follows from Theorem 2 and the polynomial-time reduction which, for each fan triplet of the form $x|y|z$ in F^+ in any given instance of the D-bounded degree $\mathcal{F}^+$ Consistency problem, includes three resolved triplets $xy|z$, $xz|y$, $yz|x$ in R^-. □

5 An Optimal Algorithm for Dense $\mathcal{R}^+\mathcal{F}^{+-}$ Consistency

Recall from Sect. 1.2 that an input to $\mathcal{R}^{+-}\mathcal{F}^{+-}$ Consistency is called *dense* if, for every $L' \subseteq L$ with $|L'| = 3$, at least one rooted triplet t with $\Lambda(t) = L'$ is in R^+, R^-, F^+, or F^-. In this section, we present the main result of the paper, namely an algorithm called `DenseBuild` that solves the special case $\mathcal{R}^+\mathcal{F}^{+-}$ Consistency (i.e., where $R^- = \emptyset$) restricted to dense inputs, and show that its running time is $O(n^3)$, which is optimal. Two tools used by `DenseBuild` are the *fan graph* and the *clique graph*, defined and studied in Sect. 5.1. Algorithm `DenseBuild` is presented in Sect. 5.2.

According to Sect. 1.2, $\mathcal{R}^+\mathcal{F}^{+-}$ Consistency is NP-hard. Intuitively, the problem becomes easier for dense inputs because if T is a tree consistent with the input then the set $Z = \{x|y|z\ :\ x, y, z \in L$ and x, y, z belong to three different subtrees attached to the root of $T\}$ forms a subset of F^+, in which case F^+ contains enough information to partition L into the leaf label sets of the subtrees rooted at the children of the root of T (see Lemma 7). Moreover, such a partition can be computed in polynomial time using Lemmas 8–10. In contrast, when the input is not dense or when one considers dense $\mathcal{R}^{+-}\mathcal{F}^{+-}$ Consistency, not all of Z may appear in the input F^+.

5.1 The Fan Graph and the Clique Graph

Consider any $L' \subseteq L$. Define $R^+|L' = \{t \in R^+\ :\ \Lambda(t) \subseteq L'\}$, $F^+|L' = \{t \in F^+\ :\ \Lambda(t) \subseteq L'\}$, and $F^-|L' = \{t \in F^-\ :\ \Lambda(t) \subseteq L'\}$. The *fan graph* $\mathcal{G}_{L'}$ is the undirected graph (L', E'), where for any $x, y \in L'$, it holds that $\{x, y\} \in E'$ if and only if $x|y|z \in F^+|L'$ for some $z \in L'$.

If T is a tree that is consistent with the input then the degree of the root of T can be determined from $\mathcal{G}_L$ as follows:

Lemma 7. *Suppose $|L| \geq 3$ and that there exists a tree T that is consistent with the input. Let p be the degree of the root of T, let $C_1, C_2, \ldots, C_m$ be the connected components of $\mathcal{G}_L$, and let $\Lambda(C_i)$ for each $i \in \{1, 2, \ldots, m\}$ be the set of vertices in C_i. The following holds:*

1. *If $m \geq 2$ then $p = 2$. Furthermore, if S' is any binary tree with m leaves and for each $i \in \{1, 2, \ldots, m\}$, S_i is a tree with $\Lambda(S_i) = \Lambda(C_i)$ such that $(F^+|\Lambda(C_i)) \subseteq t(S_i)$ and $(F^-|\Lambda(C_i)) \cap t(S_i) = \emptyset$, then the tree S obtained by replacing the m leaves in S' by the trees in $\{S_i : 1 \leq i \leq m\}$ satisfies $F^+ \subseteq t(S)$ and $F^- \cap t(S) = \emptyset$.*
2. *If $m = 1$ then $p \geq 3$. Furthermore, the value of p and the partition of L into subsets $L_1, L_2, \ldots, L_p$ are unique, where each L_i is the leaf label set of a subtree rooted at a child of the root of T.*

Proof.

