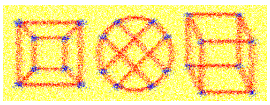
Journal of Graph Algorithms and Applications
http://jgaa.info/ vol. 30, no. 1, pp. 25–46 (2026)
DOI: 10.7155/jgaa.v30i1.2962

Relaxed Agreement Forests of Phylogenetic Trees

Virginia Ardévol Martínez[1] *Steven Chaplick*[2] *Steven Kelk*[2] *Ruben Meuwese*[2] *Matúš Mihalák*[2] *Georgios Stamoulis*[2]

[1]Université Paris-Dauphine, PSL University, CNRS, LAMSADE, France
[2]Department of Advanced Computing Sciences, Maastricht University, the Netherlands

Submitted: September 2024 Accepted: January 2026 Published: March 2026

Article type: Regular Paper Communicated by: Fabio Vadin

Abstract. The phylogenetic inference process can produce, for multiple reasons, conflicting hypotheses of the evolutionary history of a set X of biological entities, i.e., phylogenetic trees with the same set of leaf labels X but with distinct topologies. It is natural to wish to quantify the difference between two such trees T_1 and T_2. We introduce the problem of computing a *maximum relaxed agreement forest* (MRAF) and use this as a proxy for the dissimilarity of T_1 and T_2, which in this article we assume to be unrooted and binary. MRAF asks for a partition of the leaf labels X into a minimum number of blocks $S_1, \ldots, S_k$ such that the two subtrees induced in T_1 and T_2 by every S_i are isomorphic up to suppression of degree-2 nodes and taking the labels X into account. Unlike the earlier introduced maximum agreement forest (MAF) model, the subtrees induced by the S_i are allowed to overlap. We prove that it is NP-hard to compute MRAF, by reducing from the problem of partitioning a permutation into a minimum number of monotonic subsequences (PIMS). We further show that MRAF has a $O(\log n)$-approximation algorithm where $n = |X|$ and permits exact algorithms with single-exponential running time. When one of the trees is a caterpillar, we prove that testing whether an MRAF has size at most k can be answered in polynomial time when k is fixed. We also note that on two caterpillars the approximability of MRAF is related to that of PIMS. Finally, we establish a number of bounds on MRAF, compare its behaviour to MAF both theoretically and experimentally and discuss a number of open problems.

1 Introduction

The central challenge of phylogenetics, which is the study of phylogenetic (evolutionary) trees, is to infer a tree whose leaves are bijectively labeled by a set of species X and which accurately

Ruben Meuwese was supported by NWO grant *Deep kernelization for phylogenetic discordance* OCENW.KLEIN.305. Preliminary version of this paper appeared at the 49th International Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM 2024).

E-mail addresses: steven.kelk@maastrichtuniversity.nl (Steven Kelk)

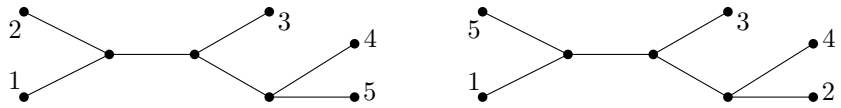

Figure 1: The two trees, while isomorphic, are not isomorphic when taking the leaf-labeling into account, and thus both MRAF and MAF cannot be of size one. An MRAF has 2 blocks, e.g., $\{1, 2, 3\}$ and $\{4, 5\}$. A MAF has 3 blocks, e.g., $\{1, 2, 3\}$, $\{4\}$, and $\{5\}$.

represents the evolutionary events that gave rise to X [28]. There are many existing techniques to infer phylogenetic trees from biological data and under a range of different objective functions [22]. The complexity of this problem arises from the fact that we typically only have indirect data available, such as DNA sequences of the species X. Different techniques regularly yield trees with differing topologies, or the same technique constructs different trees depending on which part of a genome the DNA data is extracted from [24]. Hence, it is insightful to formally quantify the dissimilarity between (pairs of) phylogenetic trees, stimulating research into various distance measures.

Here we propose a new dissimilarity measure between unrooted phylogenetic trees T_1, T_2 which is conceptually related to the well-studied *agreement forest* abstraction. An agreement forest (AF) is a partition of X into blocks which induce, in the two input trees, non-overlapping isomorphic subtrees, modulo edge subdivision and taking the labels X into account; computing such a forest of minimum size (a MAF) is NP-hard [16] although it can be computed reasonably well in practice [30]. The AF abstraction originally derives its significance from the fact that, in unrooted (respectively, rooted) phylogenetic trees, an AF of minimum size models *Tree Bisection and Reconnection* (TBR) (respectively, *rooted Subtree Prune and Regraft*, rSPR) distance [1, 5]. For background on AFs we refer to recent articles such as [8, 6]. Here we propose the *relaxed agreement forest* abstraction (RAF). The only difference is that we no longer require the partition of X to induce non-overlapping subtrees; they only have to be isomorphic (see Fig. 1). We write MRAF to denote a relaxed agreement forest of minimum size. As we will observe, in the worst case MRAF can be constant while MAF grows linearly in $|X|$.

The fact that RAFs are allowed to induce overlapping subtrees is potentially interesting from the perspective of biological modelling. Unlike an AF, multiple subtrees of the RAF can pass through a single branch of T_1 (or T_2). This allows us to view T_1 and T_2 as the union of several interleaved, overlapping, common evolutionary histories. It is beyond the scope of this article to expound upon this in detail, but it is compatible with the trend in the literature of phylogenetic trees (or networks) having multiple distinct evolutionary histories woven within them which sometimes evolve "in parallel" due to phenomena such as incomplete lineage sorting [9, 24, 17]. Indeed, one possible application of MRAF is as a model for solving the following inverse problem: given two phylogenetic trees on X that have been inferred by merging a common set of evolutionary building blocks but in different ways, what is the most parsimonious way to untangle these building blocks? MAF can also be viewed in this inverse sense, but insisting that the common evolutionary building blocks are disjoint in T_1 and T_2 might be overly restrictive. This greater modelling flexibility, rather than computational tractability issues, is our primary reason for studying MRAF.

Our results are as follows. First, we show that it is NP-hard to compute an MRAF. We reduce from the problem of partitioning a permutation into a minimum number of monotone subsequences (PIMS). We show that MRAF has a $O(\log n)$-approximation algorithm where $n = |X|$ and permits exact algorithms with single-exponential running time. When one of the two trees is a caterpillar,

we prove that "Is there a RAF with at most k components?" can be answered in polynomial time when k is fixed, i.e., the problem is in XP parameterized by k. This XP algorithm relies on solving many instances of a constrained version of the MRAF problem and we also show that this constrained version is NP-hard already on caterpillars. We also relate the approximability of MRAF to that of PIMS. Along the way we establish a number of bounds on MRAF, compare its behaviour to MAF and undertake an empirical analysis on two existing datasets.

2 Preliminaries, Basic Properties and Bounds

Let X be a set of labels (*taxa*) representing species. An *unrooted binary phylogenetic tree* T on X is a simple, connected, and undirected tree whose leaves are bijectively labeled with X and whose other vertices all have degree 3. When it is clear from the context we will simply write (*phylogenetic*) *tree* for shorthand. For two trees T and T' both on the same set of taxa X, we write $T = T'$ if there is an isomorphism between T and T' that preserves the labels X. Tree T is a *caterpillar* if deleting the leaves of T yields a path. We say that two distinct taxa $\{a, b\} \subseteq X$ form a *cherry* of a tree T if they have a common parent. The *identity caterpillar* on n leaves is simply the caterpillar with leaves $1, \dots n$ in ascending order with the exception of the two cherries $\{1, 2\}$ and $\{n-1, n\}$ at its ends; see e.g. the tree on the left in Fig. 1. Note that caterpillars are almost total orders, but not quite: the leaves in the cherries at the ends are incomparable. Managing this subtle difference is a key aspect of our results.

A *quartet* is an unrooted binary phylogenetic tree with exactly four leaves. Let T be a phylogenetic tree on X. If $\{a, b, c, d\} \subseteq X$ are four distinct leaves, we say that quartet $ab|cd$ *is induced by (or simply 'is a quartet of')* T if in T the path from a to b does not intersect the path from c to d. Note that, for any four distinct leaves $a, b, c, d \in X$, exactly one of the three quartets $ab|cd, ac|bd, ad|bc$ will be a quartet of T. It is well-known that $T_1 = T_2$ if and only if both trees induce exactly the same set of quartet topologies [27, 28][1]. For example, in Fig. 1 $12|45$ is a quartet of the first tree but not a quartet of the second tree. For $X' \subseteq X$, we write $T[X']$ to denote the unique, minimal subtree of T that connects all elements in the subset X'. We use $T|X'$ to denote the phylogenetic tree on X' obtained from $T[X']$ by suppressing degree-2 vertices. If $T_1|X' = T_2|X'$ then we say that the subtrees of T_1, T_2 induced by X' are *homeomorphic*.

Let T_1 and T_2 be two phylogenetic trees on X. Let $\mathcal{F} = \{S_1, \dots, S_k\}$ be a partition of X, where each block S_i, is referred to as a *component* of $\mathcal{F}$. We say that $\mathcal{F}$ is an *agreement forest* (AF) for T_1 and T_2 if these conditions hold:

1. For each $i \in \{1, 2, ..., k\}$ we have $T_1|S_i = T_2|S_i$.
2. For each pair $i, j \in \{1, 2, ..., k\}$ with $i \neq j$, we have that $T_1[S_i]$ and $T_1[S_j]$ are vertex-disjoint in T, and $T_2[S_i]$ and $T_2[S_j]$ are vertex-disjoint in T_2.

The *size* of $\mathcal{F}$ is simply its number of components, i.e., k. Moreover, an AF with the minimum number of components (over all AFs for T_1 and T_2) is called a *maximum agreement forest* (MAF) for T_1 and T_2. For ease of reading, we will also write MAF to denote the size of a MAF. This is NP-hard to compute [16, 1].

A *relaxed agreement forest* (RAF) is defined similarly to an AF, except without condition 2. A RAF with a minimum number of components is a *maximum relaxed agreement forest* (MRAF). We also use MRAF for the size of an MRAF; by definition, MRAF $\leq$ MAF.

[1] This is a consequence of a classical theorem in phylogenetics known as the Splits Equivalence Theorem [7].

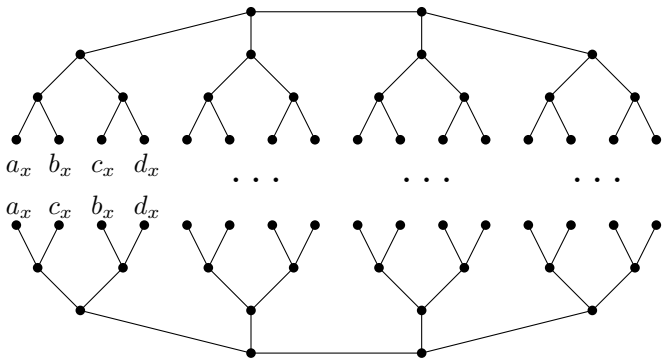

Figure 2: The construction described in the proof of Observation 2, for $n = 4$. Top is T_1, bottom is T_2. MRAF is always 2, but MAF grows linearly in n.