1. First we show that $p = 2$ by contradiction. Suppose $p \geq 3$ and let x, y, and z be any three leaves from three different subtrees rooted at the children of the root of T. Since the input is dense, at least one rooted triplet t with $\Lambda(t) = \{x, y, z\}$ is specified in R^+, F^+, or F^-; by the choice of x, y, z, it has to be $x|y|z$. But then the edges $\{x, y\}$, $\{x, z\}$, and $\{y, z\}$ are in $\mathcal{G}_L$ so x, y, and z belong to the same connected component C_i. By repeating the argument, every leaf in L belongs to C_i, which contradicts $m \geq 2$.
   Next, consider any two connected components C_i and C_j in $\mathcal{G}_L$. By the definition of $\mathcal{G}_L$, there is no fan triplet in F^+ with leaves belonging to both C_i and C_j. Hence, F^+ equals $\bigcup_{i=1}^{m}(F^+|\Lambda(C_i))$. By the definition of S, $\bigcup_{i=1}^{m}(F^+|\Lambda(C_i)) \subseteq t(S)$. Finally, since S is binary, $F^- \cap t(S) = F^- \cap (\bigcup_{i=1}^{m} t(S_i)) = \bigcup_{i=1}^{m}((F^-|\Lambda(C_i)) \cap t(S_i)) = \emptyset$.
2. To prove that $p \geq 3$, suppose on the contrary that $p = 2$. Let A and B be the two sets of leaves in the subtrees rooted at the two children of the root of T. Since $\mathcal{G}_L$ is connected, there exists some $a \in A$ and $b \in B$ such that $\{a, b\}$ is an edge of $\mathcal{G}_L$. By the definition of $\mathcal{G}_L$, there exists some $c \in L$ where $a|b|c \in F^+$. However, this is impossible since $p = 2$. This gives $p \geq 3$.
   Next, we prove the uniqueness of the partition of L by contradiction. Suppose that T_1 and T_2 are two trees with $F^+ \subseteq t(T_1)$, $F^+ \subseteq t(T_2)$, and $F^- \cap t(T_1) = F^- \cap t(T_2) = \emptyset$ and that the partitions of L induced by the children of the root of T_i are different for $i = 1$ and $i = 2$. For $i \in \{1, 2\}$, denote the root of T_i by r_i. We claim that there exist $x, y, z \in L$ such that for some $i \in \{1, 2\}$: (1) x, y appear in the same subtree rooted at a child of r_i and z in another such subtree; and (2) x, y, z appear in three different subtrees rooted at the children of r_{3-i}. To prove the claim, for some $i \in \{1, 2\}$, take any two leaves x and y in the same subtree D_i rooted at a child of r_i but in different subtrees D_{3-i}, D'_{3-i} rooted at a child of r_{3-i}. Without loss of generality, assume $i = 1$. If there exists a leaf z in another subtree D'_1 rooted at a child of r_1 and z belongs to a subtree D''_2 rooted at a child of r_2 different from D_2 and D'_2 then we are done. Otherwise, all leaves not in D_2 or D'_2 also appear in D_1 and we let a be any such leaf; moreover, all leaves not in D_1

appear in either D_2 or D_2' and we let b be any such leaf, and then define w as follows: (i) $w = x$ if b and y are in the same subtree rooted at a child of r_2, and (ii) $w = y$ if b and x are in the same subtree. The three leaves a, b, and w then satisfy the claim. Since the claim is true, $x|y|z \notin t(T_i)$ while $x|y|z \in t(T_{3-i})$. This means that if $x|y|z \in F^+$ then $F^+ \subseteq t(T_i)$ is false, if $x|y|z \in F^-$ then $F^- \cap t(T_{3-i})$ is false, and if one of $xy|z$, $xz|y$, and $yz|x$ is in R^+ then $R^+ \subseteq t(T_{3-i})$ is false, giving a contradiction in every case. □

The three lemmas below will be used by `DenseBuild` in Sect. 5.2 to construct the partition in Lemma 7.2. In the rest of this subsection, assume that $\mathcal{G}_L$ contains a single connected component and that there exists a tree T that is consistent with the input. For every $a, b \in L$, define $f(a,b) = \left|\{z : a|b|z \in F^+\}\right|$.

Lemma 8. *If $a, b \in L$ are any two leaves that maximize the value of $f(a,b)$, i.e., $f(a,b) = \max_{x,y \in L} f(x,y)$, then a and b belong to the smallest and second smallest subtrees T_a and T_b rooted at children of the root of T (with ties broken arbitrarily). Also, $f(a,b) = |\Lambda(T)| - |\Lambda(T_a)| - |\Lambda(T_b)|$.*

Proof. Consider any $x, y \in L$ and define $s = lca^T(x,y)$. For any $x|y|z \in F^+$, we have $lca^T(x,y) = lca^T(x,z) = lca^T(y,z) = s$, which means that $z \in \Lambda(T[s]) \setminus (\Lambda(T[s_x]) \cup \Lambda(T[s_y]))$, where $T[u]$ for any node u in T denotes the subtree of T rooted at u and s_x (resp., s_y) is the child of s that is an ancestor of x (resp., y). Thus, $f(x,y) = |\Lambda(T[s])| - |\Lambda(T[s_x])| - |\Lambda(T[s_y])|$.

Next, according to Lemma 7.2, since $\mathcal{G}_L$ consists of one connected component, T has at least three subtrees attached to the root. To maximize the value of $f(x,y)$, we therefore choose s to be the root of T and $T[s_x]$ and $T[s_y]$ to be the smallest and second smallest subtrees attached to s. The lemma follows. □

Lemma 9. *Let $a, b \in L$ be two leaves that maximize the value of $f(a,b)$. Define $L' = \{a,b\} \cup \{x \in L : a|b|x \notin F^+\}$ and take any $z \notin L'$. Then the leaf label sets of the two smallest subtrees attached to the root of T are $A = \{a\} \cup \{x \in L' : a|x|z \notin F^+\}$ and $B = L' \setminus A = \{b\} \cup \{x \in L' : b|x|z \notin F^+\}$.*

Proof. By Lemma 8, a and b appear in the two smallest subtrees T_a and T_b attached to the root of T. For every leaf label x in T_a or T_b, $a|b|x \notin F^+$. Thus, $L' = \Lambda(T_a) \cup \Lambda(T_b)$. Since $z \notin L'$, z is not in the subtrees containing a and b. On the other hand, for every leaf x in T_a with $x \neq a$, we have $x \in L'$ and therefore $a|x|z \notin F^+$. Hence, $\Lambda(T_a) = \{a\} \cup \{x \in L' : a|x|z \notin F^+\}$. In the same way, $\Lambda(T_b) = \{b\} \cup \{x \in L' : b|x|z \notin F^+\}$. □

Finally, suppose that a, b, and L' are defined as in Lemma 9. Let c be either one of a and b. The *clique graph* $\mathcal{Q}_L$ is the undirected graph (L'', E''), where $L'' = L \setminus L'$ and $\{x, y\} \in E''$ if and only if $c|x|y \notin F^+$. The clique graph has the following useful properties:

Lemma 10. *Let C be any connected component in $\mathcal{Q}_L$. Then C forms a complete graph. Also, the set of vertices in C equals the set of leaves in some subtree attached to the root of T.*

Proof. Let T_c be the subtree rooted at a child of the root of T that contains c. Consider any $x \in L \setminus L'$ and note that x cannot appear in T_c. For any $y \in L \setminus L'$, if $c|x|y \in F^+$ then x and y have to be in two different subtrees attached to the root of T. Conversely, for any $x, y \in L \setminus L'$ in the same subtree attached to the root of T, we have $c|x|y \notin F^+$. Therefore, the set of leaves in each subtree attached to the root of T induce a complete subgraph in $\mathcal{Q}_L$. □

5.2 Algorithm `DenseBuild`

We now develop an efficient algorithm for $\mathcal{R}^+\mathcal{F}^{+-}$ Consistency restricted to dense inputs. The algorithm is named `DenseBuild` and its pseudocode is summarized in Fig. 2. (Refer to Sect. 5.1 for the notation defined there.) The basic strategy is to use the information contained in R^+, F^+, and F^- to partition the leaf label set L into subsets corresponding to the leaf label sets of the subtrees rooted at the children of the root of the solution, and then construct each such subtree recursively. On a high level, this is similar to the BUILD algorithm of Aho *et al.* [1] which also uses top-down recursion, but `DenseBuild` has to do the leaf partitioning in a different way to take the fan triplets into account. Also, `DenseBuild` needs to distinguish between when the root has degree 2 and degree strictly larger than 2 (cf., Lemma 7).

As a preprocessing step, `DenseBuild` constructs the fan graph $\mathcal{G}_L$ and assigns a weight $w(x, y)$ to each edge $\{x, y\}$ in $\mathcal{G}_L$ equal to $\big|\{x|y|z \in F^+ : z \in L\}\big|$. In the preprocessing step, the algorithm also computes and stores the value $f(a, b)$ for every $a, b \in L$. The next lemma shows that when the algorithm calls itself recursively, it does not have to recompute any $f(a, b)$-values. For any $L' \subseteq L$ and $a, b \in L'$, define $f_{L'}(a, b) = \big|\{z : a|b|z \in F^+|L'\}\big|$.