Maximum Relaxed Agreement Forest (MRAF)	
Input:	Two unrooted binary phylogenetic trees T_1, T_2 on the same leaf set X, and a number k.
Task:	Partition X into at most k sets $S_1, \ldots, S_k$ where $T_1\|S_i = T_2\|S_i$ for each i.

Observation 1 follows directly from the definitions and the aforementioned characterization of a phylogenetic tree in terms of its quartets. Observation 2 shows that MAF and MRAF can behave very differently.

Observation 1. *(a) A RAF with at most $\lceil \frac{n}{3} \rceil$ components always exists, where $n = |X|$, because if $|X'| = 3$ and $X' \subseteq X$ we have $T_1|X' = T_2|X'$ irrespective of X' or the topology of T_1 and T_2. (b) MRAF is 1 if and only if $T_1 = T_2$. (c) A partition $\{S_1, \ldots, S_k\}$ of X is a RAF of T_1, T_2 if and only if, for each S_i, the set of quartets induced by $T_1|S_i$ is identical to the set of quartets induced by $T_2|S_i$.*

Observation 2. *There are instances where MAF is arbitrarily large, $\Omega(n)$, while MRAF is constant.*

Proof: Let T be an arbitrary unrooted phylogenetic binary tree on n taxa. We create two trees T_1 and T_2, both on $4n$ taxa. We build T_1 by replacing each leaf x in T with a subtree on $\{a_x, b_x, c_x, d_x\}$ in which a_x, b_x form a cherry and c_x, d_x form a cherry. The construction of T_2 is similar except that a_x, c_x form a cherry and b_x, d_x form a cherry. See Fig. 2. Note that $T_1|\{a_x, b_x, c_x, d_x\} \neq T_2|\{a_x, b_x, c_x, d_x\}$. MRAF here is 2 because we can take one component containing all the a_x, b_x taxa and one containing all the c_x, d_x taxa. However, MAF is at least n. This is because in any AF at least one of the four taxa in $\{a_x, b_x, c_x, d_x\}$ must be a singleton component, and there are n subsets of the form $\{a_x, b_x, c_x, d_x\}$. □

Given two trees T_1, T_2 on X we say that $X' \subseteq X$ induces a *maximum agreement subtree* (MAST) if $T_1|X' = T_2|X'$ and X' has maximum cardinality ranging over all such subsets. Clearly, $\lceil \frac{n}{\text{MAST}} \rceil$ is a lower bound on MRAF, since each component of a RAF is no larger than a MAST. A MAST can be computed in polynomial time [29]. The trivial upper bound on MRAF of $\lceil \frac{n}{3} \rceil$ (see Observation 1), which already contrasts sharply with the fact that the MAF of two trees can be as large as $n(1 - o(1))$ [2], can easily be strengthened via MASTs. For example, it can be verified

computationally or analytically that for any two trees on 6 or more taxa, a MAST has size at least 4. We can thus repeatedly choose and remove a homeomorphic size-4 subtree, until there are fewer than 6 taxa left, giving a loose upper bound on MRAF of $n/4+2$. In fact, it is known that the size of a MAST on two trees with n leaves is $\Omega(\log n)$ [23] (and that this bound is asymptotically tight). In particular, the lower bound on MAST grows to infinity as n grows to infinity. Hence, the upper bound of $n/4+2$ can be strengthened to $n/c+f(c)$ for any arbitrary constant $c>1$ where f is a function that only depends on c; this is thus $n/c+O(1)$. In fact, by iteratively removing $\Omega(\log n')$ of the *remaining* number of taxa n' we obtain a (slightly) sublinear upper bound on the size of an MRAF. Namely, while $n' \geq \log n + O(1)$, each iteration removes at least $d \log n' \geq d \log\log n$ leaves for some constant d, giving an upper bound of $\frac{n}{d\log\log n} + \log n + O(1)$ which is $O(\frac{n}{\log\log n})$. Regarding lower bounds, one can generate pairs of trees on n leaves where a MAST has $O(\log n)$ leaves [20, 23]. An MRAF will thus have size $\Omega(\frac{n}{\log n})$.

Finally, it is well-known that the quantity MAF-1 is a metric distance [1], a property also enjoyed by several other dissimilarity measures on unrooted trees [21]. However, MRAF -1 is not a metric. In particular, it violates the triangle inequality. The trees in Fig. 3 are an example of this: $[\text{MRAF}(T_1,T_2)-1] > [\text{MRAF}(T_1,T_3)-1] + [\text{MRAF}(T_3,T_2)-1\ ]$.

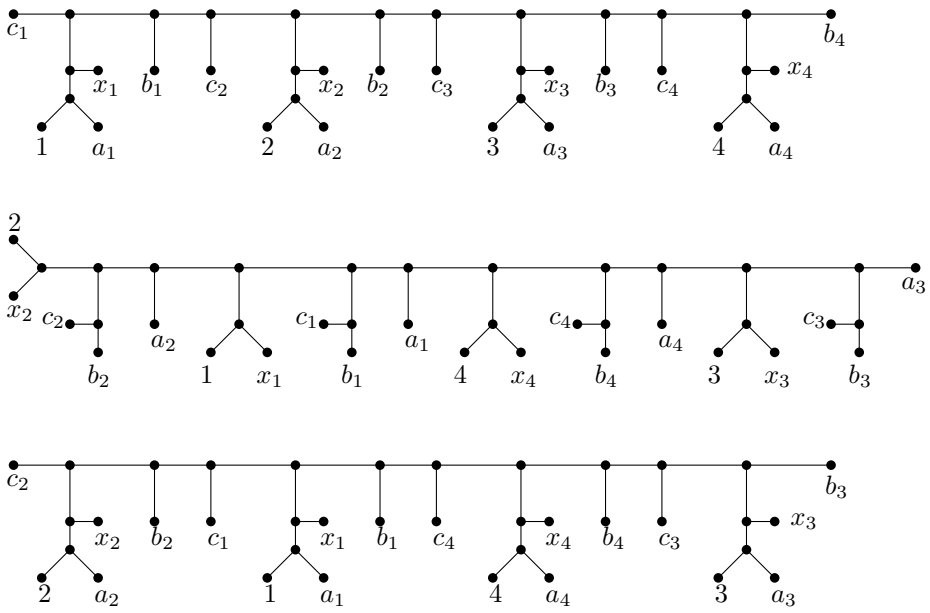

Figure 3: From top to bottom, T_1, T_2 and T_3. For T_1, T_2 (MRAF-1) is 3. For T_1, T_3 (MRAF-1) is 1. For T_3, T_2, (MRAF-1) is 1.These values can be verified by hand or using the implementation described in Section 6; the tree files are also on our GitHub page. Hence, (MRAF-1) does not obey the triangle inequality.

3 Hardness of MRAF

We discuss a related NP-hard problem regarding partitioning permutations [31].

Partition into Monotone Subsequences (PIMS)	
Input:	A permutation π of $\{1, \ldots, n\}$, and a number k.
Task:	Partition $\{1, \ldots, n\}$ into at most k sets such that each set occurs monotonically in π, i.e., either as an increasing or a decreasing sequence.

Due to the classical Erdős Szekeres Theorem [11], for any n-element permutation there is a monotone sequence in π with at least $\sqrt{n}$ elements. This can be used to efficiently partition π into at most $2\sqrt{n}$ monotone sequences [3]. Thus, we may assume that the k in the problem statement is at most $2\sqrt{n}$.

Theorem 1. *MRAF is NP-hard.*

Proof: Let (π, k) be an input to the PIMS problem, i.e., k is an integer greater than 1 and π is a permutation of $\{1, \ldots, n\}$, where we use π_i to denote the ith element of π. As remarked before, k is at most $2\sqrt{n}$. This will imply that our constructed instance of MRAF will have linear size in terms of the given permutation π, and as such any lower bounds, e.g., arising from the Exponential Time Hypothesis (ETH), will carry over from the PIMS problem to the MRAF problem. For each pair of integers (α, β) where $\alpha + \beta = k$ and $\alpha, \beta \geq 1$[2], we will construct an instance (T_1, T_2) of MRAF such that (T_1, T_2) has a solution consisting of k trees if and only if π can be partitioned into α increasing sequences and β decreasing sequences. The trees T_1 and T_2 are described as follows.

Recall that a *caterpillar* is a tree T where the subtree obtained by removing all leaves of T is a path. The path here is called the *spine* of the caterpillar. Note that, in the caterpillars used to construct T_1 and T_2, some spine vertices will have degree 2. However, to make proper binary trees one should contract any such vertex into one of its neighbors.

We first construct a leaf set $v_1, \ldots, v_n$ corresponding to the permutation. We create an identity caterpillar I whose spine is the n-vertex path $(x_1, \ldots, x_n)$ such that x_i is adjacent to v_i. Next, we create a caterpillar P whose spine is the n-vertex path $(y_1, \ldots, y_n)$ such that y_i is adjacent to v_{π_i}. Observe that already for the MRAF instance (I, P), any partition of π into r increasing subsequences and s decreasing subsequences, where $r+s = k$, leads to a solution to (I, P) consisting of k trees. However, the converse is not yet enforced. In particular, if the input to MRAF is (I, P), then the components in an MRAF (which are caterpillars) have cherries at their ends which, crucially, might be ordered differently in I than in P. This can violate monotonicity. To counter this, we extend I and P to obtain T_1, T_2 as shown in Fig. 4. The high-level idea is as follows: if a RAF component contains a subset of $v_1, \ldots, v_n$ that encodes an increasing (decreasing) subsequence, the component can also contain 2 taxa from each of the $4k$ dark-coloured (light-coloured) caterpillars (as shown in Fig. 4), so $8k$ in total from these caterpillars. We will show in due course that this is the only way for a RAF component that intersects with $v_1, \ldots, v_n$ to contain an additional $8k$ leaves from the caterpillars, and that components of this size are necessary to cover all the taxa of T_1, T_2 with k components. (Note that the fact that each such component contains taxa from caterpillars both to the 'left' and 'right' of I and P, implicitly imposes ordering on any cherries in the component from $v_1, \ldots, v_n$). It will then follow that a RAF with at most k components only exists if it is possible to partition π into (at most) α increasing subsequences and (at most) β decreasing subsequences.

We now describe the construction more formally. For T_1, we construct $8k$ caterpillars. First, for the increasing sequences, we construct $4k$ caterpillars $L_1, \ldots, L_{2k}$, $R_1, \ldots, R_{2k}$ each having 2α leaves and 2α spine vertices. Namely, for each i,

[2] $\alpha = 0$ or $\beta = 0$ makes the problem easy.