Lemma 11. *Suppose that T is a tree with $\Lambda(T) = L$ and that T is consistent with the input. Let $L' \subseteq L$ be the set of leaves in a subtree rooted at any child of the root of T. Then $f_{L'}(a, b) = f_L(a, b) = f(a, b)$ for every $a, b \in L'$.*

Proof. Fix $a, b \in L'$. For any fan triplet of the form $a|b|z \in F^+$, z also has to belong to L', and therefore $a|b|z \in F^+|L'$. Conversely, $a|b|z \in F^+|L'$ implies $a|b|z \in F^+$ by definition. Hence, $\{z : a|b|z \in F^+\} = \{z : a|b|z \in F^+|L'\}$. □

After the preprocessing step is complete, `DenseBuild` proceeds as follows. It computes the connected components $C_1, C_2, \ldots, C_m$ of $\mathcal{G}_L$ in step 1. According to Lemma 7, there are two main cases: if $m \geq 2$ then the root of any tree consistent with the input must have degree two, but if $m = 1$ then the root must have degree at least three.

In the former case (steps 2.1–2.3), the algorithm recursively constructs a tree T_i for the leaves in C_i for each $i \in \{1, 2, \ldots, m\}$, thus handling the input rooted triplets over leaves within each connected component. To handle the rest, i.e., those whose leaves belong to more than one connected component in $\mathcal{G}_L$, the algorithm constructs an instance of non-dense $\mathcal{R}^+$ Consistency whose leaf label set represents the set of connected components in $\mathcal{G}_L$ and whose set of

Algorithm `DenseBuild`

Input: Three sets R^+, F^+, F^- of rooted triplets over a leaf label set L forming a dense instance of $\mathcal{R}^+\mathcal{F}^{+-}$ Consistency.
The algorithm assumes the following preprocessing: $\mathcal{G}_L$ has been constructed and edge-weighted, and $f(a,b)$ for all $a,b \in L$ have been precomputed.
When making recursive calls, the algorithm passes $L' \subseteq L$ and $\mathcal{G}_{L'}$ as parameters.

Output: A tree T with $\Lambda(T) = L$ such that $R^+ \cup F^+ \subseteq t(T)$ and $F^- \cap t(T) = \emptyset$, if such a tree exists; otherwise, *null*.

1 Let $C_1, C_2, \ldots, C_m$ be the connected components of $\mathcal{G}_L$;
2 **if** $(m > 1)$ **then**
2.1 For $i \in \{1, 2, \ldots, m\}$, extract $\mathcal{G}_{L_i}$ from $\mathcal{G}_L$ and compute $T_i =$ `DenseBuild`$(L_i, \mathcal{G}_{L_i})$, where L_i is the set of leaf labels in C_i;
2.2 Let $R' = \{C_iC_j|C_k : \exists xy|z \in R^+$ with $x \in C_i, y \in C_j, z \in C_k\}$ and let T' be the output of the BUILD algorithm on input R';
2.3 **if** $T' = null$ or $T_i = null$ for any $i \in \{1, 2, \ldots, m\}$ then **return** *null*;
else let T be the tree obtained by arbitrarily refining T' to a binary tree and replacing each leaf C_i in T' by the tree T_i, and **return** T;
else
3 /* $(m = 1)$ */
3.1 Find $a, b \in L$ that maximize $f(a,b)$;
3.2 Let $L' = \{a,b\} \cup \{x : a|b|x \notin F^+\}$, $z \notin L'$, $L_1 = \{a\} \cup \{x \in L' : a|x|z \notin F^+\}$, and $L_2 = L' \setminus L_1$;
3.3 Build the clique graph $\mathcal{Q}_L$ and let $L_3, \ldots, L_p$ be the leaf labels in the different connected components in $\mathcal{Q}_L$;
3.4 **if** $(\{x|y|z : x \in L_i, y \in L_j, z \in L_k,$ where i,j,k are different$\} \subseteq F^+)$ and $\{xy|z \in R^+ : x \in L_i, y \in L_j, z \in L_k,$ where i,j,k are different$\} = \emptyset$ **then**
3.4.1 Decrement $w(x,y)$, $w(x,z)$, and $w(y,z)$ by one for every $x|y|z \in F^+$ such that $x \in L_i, y \in L_j, z \in L_k$, and i,j,k are different;
3.4.2 For $i \in \{1, 2, \ldots, p\}$, extract $\mathcal{G}_{L_i}$ from $\mathcal{G}_L$ and compute $T_i =$ `DenseBuild`$(L_i, \mathcal{G}_{L_i})$;
3.4.3 **if** $T_i = null$ for any $i \in \{1, 2, \ldots, p\}$ then **return** *null*;
else create a tree T by attaching the root of T_i for every $i \in \{1, 2, \ldots, p\}$ to a common root node and **return** T;
else
3.4.4 **return** *null*;
endif
endif

End `DenseBuild`

Fig. 2. Algorithm `DenseBuild`.

resolved triplets is $\{C_iC_j|C_k : \exists xy|z \in R^+$ with $x \in C_i, y \in C_j, z \in C_k\}$. It then applies the BUILD algorithm from [1] to obtain a tree T' (if one exists) consistent with all resolved triplets in R^+ involving leaves from more than one connected component. (If no such T' exists or if some T_i-tree is *null*, `DenseBuild` will return *null* and give up.) Then, `DenseBuild` arbitrarily refines T' into a binary tree as in the proof of Lemma 2 above. Finally, the output tree T is obtained by replacing each C_i-leaf in T' by the corresponding T_i-tree. By Lemma 7.1, T is consistent with all fan triplets in F^+ and no fan triplets in F^-.

In the latter case (steps 3.1–3.4.4), Lemma 7.2 ensures that the partition of L into leaf label sets of the subtrees rooted at the children of the root is uniquely defined. This partition is recovered in steps 3.1–3.3 in accordance with Lemmas 8–10. Next, step 3.4 verifies that the resulting partition $L_1, L_2, \ldots, L_p$ is valid by checking if $x|y|z \in F^+$ and $xy|z \notin R^+$ hold for every $x \in L_i$, $y \in L_j$, $z \in L_k$ where i, j, k are different. If the partition is valid then, for each $i \in \{1, 2, \ldots, p\}$, the algorithm first constructs $\mathcal{G}_{L_i}$ (to avoid building $\mathcal{G}_{L_i}$ from scratch, the weight $w(x, y)$ of each edge $\{x, y\}$ in $\mathcal{G}_{L_i}$ is updated by subtracting 1 for every fan triplet $x|y|z \in F^+$ that contributed to $w(x, y)$ in $\mathcal{G}_L$ but no longer exists on subsequent recursion levels; any edge whose weight reaches 0 is removed). Then, it recursively builds a tree T_i with $\Lambda(T_i) = L_i$. The output tree T is formed by attaching the roots of all the T_i-trees to a common root node.

Theorem 4. *Algorithm* `DenseBuild` *solves the dense version of the* $\mathcal{R}^+\mathcal{F}^{+-}$ Consistency *problem in* $O(n^3)$ *time.*

Proof. The preprocessing step constructs $\mathcal{G}_L$, assigns weights to the edges in $\mathcal{G}_L$, and computes all values of $f(a, b)$ where $a, b \in L$, which takes $T_A(n) = O(n^3)$ time in total. We now bound the time needed to execute `DenseBuild`$(L, \mathcal{G}_L)$ assuming that the preprocessing has been taken care of.

Let $T_B(n)$ be the total time used by the calls to BUILD in step 2.2 on all recursion levels, and let $T_C(n)$ be the total time for all other computations. To analyze $T_B(n)$, let $n_1, n_2, \ldots, n_k$ be the cardinalities of the leaf label sets of the constructed sets R' of resolved triplets in the successive calls to BUILD in step 2.2. By applying Henzinger *et al.* fast implementation of BUILD (Algorithm B' in [14]), we get $T_B(n) = \sum_{i=1}^{k} O(n_i^3 + n_i^2 \log n_i) = O(\sum_{i=1}^{k} n_i^3)$. Also, $n_1 + n_2 + \cdots + n_k = O(n)$ because every leaf in each such constructed instance of $\mathcal{R}^+$ Consistency corresponds to either an internal node or a leaf in the tree output by `DenseBuild`, which has $O(n)$ nodes. Thus, $T_B(n) = O(n^3)$. Next, we derive an upper bound on $T_C(n)$. For any partition of L into $L_1, L_2, \ldots, L_m$, let $c(L_1, L_2, \ldots, L_m)$ denote the number of possible fan triplets of the form $x|y|z$ such that $x \in L_i, y \in L_j, z \in L_k$ and i, j, k are different. Observe that $c(L_1, L_2, \ldots, L_m) = O\big(\binom{|L|}{3} - \sum_{i=1}^{m} \binom{|L_i|}{3}\big)$. Then $T_C(n)$ consists of the time needed to find the m connected components in $\mathcal{G}_L$, which is $O(|L|^2)$, plus the time to:

– If $m \geq 2$:
  (a) build $\mathcal{G}_{L_i}$ for all $i \in \{1, 2, \ldots, m\}$ ($O(c(L_1, L_2, \ldots, L_m))$ time);
  (b) construct R' (also $O(c(L_1, L_2, \ldots, L_m))$ time); and
  (c) handle the recursive calls ($\sum_{i=1}^{m} T_C(|L_i|)$ time).
– If $m = 1$:
  (a) find the partition of L into $L_1, L_2, \ldots, L_p$ in steps 3.1–3.3 ($O(|L|^2)$ time);
  (b) verify that the partition is valid in step 3.4 ($O(c(L_1, L_2, \ldots, L_p))$ time); and
  (c) handle the recursive calls ($\sum_{i=1}^{p} T_C(|L_i|)$ time).

Define $q = \max\{m, p\}$. In total, $T_C(n) = O(|L|^2) + O(c(L_1, L_2, \ldots, L_q)) + \sum_{i=1}^{q} T_C(|L_i|)$, which gives $T_C(n) = O(n^3)$ by induction.

Finally, $T_A(n) + T_B(n) + T_C(n) = O(n^3)$. □

6 Concluding Remarks

The newly derived results (see Tables 1, 2, 3 for a summary) highlight the following open problems:

- What is the computational complexity of the D-bounded degree $\mathcal{F}^+$ Consistency problem when $D = 3$? I.e., is the following problem solvable in polynomial time: Given a set F^+ of fan triplets, does there exist a degree-3 tree consistent with all of F^+?
- For the special case of $D = 3$, do the following problems have the same computational complexity or not: D-bounded degree $\mathcal{F}^+$ Consistency, D-bounded degree $\mathcal{F}^{+-}$ Consistency, D-bounded degree $\mathcal{R}^+\mathcal{F}^+$ Consistency, and D-bounded degree $\mathcal{R}^+\mathcal{F}^{+-}$ Consistency?
- How does the complexity of $\mathcal{R}^{+-}\mathcal{F}^{+-}$ Consistency and its problem variants change when other parameters such as the *height* of the output tree are restricted or if one requires the output tree to be *ordered* in such a way that its left-to-right sequence of leaves must equal a prespecified sequence? Note that the analogue of $\mathcal{R}^+$ Consistency in the *unrooted* setting where the input is a set of "quartets" (unrooted, distinctly leaf-labeled trees with four leaves where every internal node has three neighbors) is already NP-hard [22].
- Can fixed-parameter tractable algorithms be developed for any of the NP-hard variants of $\mathcal{R}^{+-}\mathcal{F}^{+-}$ Consistency?

One may also consider a minimization version of the D-bounded degree $\mathcal{F}^+$ Consistency problem, in which the input is a set F^+ of fan triplets and the objective is to construct a tree with as small degree as possible that is consistent with all fan triplets in F^+. However, this is a difficult problem since Lemma 5 and the polynomial-time inapproximability result for the minimization version of K-Coloring by Zuckerman [25, Theorem 1.2] imply that the problem cannot be approximated within a ratio of $n^{1-\epsilon}$ for any constant $\epsilon > 0$ in polynomial time, unless P = NP.

Acknowledgments. The authors would like to thank Sylvain Guillemot and Avraham Melkman for some discussions related to the topic of this paper. J.J. was partially funded by The Hakubi Project at Kyoto University and KAKENHI grant number 26330014.