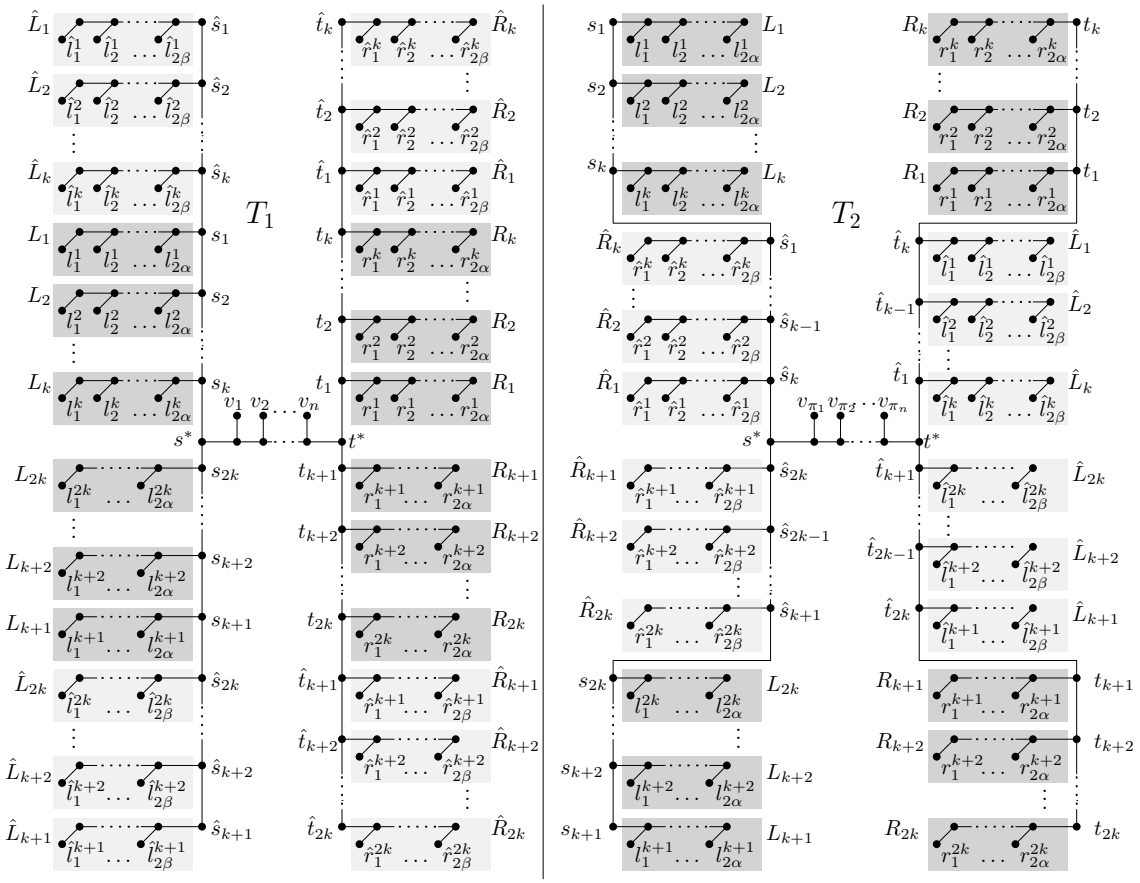

Figure 4: The two trees T_1, T_2 constructed from an instance of PIMS in the NP-hardness proof. The dark (light) grey leaves are used to induce increasing (decreasing) subsequences in the permutation-encoding taxa in the centre of the trees.

- L_i is the caterpillar with leaf set $\{l_1^i, \ldots, l_{2\alpha}^i\}$ and spine $(w_1^i, \ldots, w_{2\alpha}^i)$ where, for each j, l_j^i is adjacent to w_j^i; and
- R_i is the caterpillar with leaf set $\{r_1^i, \ldots, r_{2\alpha}^i\}$ and spine $(z_1^i, \ldots, z_{2\alpha}^i)$ where, for each j, r_j^i is adjacent to z_j^i.

For the decreasing sequences, we similarly construct $4k$ caterpillars $\hat{L}_1, \ldots, \hat{L}_{2k}$, $\hat{R}_1, \ldots, \hat{R}_{2k}$ each having 2β leaves and 2β spine vertices. Namely, for each i,

- $\hat{L}_i$ is the caterpillar with leaf set $\{\hat{l}_1^i, \ldots, \hat{l}_{2\beta}^i\}$ and spine $(\hat{w}_1^i, \ldots, \hat{w}_{2\beta}^i)$ where, for each j, $\hat{l}_j^i$ is adjacent to $\hat{w}_j^i$; and
- $\hat{R}_i$ is the caterpillar with leaf set $\{\hat{r}_1^i, \ldots, \hat{r}_{2\beta}^i\}$ and spine $(\hat{z}_1^i, \ldots, \hat{z}_{2\beta}^i)$ where, for each j, $\hat{r}_j^i$ is adjacent to $\hat{z}_j^i$.

To form T_1, we create two $(4k+1)$-paths $Q_{\text{start}} = (\hat{s}_1, \ldots, \hat{s_k}, s_1, \ldots, s_k, s^*, s_{2k}, \ldots, s_{k+1}, \hat{s}_{2k}, \ldots, \hat{s}_{k+1})$ and $Q_{\text{end}} = (\hat{t}_k, \ldots, \hat{t}_1, t_k, \ldots, t_1, t^*, t_{k+1}, \ldots, \hat{t}_{2k}\ \hat{t}_{k+1}, \ldots, \hat{t}_{2k})$ such that s^* is adjacent to x_1 (i.e., to the "start" of I) and t^* is adjacent to x_n (i.e., to the "end" of I), and for each $i \in \{1, \ldots, 2k\}$:

- s_i is adjacent to $w^i_{2\alpha}$, i.e., the "end" of L_i is attached to s_i, and t_i is adjacent to z^i_1, i.e., the "start" of R_i is attached to t_i; and
- $\hat{s}_i$ is adjacent to $\hat{w}^i_{2\alpha}$, i.e., the "end" of $\hat{L}_i$ is attached to $\hat{s}_i$, and $\hat{t}_i$ is adjacent to $\hat{z}^i_1$, i.e., the "start" of $\hat{R}_i$ is attached to $\hat{t}_i$.

To build T_2, we use the same $8k$ caterpillars $L_i, R_i, \hat{L}_i, \hat{R}_i$ but attach them differently to the "central" path P of T_2. First we make an adjustment to Q_{start} and Q_{end}. In T_2, these become: $Q_{\text{start}} = (s_1, \ldots, s_k, \hat{s}_1, \ldots, \hat{s_k}, s^*, \hat{s}_{2k}, \ldots, \hat{s}_{k+1}\ s_{2k}, \ldots, s_{k+1})$ and $Q_{\text{end}} = (t_k, \ldots, t_1, \hat{t}_k, \ldots, \hat{t}_1, t^*, \hat{t}_{k+1}, \ldots, \hat{t}_{2k}, t_{k+1}, \ldots, \hat{t}_{2k})$ – this swap is done to highlight that in T_2 the $\hat{L}_i, \hat{R}_i$ caterpillars are closer to the central path P than the L_i, R_i caterpillars. Similar to T_1, in T_2, we have s^* adjacent to y_1 (i.e., the "start" of P) and t^* is adjacent to y_n (i.e., the "end" of P). The next part is where we see a difference regarding how we attach the caterpillars (L_i, R_i) of the increasing sequences vs. those $(\hat{L}_i, \hat{R}_i)$ of decreasing sequences. For each $i \in \{1, \ldots, 2k\}$:

- s_i is adjacent to w^i_1, i.e., the "start" of L_i is attached to s_i and as such L_i occurs "reversed" in T_2 with respect to T_1, and
- t_i is adjacent to $z^i_{2\alpha}$, i.e., the "end" of R_i is attached to t_i.

For each $i \in \{1, \ldots, k\}$:

- $\hat{s}_{k-i+1}$ ($\hat{s}_{2k-i+1}$) is adjacent to $\hat{z}^i_{2\beta}$ ($\hat{z}^{k+i}_{2\beta}$), i.e., the "end" of $\hat{R}_i$ ($\hat{R}_{i+k}$) is attached to $\hat{s}_{k-i+1}$ (and $\hat{s}_{2k-i+1}$) and as such $\hat{R}_i$ ($\hat{R}_{k+i}$) occurs "on the opposite side" in T_2 with respect to its location in T_1, and
- $\hat{t}_{k-i+1}$ ($\hat{t}_{2k-i+1}$) is adjacent to $\hat{w}^i_1$ ($\hat{w}^{k+i}_1$), i.e., the "start" of $\hat{L}_i$ ($\hat{L}_{k+i}$) is attached to $\hat{t}_{k-i+1}$ ($\hat{t}_{2k-i+i}$).

This completes the construction of T_1 and T_2 from π. It is easy to see that this construction can be performed in polynomial time and that the two trees contain in total precisely $32k^2 + 16k + 4n + 4$ vertices i.e., since $k \leq 2\sqrt{n}$, our instance of MRAF has $O(n)$ size.

Suppose π can be partitioned into α increasing sequences $\tau_1, \ldots, \tau_\alpha$ and β decreasing sequences $\sigma_1, \ldots, \sigma_\beta$. The leaf set corresponding to τ_i consists of $\{v_p : p \in \tau_i\}$ together with two leaves from each of L_j and R_j ($j \in \{1, \ldots, 2k\}$), i.e., $l^j_{2i-1}, l^j_{2i}, r^j_{2i-1}, r^j_{2i}$. Similarly, the leaf set corresponding to σ_i consists of $\{v_p : p \in \sigma_i\}$ together with two leaves from each of $\hat{L}_j$ and $\hat{R}_j$ ($j \in \{1, \ldots, 2k\}$), i.e., $\hat{l}^j_{2i-1}, \hat{l}^j_{2i}, \hat{r}^j_{2i-1}, \hat{r}^j_{2i}$. It can be verified that this is a valid solution to MRAF.

Now suppose that we have a solution $S_1, \ldots, S_k$ to MRAF (T_1, T_2). We need to show that this leads to a solution to the PIMS problem on π consisting of (at most) α increasing sequences and (at most) β decreasing sequences.