References

1. Aho, A.V., Sagiv, Y., Szymanski, T.G., Ullman, J.D.: Inferring a tree from lowest common ancestors with an application to the optimization of relational expressions. SIAM J. Comput. **10**(3), 405–421 (1981)
2. Bininda-Emonds, O.R.P.: The evolution of supertrees. TRENDS Ecol. Evol. **19**(6), 315–322 (2004)
3. Bininda-Emonds, O.R.P., Cardillo, M., Jones, K.E., MacPhee, R.D.E., Beck, R.M.D., Grenyer, R., Price, S.A., Vos, R.A., Gittleman, J.L., Purvis, A.: The delayed rise of present-day mammals. Nature **446**(7135), 507–512 (2007)
4. Bryant, D.: Building trees, hunting for trees, and comparing trees: theory and methods in phylogenetic analysis. Ph.D. thesis, University of Canterbury, Christchurch, New Zealand (1997)
5. Byrka, J., Gawrychowski, P., Huber, K.T., Kelk, S.: Worst-case optimal approximation algorithms for maximizing triplet consistency within phylogenetic networks. J. Discrete Algorithms **8**(1), 65–75 (2010)
6. Byrka, J., Guillemot, S., Jansson, J.: New results on optimizing rooted triplets consistency. Discrete Appl. Math. **158**(11), 1136–1147 (2010)
7. Chor, B., Hendy, M., Penny, D.: Analytic solutions for three taxon ML trees with variable rates across sites. Discrete Appl. Math. **155**(6–7), 750–758 (2007)
8. Constantinescu, M., Sankoff, D.: An efficient algorithm for supertrees. J. Classif. **12**(1), 101–112 (1995)
9. Jansson, J., Lingas, A., Lundell, E.-M.: The approximability of maximum rooted triplets consistency with fan triplets and forbidden triplets. In: Cicalese, F., Porat, E., Vaccaro, U. (eds.) CPM 2015. LNCS, vol. 9133, pp. 272–283. Springer, Cham (2015). doi:10.1007/978-3-319-19929-0_23
10. Felsenstein, J.: Inferring Phylogenies. Sinauer Associates, Inc., Sunderland (2004)
11. Garey, M., Johnson, D.: Computers and Intractability - A Guide to the Theory of NP-Completeness. W. H. Freeman and Company, New York (1979)
12. Gąsieniec, L., Jansson, J., Lingas, A., Östlin, A.: On the complexity of constructing evolutionary trees. J. Comb. Optim. **3**(2–3), 183–197 (1999)
13. He, Y.J., Huynh, T.N.D., Jansson, J., Sung, W.-K.: Inferring phylogenetic relationships avoiding forbidden rooted triplets. J. Bioinform. Comput. Biol. **4**(1), 59–74 (2006)
14. Henzinger, M.R., King, V., Warnow, T.: Constructing a tree from homeomorphic subtrees, with applications to computational evolutionary biology. Algorithmica **24**(1), 1–13 (1999)
15. Holm, J., de Lichtenberg, K., Thorup, M.: Poly-logarithmic deterministic fully-dynamic algorithms for connectivity, minimum spanning tree, 2-edge, and biconnectivity. J. ACM **48**(4), 723–760 (2001)
16. Jansson, J., Lemence, R.S., Lingas, A.: The complexity of inferring a minimally resolved phylogenetic supertree. SIAM J. Comput. **41**(1), 272–291 (2012)
17. Jansson, J., Ng, J.H.-K., Sadakane, K., Sung, W.-K.: Rooted maximum agreement supertrees. Algorithmica **43**(4), 293–307 (2005)

18. Ng, M.P., Wormald, N.C.: Reconstruction of rooted trees from subtrees. Discrete Appl. Math. **69**(1–2), 19–31 (1996)
19. Semple, C.: Reconstructing minimal rooted trees. Discrete Appl. Math. **127**(3), 489–503 (2003)
20. Semple, C., Daniel, P., Hordijk, W., Page, R.D.M., Steel, M.: Supertree algorithms for ancestral divergence dates and nested taxa. Bioinformatics **20**(15), 2355–2360 (2004)
21. Snir, S., Rao, S.: Using max cut to enhance rooted trees consistency. IEEE/ACM Trans. Comput. Biol. Bioinform. **3**(4), 323–333 (2006)
22. Steel, M.: The complexity of reconstructing trees from qualitative characters and subtrees. J. Classif. **9**(1), 91–116 (1992)
23. Sung, W.: Algorithms in Bioinformatics: A Practical Introduction. Chapman & Hall/CRC, Boca Raton (2010)
24. Willson, S.J.: Constructing rooted supertrees using distances. Bull. Math. Biol. **66**(6), 1755–1783 (2004)
25. Zuckerman, D.: Linear degree extractors and the inapproximability of Max Clique and Chromatic Number. Theory Comput. **3**(1), 103–128 (2007)