Lemma 1. *If some S_j uses three leaves of any caterpillar $C \in \{L_i, R_i, \hat{L}_i, \hat{R}_i : i \in \{1, \ldots, 2k\}\}$ then all elements of S_j are leaves of C.*

Proof: Let $C \in \{L_i, R_i, \hat{L}_i, \hat{R}_i : i \in \{1, \ldots, 2k\}\}$ and suppose S_j contains three leaves a, b, c of C, and one leaf d that is not in C. Without loss of generality, let a, b, c be ordered in increasing distance from the permutation-encoding part of T_1 (i.e., I); there may be a tie between b and c

but this does not matter. Observe that T_1 induces the quartet $da|bc$ but T_2 induces the quartet $cd|ba$ or $bd|ca$. This is because in T_2, C is attached to the rest of the tree by the opposite end used to attach C to the rest of T_1. Recall that a necessary and sufficient condition for S_j to be a component of a RAF is that they induce exactly the same set of quartet topologies in both trees (Observation 1(c)); contradiction. □

A consequence of this lemma is that if some S_j uses more than two leaves from any single one of our left/right caterpillars, then S_j can contain at most $\max\{2\alpha, 2\beta\} < 2k$ elements. In the next part we will see that every S_j must contain precisely $8k$ leaves from our left/right caterpillars in order to cover them all. In particular, this means that no S_j contains more than two leaves from any single left/right caterpillar. Note that, the total number of leaves is $n+4k\cdot 2\alpha+4k\cdot 2\beta = n+8k^2$ where the set of n leaves is $\{v_1, \ldots, v_n\}$ (i.e., corresponding to the permutation) and the $8k^2$ leaves are the leaves of the left/right caterpillars. We now define the following eight leaf sets related to our caterpillars $L_i, R_i, \hat{L}_i, \hat{R}_i$.

- $\mathcal{L}_1 = \{l : l \text{ is a leaf of some } L_i,\ i \in \{1, \ldots, k\}\}$,
- $\mathcal{L}_2 = \{l : l \text{ is a leaf of some } L_i,\ i \in \{k+1, \ldots, 2k\}\}$,
- $\mathcal{R}_1 = \{r : r \text{ is a leaf of some } R_i,\ i \in \{1, \ldots, k\}\}$,
- $\mathcal{R}_2 = \{r : r \text{ is a leaf of some } R_i,\ i \in \{k+1, \ldots, 2k\}\}$.

The definition of $\hat{\mathcal{L}}_1, \hat{\mathcal{L}}_2, \hat{\mathcal{R}}_1, \hat{\mathcal{R}}_2$ is analogous.

Lemma 2. *No S_j can contain five elements where each one belongs to a different set among:*

$$\mathcal{L}_1, \mathcal{L}_2, \mathcal{R}_1, \mathcal{R}_2, \hat{\mathcal{L}}_1, \hat{\mathcal{L}}_2, \hat{\mathcal{R}}_1, \hat{\mathcal{R}}_2.$$

Proof: Let $a, b, c, d, e \in S_j$ be chosen from five distinct sets from the eight listed. Consider the four set pairs $\{\hat{\mathcal{L}}_1, \mathcal{L}_1\}$, $\{\mathcal{L}_2, \hat{\mathcal{L}}_2\}$, $\{\hat{\mathcal{R}}_1, \mathcal{R}_1\}$, $\{\mathcal{R}_2, \hat{\mathcal{R}}_2\}$ which together partition the 8 sets. Note that the two sets in each pair are "adjacent" in T_1, but in T_2 they are on opposite sides of the permutation-encoding part P. By the pigeonhole principle at least one of these four set pairs must have elements from a, b, c, d, e in both sets of the pair. Without loss of generality, suppose $a \in \hat{\mathcal{L}}_1$ and $b \in \mathcal{L}_1$. Then in $T_1|\{a, b, c, d, e\}$, leaves $\{a, b\}$ form a cherry. However, due to a and b being on opposite sides of P in T_2, their options for forming a cherry there are highly constrained. If, say, $c \in \mathcal{R}_1$ then $|S_j| \leq 3$ because $\{a, c\}$ then forms a cherry in $T_2|\{a, b, c, d, e\}$ and the only way for a taxon (here a) to be in two distinct cherries in $T_1|S_j = T_2|S_j$, is if S_j has exactly 3 leaves. The same analysis holds if $c \in \hat{\mathcal{R}}_1$. Hence, again by the pigeonhole principle, at least one taxon from $\{c, d, e\}$ must be in $\hat{\mathcal{R}}_2 \cup \mathcal{L}_2$, and at least one taxon from $\{c, d, e\}$ must be in $\hat{\mathcal{L}}_2 \cup \mathcal{R}_2$. But then $\{a, b\}$ is certainly not a cherry in $T_2|\{a, b, c, d, e\}$. Hence, $T_1|\{a, b, c, d, e\} \neq T_2|\{a, b, c, d, e\}$; contradiction. □

Now, observe that a component S_j can contain at most $2k$ taxa from each of the 8 sets listed above. That is because each of the 8 sets is formed from k caterpillars (e.g., $\mathcal{L}_1$ is formed from the caterpillars $L_1, ..., L_k$) and each of these k caterpillars contributes at most 2 taxa to a RAF component. (If one of the k caterpillars contributed more than 2 taxa, we would automatically be limited to at most $2k$ taxa, by Lemma 1.) It follows from this that a component S_j can in total intersect with at most $4 \times 2k = 8k$ taxa ranging over all the 8 sets: intersecting with more would require intersecting with at least 5 of the 8 sets, which as we have shown in Lemma 2 is not possible.

Given that there are k components in the RAF, and T_1, T_2 have $n + 8k^2$ taxa, each of the k components must therefore contain *exactly* $8k$ taxa from the 8 sets, and each component intersects with *exactly* 4 of the 8 sets (as this is the only way to achieve $8k$). Consider:

Lemma 3. *The only way for S_j to intersect with 4 sets* and *a permutation-encoding taxon v_i, is if the 4 sets are* $\{\mathcal{L}_1, \mathcal{L}_2, \mathcal{R}_1, \mathcal{R}_2\}$ or $\{\hat{\mathcal{L}}_1, \hat{\mathcal{L}}_2, \hat{\mathcal{R}}_1, \hat{\mathcal{R}}_2\}$.[3]

Proof: Recall the four pairs $\{\hat{\mathcal{L}}_1, \mathcal{L}_1\}$, $\{\mathcal{L}_2, \hat{\mathcal{L}}_2\}$, $\{\hat{\mathcal{R}}_1, \mathcal{R}_1\}$, $\{\mathcal{R}_2, \hat{\mathcal{R}}_2\}$ which are "adjacent" in T_1. To this list we can add four other pairs, which are the sets which are "adjacent" in T_2. These are $\{\mathcal{L}_1, \hat{\mathcal{R}}_1\}$, $\{\hat{\mathcal{R}}_2, \mathcal{L}_2\}$, $\{\mathcal{R}_1, \hat{\mathcal{L}}_1\}$, $\{\hat{\mathcal{L}}_2, \mathcal{R}_2\}$. All these in total 8 pairs have the property that they are "adjacent" in one of the the two input trees, but split across the permutation-encoding part of the other. Suppose S_j contains taxa from both sets in one of these 8 pairs, $\{\hat{\mathcal{L}}_1, \mathcal{L}_1\}$ say (the other cases are symmetrical). Here S_j has only two ways to intersect with two further sets whilst ensuring the same topology in both trees: (1) $\{\hat{\mathcal{R}}_2, \mathcal{L}_2\}$, (2) $\{\hat{\mathcal{L}}_2, \mathcal{R}_2\}$. However, whether (1) is chosen or (2), if S_j contains some permutation-encoding taxon v_i, v_i will be in a different location (with respect to these four sets) in T_1 than in T_2, contradicting that $T_1|S_j = T_2|S_j$. This means that for each of the 8 pairs, S_j must avoid intersecting with both the sets in the pair. This leaves at most $4 \times 4 = 16$ possible valid combinations: one of the 4 pairs $\{\mathcal{L}_1, \mathcal{L}_2\}$, $\{\mathcal{L}_1, \hat{\mathcal{L}}_2\}$, $\{\hat{\mathcal{L}}_1, \mathcal{L}_2\}$, $\{\hat{\mathcal{L}}_1, \hat{\mathcal{L}}_2\}$ from the left side of T_1, and one of the 4 pairs $\{\mathcal{R}_1, \mathcal{R}_2\}$, $\{\mathcal{R}_1, \hat{\mathcal{R}}_2\}$, $\{\hat{\mathcal{R}}_1, \mathcal{R}_2\}$, $\{\hat{\mathcal{R}}_1, \hat{\mathcal{R}}_2\}$ from the right side of T_1. With some checking it can be verified that, of these 16, only $\{\mathcal{L}_1, \mathcal{L}_2, \mathcal{R}_1, \mathcal{R}_2\}$ and $\{\hat{\mathcal{L}}_1, \hat{\mathcal{L}}_2, \hat{\mathcal{R}}_1, \hat{\mathcal{R}}_2\}$ induce the same quartet topology in T_1 and T_2. □

The above lemma shows that the only way for S_j to intersect with four sets *and* a permutation-encoding taxon v_i, is if the four sets are $\{\mathcal{L}_1, \mathcal{L}_2, \mathcal{R}_1, \mathcal{R}_2\}$ or $\{\hat{\mathcal{L}}_1, \hat{\mathcal{L}}_2, \hat{\mathcal{R}}_1, \hat{\mathcal{R}}_2\}$. The permutation-encoding taxa v_i contained in components of the first type, necessarily induce increasing subsequences, and those contained in the second type are descending. There can be at most α components of the first type, and at most β of the second, which means that the permutation π can be partitioned into at most α increasing and β decreasing sequences. This concludes the proof of Theorem 1. □

4 Exact Algorithms and Limitations

We start here with some context on exact approaches via the PIMS problem. Note that, implicitly, there is a known exact algorithm with runtime $O(2^n)$ [15] for the PIMS problem. In fact the algorithm applies to any graph class where one can compute both the chromatic number and the clique cover number in polynomial time, e.g., as is the case for perfect graphs (which include permutation graphs). The algorithm is quite simple. For a given graph G, guess a bit-string where the entries are in bijection with the vertices of G and the value of an entry is 0 if the corresponding vertex is to be covered by an independent set and 1 otherwise (i.e., if that vertex is covered by a clique). For each such guess, compute the chromatic number of the subgraph induced by the 0-vertices, and the clique cover number of the graph induced by the 1-vertices. Heggerness et al. [15] use this algorithm as the starting point to obtain an FPT algorithm for the co-chromatic number problem on perfect graphs.

Despite the partial similarity between PIMS and MRAF (on caterpillars), it is not obvious how to lift the PIMS machinery to MRAF, even when the input is restricted to caterpillars.

[3] The GitHub page for this article includes an alternative and independent computational verification of this fact, based on enumerating all MASTs of two 9-taxon trees using constraint programming.

Nevertheless, in this section we present some baseline exact algorithms for MRAF and discuss further limitations.

In Section 4.1, we observe a single-exponential exact algorithm for MRAF and then show that when one input tree is a caterpillar, MRAF is in XP parameterized by k. Our XP algorithm relies on solving XP-many instances of a particular *constrained* version of the MRAF problem. In Section 4.2, we show that this constrained MRAF problem is itself NP-hard. Finally, in Section 4.3, we remark on the (lack of) applicability of known reduction rules for MAF, i.e., noting that an FPT algorithm for MRAF (even in the case where one tree is a caterpillar) will most likely require new ideas.

4.1 Single-Exponential and XP Algorithms

Recall that the NP-hard SetCover problem is to compute, for an input (U, F), where U is a set of elements, and F is a family of subsets of U, a minimum-size subset of F whose union is U [13]. Recall also that any subset of F whose union is U is called a *set cover* of U. We can view the MRAF problem as a SetCover problem, and use existing exponential time algorithms to solve it.

Observation 3. *Let T_1, T_2 be two unrooted binary phylogenetic trees on X. Let $U = X$ and let F be the set of all subsets of X that induce homeomorphic trees in T_1, T_2. Each RAF of T_1, T_2 is a set cover of (U, F), and each set cover of (U, F) can be transformed in polynomial time into a RAF of T_1, T_2 with the same or smaller size, by allocating each element of X to exactly one of the selected subsets of the set cover. In particular, any optimum solution to the set cover instance (U, F) can be transformed in polynomial time to yield an MRAF of T_1, T_2 of the same size.*

Proposition 1. *MRAF can be solved in time $O(c^n)$, $n = |X|$, for some constant c.*

Proof: The construction in Observation 3 yields $|U| = n$ and $|F| \leq 2^n$. Minimum set cover can be solved in time $O(2^{|U|} \cdot (|U| + |F|)^{O(1)})$ [4]. □

Proposition 1 concerns general instances. When one of the given trees is a caterpillar, we can show that MRAF is in XP (parameterized by the solution size k). We use dynamic programming to design an XP-algorithm for the problem. We will assume that $n > 3k$, as otherwise an arbitrary partition $S_1, \ldots, S_k$ of taxa where each S_i has at most three taxa is an MRAF. For $n > 3k$ it follows that if there is an MRAF for T_1 and T_2, then there always is an MRAF $S_1, \ldots, S_k$ where no S_i is a singleton. To see this, observe that for any MRAF with a singleton S_i, it must contain a component S_j with $|S_j| \geq 4$ (since $n > 3k$), and moving any element from S_j to S_i gives another MRAF where S_i is not a singleton (and S_j also not); repeating this operation iteratively to the resulting partition will eventually result in a partition with no singleton.

We let T_1 be the caterpillar, and T_2 an arbitrary tree. Similarly to our hardness result, we consider, without loss of generality, T_1 to consist of a spine (a path) $(y_1, \ldots, y_n)$ and leaves $v_1, \ldots, v_n$, where leaf v_i, $i = 1, \ldots, n$, is adjacent to vertex y_i. See Fig. 5 for an illustration. The spine naturally orders the leaves (up to arbitrarily breaking ties on the end cherry taxa) and this will guide our dynamic-programming approach. We write $u \prec v$ for two leaves u and v, if u appears before v in the considered ordering along the spine of T_1. We decide whether an MRAF $S_1, \ldots, S_k$ of T_1 and T_2 exists as follows: we enumerate over all possible pairs of vertices l_i, r_i, $i = 1, \ldots, k$, and check (compute) whether there exists an MRAF where the first leaf of S_i, $i = 1, \ldots, k$, is l_i and the last leaf of S_i is r_i. We call such an MRAF an *MRAF constrained by l_i, r_i, $i = 1, \ldots, k$*, or simply a *constrained MRAF* if l_i and r_i are clear from the context. If for one of the guesses

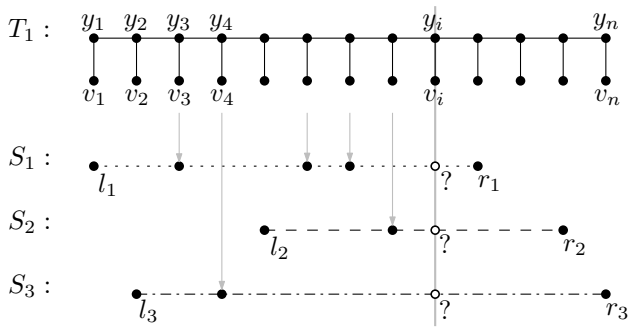

Figure 5: Caterpillar T_1 induces a natural ordering on the taxa (leaves). The gray vertical arrows assign each taxa to one of the sets S_1, S_2, S_3. At iteration i, the question marks denote possible assignment of v_i.

(enumerations) we find a constrained MRAF, we output YES, and otherwise (if for all guesses we do not find an MRAF) we output NO. In particular, for a given such guess we formalize the constrained MRAF problem as follows.

Constrained Maximum Relaxed Agreement Forest (constrained MRAF)
Input: Two unrooted binary phylogenetic trees T_1, T_2 where T_1 is a caterpillar and T_1 and T_2 have the same leaf set X, a number k, and a subset $\{l_1, \ldots, l_k, r_1, \ldots, r_k\}$ of X.
Task: Partition X into k sets $S_1, \ldots, S_k$ where for each $i \in \{1, \ldots, k\}$, $T_1|S_i = T_2|S_i$, $l_i, r_i \in S_i$, and for each $v \in S_i \setminus \{l_i, r_i\}$, $l_i \prec v \prec r_i$.

We now present our algorithm to decide, for input T_1, T_2, and pairs l_i, r_i, $i = 1, \ldots, k$, whether a constrained MRAF exists. We define $L := \{l_1, \ldots, l_k\}$ and $R := \{r_1, \ldots, r_k\}$. We view the process of computing a constrained MRAF $S_1, \ldots, S_k$ as an iteration over v_i, $i = 1, \ldots, n$, and assigning $v_i \notin (L \cup R)$ to one of the components $S_1, \ldots, S_k$ (every taxon $v_i \in (L \cup R)$ is already assigned to S_i). Fig. 5 illustrates this by the gray arrows from each taxon to one of the sets S_i.

When assigning v_i to one of $S_1, S_2, \ldots, S_k$, we need to obey certain constraints. Clearly, in the constrained MRAF, taxon v_i can only be assigned to component S_j if and only if $l_j \prec v_i \prec r_j$.

Tree T_2 further limits how taxon v_i can be assigned to components S_j (because we want that $T_1|S_j = T_2|S_j$). Clearly, for any $S_j \subset X$, $T_1|S_j$ is a caterpillar of maximum degree at most three. Thus, since l_j and r_j are the first and last leaf in $T_1|S_j$, they also need to be first and last in $T_2|S_j$. Hence, the inner vertices of the unique path P_j from l_j to r_j in T_2 is the subdivision of the spine of $T_2|S_j$. For a vertex $w \in P_j$ that has a neighbor $w' \notin P_j$ we define a *bag* B_w *of* P_j to be the maximal subtree of T_2 rooted at w' that does not include w (observe that w can have at most one neighbor $w' \notin P_j$). See Fig. 6 for an illustration. Observe that for any bag B_w of P_j, at most one taxon from B_w can be assigned to S_j. (Because if two taxa $v_a, v_{a'} \in B_w$, $a < a'$, are assigned to S_j then $l_j v_a | v_{a'} r_j$ will not be a quartet of $T_2|S_j$, while it is a quartet of $T_1|S_j$, and thus $T_1|S_j \neq T_2|S_j$.) The path P_j of $T_2|S_j$ naturally orders all bags of P_j. It follows that for two bags B_w and $B_{w'}$ where B_w appears before $B_{w'}$ in the ordering along P_j, we can select taxa $v_a \in B_w$ and $v_b \in B_{w'}$ into S_j if and only if $v_a \prec v_b$, i.e., if v_a appears before v_b in the caterpillar T_1. We write $v \prec_{P_j} v'$ for taxa v, v' such that v is from a bag B_w and v' is from a bag $B_{w'}$, and B_w appears before bag $B_{w'}$ along path P_j. Relation $\prec_{P_j}$ is thus a partial ordering of X, where any two taxa from the same bag are uncomparable. Observe now that any assignment of taxa to

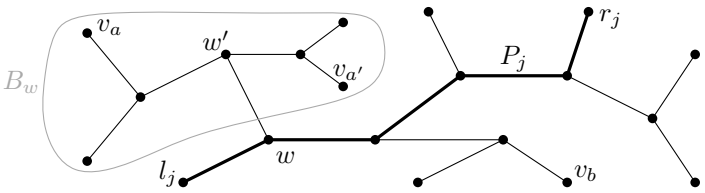

Figure 6: A bag B_w on the (l_j, r_j)-path P_j. At most one of $v_a, v_{a'}$ can occur in S_j.

S_j that satisfies the above conditions, i.e., (i) for every $v_i \in S_j$, $l_j \prec v_i \prec r_i$, (ii) for every bag B_w of P_j there is at most one vertex $v_i \in B_w \cap S_j$, and (iii) for any two taxa $v_p, v_q \in S_j$, $p < q$, $v_p \prec_{P_j} v_q$, we have $T_1|S_j = T_2|S_j$.

We can thus assign taxon v_i, where $l_j \prec v_i \prec r_j$, to component S_j whenever the previously assigned taxon $v_{i'}$ to S_j satisfies $v_{i'} \prec_{P_j} v_i$. We thus do not need to know all previously assigned taxa to S_j, only the last assigned. We compute a (partial) constrained MRAF for taxa $X_i := \{v_1, v_2, \ldots, v_i\} \cup (L \cup R)$ iteratively for $i = 0, 1, 2, \ldots, n$. We set $X_0 := L \cup R$. For $\vec{z} = (z_1, \ldots, z_k) \in (X_i \setminus R)^k$ and $i = 0, 1, \ldots, n$ we define a Boolean function $\text{craf}^{(i)}(\vec{z})$ as follows: $\text{craf}^{(i)}(\vec{z}) :=$ TRUE if and only if there exists a constrained MRAF $S_1^i, S_2^i, \ldots, S_k^i$ of X_i such that the last taxon from $X_i \setminus R$ in S_ℓ^i, $\ell = 1, \ldots, k$, is z_ℓ.

Clearly, $\text{craf}^{(0)}(\vec{z}) =$ TRUE if and only if $\vec{z} = (l_1, l_2, \ldots, l_k)$. Furthermore, $\text{craf}^{(i)}(z_1, \ldots, z_k)$ is FALSE, whenever any of z_j is left of l_j or right of r_j (i.e., whenever $z_j \prec l_j$ or $r_j \prec z_j$). Also, $\text{craf}^{(i)}(z_1, \ldots, z_k)$ is FALSE, whenever $z_j = z_\ell$ for any two $j \neq \ell$. Also observe that if no z_j is equal to taxon v_i, then $\text{craf}^{(i)}(z_1, \ldots, z_k)$ is FALSE, because in every partition of X_i, the last element v_i of $X_i \setminus R$ needs to be last in one of the sets S_j. Now, whenever one of z_j is equal to v_i, the function craf^i can be computed recursively as:

$$\text{craf}^{(i)}(z_1, \ldots, z_{j-1}, z_j = v_i, z_{j+1} \ldots, z_k) = \bigvee_{\substack{z \in X_{i-1} \setminus R \\ z \prec_{P_j} v_i}} \text{craf}^{(i-1)}(z_1, \ldots, z_{j-1}, z, z_{j+1}, \ldots, z_k). \tag{1}$$

This recurrence follows simply because removing v_i from every constrained MRAF of X_i gives a constrained MRAF of X_{i-1}. Now we can compute $\text{craf}^{(i)}$ bottom-up using the dynamic programming. For every value $i = 1, \ldots, n$ we enumerate $O(n^k)$ vectors $\vec{z}$, and compute the value $\text{craf}^{(i)}(\vec{z})$ using the recursive relation from Eq. (1) (where applicable), thus looking at at most $O(n)$ different entries of $\text{craf}^{(i-1)}$. This thus leads to the overall runtime of $O(n \cdot n^k \cdot n)$ for an algorithm checking whether a constrained MRAF exists. Accounting for the enumeration of the $O(n^{2k})$ pairs l_i, r_i, $i = 1, \ldots, k$, results in the following theorem.

Theorem 2. *MRAF can be computed in time $O(n^{3k+2})$ whenever one of the trees is a caterpillar.*

Observe that the overall runtime of the algorithm has two sources (reasons) for the runtime component of $n^{\theta(k)}$: the enumeration of the k pairs l_i, r_i, $i = 1, \ldots, k$, and the enumeration of all $O(n^k)$ vectors $\vec{z}$ for the computation of $\text{craf}^{(i)}(\vec{z})$ by the dynamic programming. It is natural to ask whether both of these "run-times" are inevitable. In the following, we show that this may be the case for the second source of the exponential term, by showing that computing the constrained MRAF for two caterpillar trees is NP-hard.

4.2 Hardness of Constrained MRAF

Recall that we defined the computational problem *constrained* MRAF to be the computational problem MRAF with input trees T_1, T_2, and parameter k, and additional input of k pairs of taxa l_i, r_i, $i = 1, \ldots, k$, where the input tree T_1 is a caterpillar, and where every solution $S_1, S_2, \ldots, S_k$ is constrained to have, for every $i = 1, \ldots, k$, l_i as the first vertex of S_i and r_i as the last vertex of S_i, in the ordering induced by the spine of the caterpillar T_1. In this subsection we prove that constrained MRAF is NP-hard.

To prove the theorem, we first show that an auxiliary variant of PIMS is NP-hard, and then reduce from that variant to constrained MRAF. Recall that PIMS is a computational problem that asks, for an input permutation (a sequence) π of n integers $1, 2, \ldots, n$ and for an input integer $k > 0$, whether π can be partitioned into at most k monotone subsequences. Observe now that the variant of PIMS where instead of one parameter k we have two parameters k_1 and k_2 and where we ask whether the input sequence π can be partitioned into at most k_1 increasing and at most k_2 decreasing subsequences is NP-hard: having a polynomial-time algorithm for the variant, we can solve any instance of PIMS by iterating over $k_1 = 0, 1, \ldots, k$ and solving the variant with input π, k_1, and k_2 where $k_2 = k - k_1$, and answering YES if and only if one of the iterations answers YES. We refer to the variant as PIMS-(up,down).

Theorem 3. *Constrained MRAF is NP-hard, even when both input trees are caterpillars.*

Proof: We reduce the NP-hard problem PIMS-(up,down) to constrained MRAF. Let π, k_1, k_2, be the input to PIMS-(up,down). We set $k = k_1 + k_2$ and create T_1, T_2, and $l_1, \ldots, l_k$ and $r_1, \ldots, r_k$ – the input to constrained MRAF – as follows.

Each of T_1 and T_2 contains $k_1 + k_2 + n + k_1 + k_2 = 2k + n$ leaves. Figure 7 illustrates the labeling of the leaves of T_1 and T_2. The first k_1 leaves of T_1, ordered along the spine of T_1, are labeled with taxa $-1, -2, \ldots, -k_1$, respectively, and the next k_2 leaves of T_1 are labeled with taxa $h + 1, h + 2, \ldots, h + k_2$, respectively, where h is a large number,[4] say, $2n$. The next n leaves of T_1 are labeled, in order, by the elements of the permutation π; the $(k_1 + K_2 + i)$-th leaf of T_1 is labeled by π_i, for $i = 1, \ldots, n$. After that, next k_1 leaves of T_1 are labeled, respectively, with taxa $n + 1, n + 2, \ldots, n + k_1$. Finally, the last k_2 leaves of T_1 are labeled, respectively, with taxa $-(h + 1), -(h + 2), \ldots, -(h + k_2)$.

We set $l_1, l_2, \ldots, l_k$ to be the first k leaves of T_1, respectively, and we set $r_1, r_2, \ldots, r_k$ to be the last k leaves of T_1, respectively. Observe that $l_i < r_i$ for $i \leq k_1$ and that $l_i > r_i$ for $k_1 + 1 \leq i \leq k_1 + k_2$.

We set the leaves of T_2 as follows. The first k_1 leaves of T_2 are identical to the first k_1 leaves of T_1 (i.e., leaves $l_1, \ldots, l_{k_1}$). The next k_2 leaves of T_2 are the last k_2 leaves of T_1, i.e., leaves $r_{k_1+1}, r_{k_1+2}, \ldots, r_{k_1+k_2}$. Then, T_2 contains n leaves labeled, respectively, $1, 2, 3, \ldots, n$. The last k leaves of T_2 are, respectively, leaves $r_1, \ldots, r_{k_1}$, followed by leaves $l_{k_1+1}, l_{k_1+2}, \ldots, l_{k_1+k_2}$.

To complete the proof, we now show that π, k_1, k_2 is a YES-instance of PIMS-(up,down) if and only if the constructed trees T_1, T_2 and taxa $l_1, \ldots, l_k$, $r_1, \ldots, r_k$ is a YES-instance of constrained MRAF.

Let $S_1, \ldots, S_k$ be a solution (subsequences of π) to PIMS-(up,down), where $S_1, \ldots, S_{k_1}$ are increasing, and $S_{k_1+1}, \ldots, S_k$ are decreasing subsequences. For every $i = 1, \ldots, n$, we prepend l_i to S_i and append r_i to S_i, creating a *monotone* sequence S'_i that starts with l_i and ends with r_i. Observe now that, when seeing S'_i as sets of taxa, $T_1|S'_i = T_2|S'_i$, and thus $S'_1, \ldots, S'_k$ is a solution to the created instance of constrained MRAF.

[4] We only need that $n + k_1 < h + 1$.

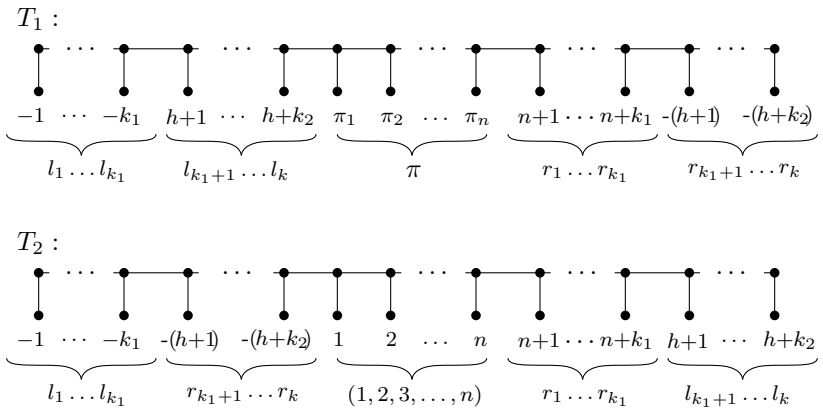

Figure 7: Construction of the input trees T_1, T_2 for the constrained MRAF from a given input (π, k_1, k_2) of PIMS-(up,down).

For the other direction, let $S'_1, \ldots, S'_k$ (sets of taxa) be a solution to the created instance of constrained MRAF. Observe that every S'_i contains l_i, r_i, and some elements from $1, 2, \ldots, n$. Consider S'_i, $1 \leq i \leq k_1$. Then, since $l_i < 1 < n < r_i$, the leaves of $T_2|S'_i$ form an increasing sequence (when considered along the spine from left to right). Since $T_1|S'_i = T_2|S'_i$, $T_1|S'_i$ also forms an increasing sequence, and thus this S'_i induces an increasing subsequence S_i of π (elements of $S'_i \setminus \{l_i, r_i\}$). Consider now S'_i, $k_1 < i \leq k$. Then, since $r_i < 1 < n < l_i$, the leaves of $T_2|S'_i$ form an increasing sequence (when considered along the spine of T_2 from r_i to l_i; left-to-right in Fig. 7). Thus, the reverse sequence (leaves of $T_2|S'_i$ considered from right to left, i.e., from l_i to r_i) forms a decreasing subsequence. Since $T_1|S'_i = T_2|S'_i$, $T_1|S'_i$ also forms a decreasing subsequence in T_1 from l_i to r_i, and thus induces a decreasing subsequence S_i of π (elements of $S'_i \setminus \{l_i, r_i\}$). Since $S'_1, \ldots, S'_k$ is a partition of all taxa, $S_1, \ldots, S_k$ is a partition of (the elements of) π, and $S_1, \ldots, S_{k_1}$ induce increasing subsequences of π, and $S_{k_1+1}, \ldots, S_k$ induce decreasing subsequences of π, and thus $S_1, \ldots, S_k$ (when seen as subsequences of π) is a solution to PIMS-(up,down). □

4.3 Reduction Rules

Finally, before turning to approximation algorithms, we consider reduction rules, which are an important part of the algorithm designer's toolbox when implementing exact algorithms. We consider some reduction rules that have been used for the MAF problem.

Observe that if two trees T_1, T_2 have a common cherry $\{a, b\}$, the well-known *common cherry reduction* – in each tree, we delete a, b and relabel their parent ab – preserves MRAF. When applied to exhaustion this is called the *subtree* reduction rule. This is known to be very effective in the phylogenetics literature when pre-processing input trees to reduce their size – and will thus help with (exact) computation of MRAF in practice, given its NP-hardness.

On the other hand, the much-studied *common-chain* reduction rule is not guaranteed to preserve MRAF. The definition of this reduction rule is rather technical (see e.g. [30] for a formal definition) but essentially it shrinks a long common caterpillar to a shorter one in both trees. Fig. 8 illustrates (as can be verified by hand or using the exact solver we provide in Section 6) that shortening the common chain lowers MRAF. This is in contrast to MAF, where both the subtree and common chain reduction rules preserve MAF, and in fact yield a linear kernel [1]. We note that if a pair of

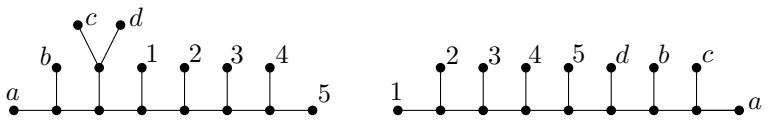

Figure 8: Two trees with a common chain of size 5, comprising the leaves $C = \{1, 2, 3, 4, 5\}$. The MRAF of these trees is 3. Deleting any one taxon from C (thus shortening the common chain to length 4) yields a pair of trees with MRAF equal to 2. In contrast, such a transformation is known to preserve MAF.

trees has no common cherries or common chains, the ratio $\frac{n}{\text{MRAF}}$ can still be arbitrarily large. For example, two caterpillars of the form $1, z, 2, y, 3, x...$ and $1, a, 2, b, 3, c...$ have no common cherries or chains, and an MRAF of size 2, but an arbitrarily large number of leaves. Hence, any attempt to establish a fixed parameter tractability result for MRAF via kernelization must consider a different strategy.

5 Approximation Algorithms

We now provide a polytime approximation algorithm for MRAF (Lemma 4) and relate the approximability of PIMS to that of MRAF on caterpillars (Lemma 5).

Lemma 4. *There is a $O(\log n)$-approximation algorithm for computing MRAF, where $n = |X|$. This algorithm cannot be better than a $(4/3)$-approximation algorithm.*

Proof: Given an instance (U, F) of Set Cover, the natural greedy algorithm yields a $O(\log |U|)$ approximation. Recall the encoding of MRAF as a Set Cover instance in Observation 3. We cannot construct this directly, since $|F|$ is potentially exponential in n, but this is not necessary to simulate the greedy algorithm. Let X' be the set of currently uncovered elements of X, initially $X = X'$. We compute a MAST of $T_1|X'$ and $T_2|X'$ in polynomial time [29]. Let S be the leaf-set of this MAST; we add this to our RAF. We then delete S from X' and iterate this process until X' is empty. Fig. 9 shows that this algorithm cannot be better than a $(4/3)$-approximation. □

Lemma 5. *Let π be a permutation of $\{1, \ldots, n\}$ and let T_1 and T_2 be two caterpillars on leaves $\{1, \ldots n\}$ where T_1 is the identity caterpillar and the ith leaf of T_2 is $\pi(i)$. For any solution to the MRAF problem of size k, there is a corresponding solution to the PIMS problem of size at most $k + 2\sqrt{2k}$.*

Proof: We start with an agreement forest for the two caterpillars; each component is itself a caterpillar. We "cut off" one leaf from each end of the components in this forest. (This is because the "interior" of each component induces a monotonic subsequence, but the cherries at the end of each component potentially violate this). This leaves behind a subpermutation of π of length $2k$, which can always be partitioned into at most $2\sqrt{2k}$ monotone subsequences. □

We can create an instance of PIMS from a caterpillar instance of MRAF by treating one caterpillar as the identity and the other as the permutation. Any solution for this PIMS instance yields a feasible MRAF solution. Hence:

$$\text{MRAF} \leq \text{PIMS} \leq \text{MRAF} + 2\sqrt{2 \cdot \text{MRAF}}.$$

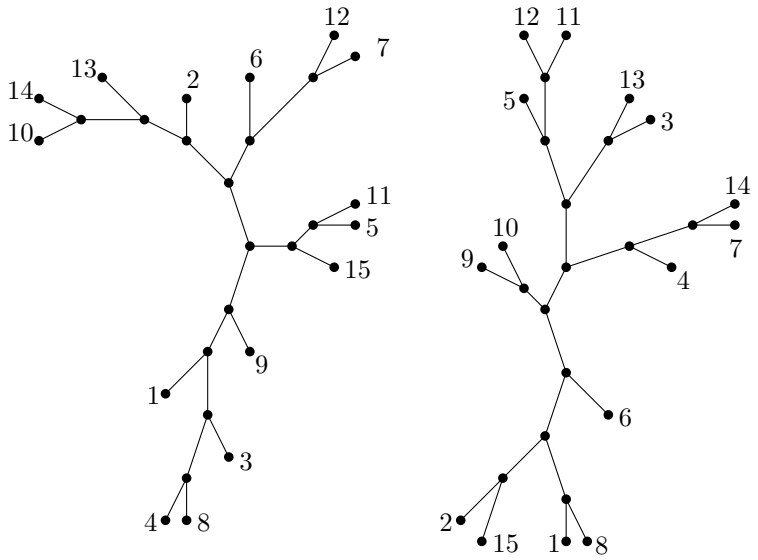

Figure 9: These two trees have an MRAF of size 3, $\{1, 7, 8, 10, 12, 15\}, \{5, 11, 13, 14\}, \{2, 3, 4, 6, 9\}\}$ and a unique MAST of size 7: $\{1, 5, 7, 8, 9, 11, 14\}$. However, any RAF that contains a component of size 7, has 4 or more components. This has been verified computationally. Due to the uniqueness of the MAST, this shows that even an algorithm that can non-deterministically decide what the 'correct' MAST is to choose next, cannot always solve MRAF optimally. We also have a very similar example, also proving a (4/3) lower bound, where both trees are caterpillars. This example, and the aforementioned computational proof, are available on the GitHub page.

Recall that MRAF on caterpillars is in XP by Theorem 2.

PIMS is also in XP. Specifically, the PIMS problem is equivalent to the *co-chromatic number* problem on permutation graphs, i.e., partitioning the vertices of a permutation graph into cliques and independent sets. When a graph can be partitioned into r cliques and s independent sets it is sometimes called an (r, s)-*split graph*. It is known that the perfect (r, s)-split graphs can be characterized by a finite set of forbidden induced subgraphs [19]. This implies that their recognition is in XP parameterized by r and s, i.e., when r and s are fixed, (r, s)-split graphs can be recognized in polynomial time—this was later improved to FPT [15]. These XP results are relevant here because they allow us to show a strong relationship between the approximability of PIMS and MRAF on caterpillars.

Lemma 6. *If one of the problems PIMS or MRAF on caterpillars has a polynomial time c-approximation, c constant, then for each fixed constant $\epsilon > 0$ the other has a polynomial-time $(c + \epsilon)$-approximation. In particular, PIMS permits a Polynomial-Time Approximation Scheme (PTAS) if and only if MRAF on caterpillars permits a PTAS.*

Proof: Given a polynomial-time c-approximation for MRAF, and $\epsilon > 0$, we first use the XP algorithm for PIMS to check in polynomial time whether PIMS $\leq \frac{8c}{\epsilon^2}$. If so, we are done. Otherwise we use the c-approximation to MRAF to obtain a RAF, and then transform it to a solution to

PIMS of size at most $c \cdot \text{MRAF} + 2\sqrt{2c \cdot \text{MRAF}}$. We need to show that

$$\frac{c \cdot \text{MRAF} + 2\sqrt{2c \cdot \text{MRAF}}}{\text{PIMS}} \leq (c + \epsilon).$$

Using $\text{PIMS} > \frac{8c}{\epsilon^2}$ and the fact that $\text{MRAF} \leq \text{PIMS}$ establishes the inequality.

In the other direction, suppose we have a polynomial-time c-approximation for PIMS. We first use the XP algorithm for MRAF to check in polynomial time whether $\text{MRAF} \leq \frac{8}{\epsilon^2}$. If so, we are done. Otherwise, we use this to obtain a solution to PIMS of size at most $c \cdot \text{PIMS}$, and thus a RAF of the same size. We need to show that $\frac{c \cdot \text{PIMS}}{\text{MRAF}}$ is at most $(c + \epsilon)$. Recalling that $\text{PIMS} \leq \text{MRAF} + 2\sqrt{2 \cdot \text{MRAF}}$, it is sufficient to show,

$$\frac{c(\text{MRAF} + 2\sqrt{2 \cdot \text{MRAF}})}{\text{MRAF}} \leq (c + \epsilon).$$

It is straightforward to verify that the inequality holds when $\text{MRAF} > \frac{8}{\epsilon^2}$. □

PIMS has a polynomial-time 1.71-approximation [12]. Hence, for every constant $\epsilon > 0$ MRAF on caterpillars has a polynomial-time $(1.71 + \epsilon)$-approximation. We remark that currently it is unknown whether PIMS is APX-hard. This means that the above observations do not automatically imply the NP-hardness of MRAF on caterpillars via PIMS.

6 Implementation and Experimental Observations

MRAF can be modelled as the *weak chromatic number* of hypergraph: the minimum number of colours assigned to vertices, such that no hyperedge is monochromatic. The set of vertices is X and there is a hyperedge $\{a, b, c, d\}$ whenever the two trees have a different quartet topology on $\{a, b, c, d\}$, leveraging Observation 1.

We implemented this as a constraint program (CP) using MiniZinc [25]. For trees with ≤ 30 leaves the CP solves quickly. Code is available at `https://github.com/skelk2001/relaxed_agreement_forests`. We used this to extend the analysis of [18] on the grass dataset of [14], consisting of fifteen pairs of trees. See Table 1; to facilitate comparison, the rows are primarily ordered by MAF, and then secondarily by MRAF. As expected MRAF grows more slowly than MAF: MAF increases from 2 to 16, but MRAF only increases from 2 to 5. (In turn, MRAF grows more quickly than the lower bound of $\lceil n/\text{MAST} \rceil$, which only increases from 2 to 3 for the very last pair of trees). An FPT algorithm parameterized by MRAF, if it exists, might therefore scale well in practice. FPT algorithms for MAF struggle for MAF ≥ 25 [30]. In fact, MRAF seems more comparable to the *treewidth* of the *display graph* of the input tree pair $tw(D)$ (where D is obtained by identifying vertices with the same leaf label: the treewidth of this graph is bounded by a function of MAF [18]). We obtained similar results on a more challenging dataset comprising the 163 tree pairs from the dataset in [30] that had at most 50 leaves after pre-processing. This dataset is more challenging because many of the tree pairs are highly dissimilar. The results are summarized in Table 2. It is worth noting that in this table the average values for MAF and MRAF differ by (roughly) a factor of 3, which is similar to the trend observed in the lower rows of Table 1. The standard deviation for MRAF is smaller than the corresponding value for MAF but this is presumably only because MRAF values are somewhat smaller than MAF values. Next, the third row of Table 2 highlights that for at least one pair of trees, the additive gap between MRAF and MAF was 20. This is very large, given that the trees in this dataset all have at most

50 leaves. Observation 2 had already established that for specially constructed tree pairs the gap between MRAF and MAF can be arbitrarily large; this example shows that also in an *empirical* setting the gap can be very large. Finally, the fourth row of Table 2 shows that, on average, the treewidth of the display graph (7.28) is larger than MRAF (4.47). The fifth row shows that on this dataset $tw(D)$ is always sandwiched between MRAF-1 and MRAF$+7$, and that on average $tw(D)$ is additively 2.81 units larger than MRAF. The fact that $tw(D) \geq$ MRAF-1 is particularly interesting, and warrants further investigation.

7 Discussion and Open Problems

It remains unclear whether it is NP-hard to compute MRAF on caterpillars, although it seems likely. Can the finite forbidden obstructions that characterize solutions to PIMS be mapped to MRAF on caterpillars and then generalized to general trees? Indeed, how far can MRAF be viewed as a generalization of the PIMS problem to partial orders? Is MRAF on caterpillars FPT? Does it (or PIMS) have a polynomial kernel? What should reduction rules look like, given that rules for MAF seem of limited use? Strikingly, we do not know whether it is NP-hard to determine whether MRAF ≤ 2 for two general trees, so the FPT landscape is also unclear. How far can the logarithmic approximation for MRAF on general trees, and the 1.71 approximation for MRAF on caterpillars (equivalently, PIMS) be improved? What is the relationship between MRAF and the treewidth of the display graph, if any? How does MRAF behave on multiple and/or non-binary trees? Such generalisations already exist for MAF [6]. The emergence of cancer phylogenetics (see e.g. [10]) has led to variants of 'agreement' problems where not just leaves, but also internal nodes are (or can be) labelled [26]. How can the RAF model be adapted to such a context, and how challenging are the resulting optimisation problems?

Finally, it will be instructive to elucidate the biological interpretation of this model. MRAF has some superficial resemblance to the type of 'overlap' phenomena that occur when modelling incomplete lineage sorting and, relatedly, when undertaking species-gene tree reconciliation: namely, when several branches of a gene tree pass through a single branch of the species tree within which it is embedded [9, 24, 17]. However, MRAF does not distinguish between a species tree and a gene tree, and the overlap patterns induced by (say) incomplete lineage sorting are due to the *same* gene tree embedding overlapping with itself. In contrast, the overlap induced by solutions to MRAF is the result of embedding the common building blocks of two trees into one of the two input trees - and either input tree can be chosen. Hence, there is still some work to be done on the biological modelling front. In the meantime, and as noted in the introduction, MRAF can be used as a model-agnostic decomposition tool. Given two topologically-distinct trees which have possibly been constructed by merging common subtrees in different but unknown ways, MRAF can provide a most-parsimonious decomposition of the two trees into common building blocks, and thus perhaps offer clues about how the original input trees were constructed.

Acknowledgements

We thank the two anonymous reviewers for their constructive feedback which helped to improve the article.

tree pair	$\|X\| = n$	**MAF**	**MRAF**	**tw(D)**	**MAST**	$\lceil n/\text{MAST} \rceil$
rpoC_waxy	10	2	2	3	8	2
phyB_waxy	14	3	2	3	11	2
rbcL_waxy	12	4	2	3	9	2
phyB_rpoC	21	5	2	3	15	2
phyB_rbcL	21	5	3	3	14	2
ndhF_waxy	19	5	3	4	11	2
waxy_ITS	15	6	3	4	10	2
ndhF_phyB	40	7	3	3	30	2
rbcL_rpoC	26	7	4	5	14	2
ndhF_rbcL	36	7	4	3	20	2
phyB_ITS	30	8	4	4	17	2
ndhF_rpoC	34	9	3	5	20	2
rbcL_ITS	29	11	4	5	17	2
rpoC_ITS	31	11	4	6	16	2
ndhF_ITS	46	16	5	6	20	3

Table 1: Comparison of MAF and MRAF for the fifteen tree pairs in the data set [14] analysed in [18]. We also include MAST, the lower bound on MRAF given by $\lceil \frac{n}{\text{MAST}} \rceil$, and $tw(D)$ which is the treewidth of the display graph obtained from the tree pair. The table is ordered by MAF values (in red), and then by MRAF (in blue).

	Min	**Max**	**Avg**	**Stdev**
MAF	5	27	13.07	7.19
MRAF	2	8	4.47	1.27
MRAF-MAF	-20	-1	-8.60	6.14
tw	3	13	7.28	2.46
tw-MRAF	-1	7	2.81	1.58

Table 2: Summary statistics for the 163 tree pairs obtained from the dataset in [30] by restricting to trees which, after subtree reduction, have at most 50 taxa.

References

[1] B. Allen and M. Steel. Subtree transfer operations and their induced metrics on evolutionary trees. *Annals of Combinatorics*, 5:1–15, 2001. doi:10.1007/s00026-001-8006-8.

[2] R. Atkins and C. McDiarmid. Extremal distances for subtree transfer operations in binary trees. *Annals of Combinatorics*, 23:1–26, 2019. doi:10.1007/s00026-018-0410-4.

[3] R. Bar-Yehuda and S. Fogel. Partitioning a sequence into few monotone subsequences. *Acta Informatica*, 35(5):421–440, 1998. doi:10.1007/s002360050126.

[4] A. Björklund, T. Husfeldt, and M. Koivisto. Set partitioning via inclusion-exclusion. *SIAM Journal on Computing*, 39(2):546–563, 2009. doi:10.1137/070683933.

[5] M. Bordewich and C. Semple. On the computational complexity of the rooted subtree prune and regraft distance. *Annals of Combinatorics*, 8(4):409–423, 2005. doi:10.1007/s00026-004-0229-z.

[6] L. Bulteau and M. Weller. Parameterized algorithms in bioinformatics: an overview. *Algorithms*, 12(12):256, 2019. doi:10.3390/a12120256.

[7] P. Buneman. The recovery of trees from measures of dissimilarity. In F. Hodson, D. Kendall, and P. Tautu, editors, *Mathematics in the Archaeological and Historical Sciences*, pages 387–395. Edinburgh University Press, Edinburgh, 1971.

[8] J. Chen, F. Shi, and J. Wang. Approximating maximum agreement forest on multiple binary trees. *Algorithmica*, 76:867–889, 2016. doi:10.1007/s00453-015-0087-6.

[9] J. Degnan and N. Rosenberg. Gene tree discordance, phylogenetic inference and the multispecies coalescent. *Trends in Ecology & Evolution*, 24(6):332–340, 2009. doi:10.1016/j.tree.2009.01.009.

[10] Z. DiNardo, K. Tomlinson, A. Ritz, and L. Oesper. Distance measures for tumor evolutionary trees. *Bioinformatics*, 36(7):2090–2097, 11 2019. doi:10.1093/bioinformatics/btz869.

[11] P. Erdős and G. Szekeres. A combinatorial problem in geometry. *Compositio Mathematica*, 2:463–470, 1935. doi:10.1007/978-0-8176-4842-8_3.

[12] F. Fomin, D. Kratsch, and J.-C. Novelli. Approximating minimum cocolorings. *Information Processing Letters*, 84(5):285–290, 2002. doi:10.1016/S0020-0190(02)00288-0.

[13] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.

[14] G. P. W. Group and N. B. et al. Phylogeny and subfamilial classification of the grasses (poaceae). *Annals of the Missouri Botanical Garden*, 88(3):373–457, 2001. doi:10.2307/3298585.

[15] P. Heggernes, D. Kratsch, D. Lokshtanov, V. Raman, and S. Saurabh. Fixed-parameter algorithms for cochromatic number and disjoint rectangle stabbing via iterative localization. *Information and Computation*, 231:109–116, 2013. doi:10.1016/j.ic.2013.08.007.

[16] J. Hein, T. Jiang, L. Wang, and K. Zhang. On the complexity of comparing evolutionary trees. *Discrete Applied Mathematics*, 71(1-3):153–169, 1996. doi:10.1016/S0166-218X(96)00062-5.

[17] L. v. Iersel, M. Jones, and C. Scornavacca. Improved maximum parsimony models for phylogenetic networks. *Systematic biology*, 67(3):518–542, 2018. doi:10.1093/sysbio/syx094.

[18] S. Kelk, L. van Iersel, C. Scornavacca, and M. Weller. Phylogenetic incongruence through the lens of monadic second order logic. *Journal of Graph Algorithms and Applications*, (2):189–215, 2016. doi:10.7155/jgaa.00390.

[19] A. Kézdy, H. Snevily, and C. Wang. Partitioning permutations into increasing and decreasing subsequences. *Journal of Combinatorial Theory Series A*, 73(2):353–359, 1996. doi:10.1016/S0097-3165(96)80012-4.

[20] E. Kubicka, G. Kubicki, and F. McMorris. On agreement subtrees of two binary trees. *Congressus Numerantium*, pages 217–217, 1992.

[21] M. Kuhner and J. Yamato. Practical performance of tree comparison metrics. *Systematic Biology*, 64(2):205–214, 2015. doi:10.1093/sysbio/syu085.

[22] P. Lemey, M. Salemi, and A.-M. Vandamme. *The phylogenetic handbook: a practical approach to phylogenetic analysis and hypothesis testing*. Cambridge U.P., 2009. doi:10.1017/CBO9780511819049.

[23] A. Markin. On the extremal maximum agreement subtree problem. *Discrete Applied Mathematics*, 285:612–620, 2020. doi:10.1016/j.dam.2020.07.007.

[24] L. Nakhleh. Computational approaches to species phylogeny inference and gene tree reconciliation. *Trends in Ecology & Evolution*, 28(12):719–728, 2013. doi:10.1016/j.tree.2013.09.004.

[25] N. Nethercote, P. Stuckey, R. Becket, S. Brand, G. Duck, and G. Tack. MiniZinc: Towards a standard CP modelling language. In *CP2007*, pages 529–543, 2007. doi:10.1007/978-3-540-74970-7_38.

[26] Y. Qi and M. El-Kebir. Sapling: Inferring and Summarizing Tumor Phylogenies from Bulk Data Using Backbone Trees. In S. P. Pissis and W.-K. Sung, editors, *24th International Workshop on Algorithms in Bioinformatics (WABI 2024)*, volume 312 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 7:1–7:19, Dagstuhl, Germany, 2024. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.WABI.2024.7.

[27] C. Semple and M. Steel. *Phylogenetics*. Oxford University Press, 2003.

[28] M. Steel. *Phylogeny: Discrete and Random Processes in Evolution*. SIAM, 2016. doi:10.1137/1.9781611974485.

[29] M. Steel and T. Warnow. Kaikoura tree theorems: Computing the maximum agreement subtree. *Information Processing Letters*, 48(2):77–82, 1993. doi:10.1016/0020-0190(93)90181-8.

[30] R. van Wersch, S. Kelk, S. Linz, and G. Stamoulis. Reflections on kernelizing and computing unrooted agreement forests. *Annals of Operations Research*, 309(1):425–451, 2022. doi:10.1007/s10479-021-04352-1.

[31] K. Wagner. Monotonic coverings of finite sets. *Journal of Information Processing and Cybernetics*, 20(12):633–639, 1984.