The difficulty of constructing a leaf-labelled tree including or avoiding given subtrees

Meei Pyng Ng[*]
Department of Mathematics and Statistics
University of Melbourne
Parkville, VIC 3052, AUSTRALIA

Mike Steel
Department of Mathematics and Statistics
University of Canterbury
Christchurch, NEW ZEALAND

Nicholas C. Wormald[†]
Department of Mathematics and Statistics
University of Melbourne
Parkville, VIC 3052, AUSTRALIA

[*] Research supported by the Australian Research Council

[†] Research supported by the Australian Research Council

Abstract

Given a set of trees with leaves labelled from a set L, is there a tree T with leaves labelled by L such that each of the given trees is homeomorphic to a subtree of T? This question is known to be NP-complete in general, but solvable in polynomial time if all the given trees have one label in common (equivalently, if the given trees are rooted). Here we show that this problem is NP-complete even if there are two labels x and y such that each given tree contains x or y. However, if it is known that the distance between x and y is less than 4, then the problem is solvable in polynomial time. We give an algorithm for doing this. On the other hand, we show that the question of whether a fully resolved (binary) tree exists which has *no* subtree homeomorphic to one of the given ones is NP-complete, even when the given trees are rooted. This sheds some light on the complexity of determining whether a probability assignment to trees is coherent.

Keywords: phylogenetic trees, compatibility, NP-complete, probability, reconstruction.

1 Introduction and definitions

A *phylogenetic tree* on a label set L is a tree with no vertices of degree 2 and exactly $|L|$ leaves, each of which is labelled with a distinct element of L. Such trees are used to represent evolutionary relationships in biology. A *binary (phylogenetic) tree* is one with all non-leaf vertices having degree 3.

Suppose that T is a phylogenetic tree on L and A is a subset of L. Consider the minimal subtree of T that connects leaves from A, and suppress all vertices of degree 2 (i.e. make the tree homeomorphically irreducible) to obtain a phylogenetic tree on A, denoted $T|_A$. If T' is a binary tree, we say T is *compatible* with T' if $T|_A = T'$ for some subset A of L. A set S of binary trees is said to be *consistent* if there exists a phylogenetic tree T'' that is compatible with all the trees in S. We then say T'' *realises* S.

A general problem considered in recent literature (Aho et al. [1], Ng and Wormald [5], Steel [6] and Constantinescu and Sankoff [2]) is to determine whether there exists a tree that realises S. This general problem has been shown to be NP-complete (see Steel [6]). On the other hand, it has been shown that the problem of determining whether a set of rooted trees is consistent can be solved in polynomial time (see [1]; Hensinger et al. [4] and Ng and Wormald [5] consider similar questions).

Suppose L_0 is a subset of L such that every input tree has at least one label in L_0. The question we consider in this paper is: what is the complexity of the consistency problem if $|L_0|$ is fixed, that is, independent of $n = |L|$? This question was posed by Steel [6]. If $|L_0| = 1$, then the input trees can be considered as rooted trees, and so consistency can be determined in polynomial time. We shall show that the problem is NP-complete for $|L_0| = 2$. It then follows trivially that the problem is also NP-complete for any fixed value of $|L_0| \geq 2$. The proof used in [6] for the general case does not extend to the case when $|L_0| = 2$. It is interesting to note that the proof here is simpler, even though the result is stronger.

In defining the concepts of compatibility and consistency, we have confined them to the case when all the input trees are binary, as we intend to apply them only to input quartets. In general when the input trees are phylogentic trees, two different types of compatibility can be defined. We define strong compatibility for general trees exactly as compatibility is defined for binary trees. This definition is used in [5]. We say T is *weakly compatible* with T' if T' can be obtained from $T|_A$ by contracting certain edges. This definition was used in [6]. These two definitions coincide when T' is a binary tree. We focus on binary trees in this paper, which certainly suffices to prove NP-completeness statements about general trees. Moreover, all results on the existence of a binary tree compatible with a set of input binary trees immediately give results on the existence of a general tree strongly compatible with the same input, since the two questions are equivalent (see [5]).

If we consider the case that L_0 contains just two labels, x and y, and restrict ourselves to asking for a compatible binary tree in which the distance between x and y is less than 4, this is no longer NP-complete. This is trivially true if the distance is 2 (when it becomes equivalent to the case of rooted trees). It is proved in Section 3 for distance 3 by giving a polynomial-time algorithm. It is reasonable to presume the same will hold when the roots have distance k for any fixed k, but we do not have a proof of this. On the other hand, the number 3 may be special since this is the diameter of a quartet, so it is plausible that the problem is NP-complete for $k \geq 4$. Another variation of the problem that we also show to be solvable in polynomial time is when all but a bounded number of the input trees contain a common root.

In Section 4 we consider a related problem: whether a given probabilistic distribution of subtrees can be generated in a natural way by a model of a random tree. In considering this problem, we show the NP-completess of a problem asking for the existence of a tree avoiding a given set of subtrees.

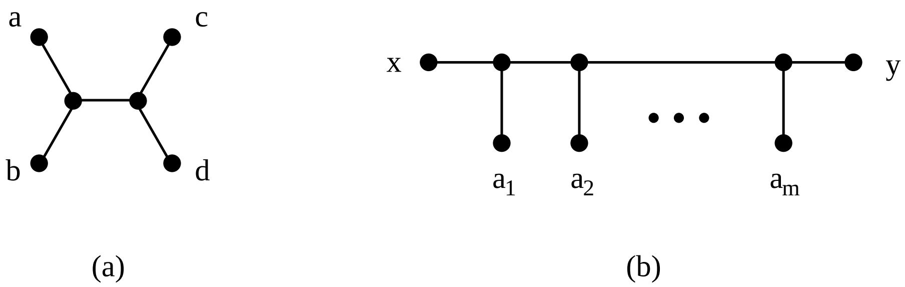

Figure 1: A quartet and a caterpillar

We finish this section with some additional definitions. A *quartet* is a binary phylogenetic tree on a label set of size 4. We denote a quartet on the label set $\{a, b, c, d\}$ by $ab|cd$, if a and b are the labels of two closest leaves, as shown in Figure 1(a). A *caterpillar* is a binary tree that has at most two vertices that are each adjacent to precisely two leaves. If x and y label two leaves that are maximally far apart on a caterpillar, we shall call it an xy-caterpillar. We write $xa_1|a_2 \dots a_{m-1}|a_m y$ to denote the xy-caterpillar shown in Figure 1(b). We note that, for each pair, x, y, there is a one-to-one correspondence between the xy-caterpillars on the label set L and linear orderings on the set $L \setminus \{x, y\}$. Note that, if T is an xy-caterpillar, and $A \subseteq L$ with $x, y \in A$, then $T|_A$ is also an xy-caterpillar.

2 Complexity of the problem for two roots

We henceforth consider the case where the input trees are all quartets. Suppose Q is a set of quartets, each of whose leaves are labelled from a set L, and for some set $L_0 \subset L$, suppose each quartet in Q has at least one label in L_0. We shall show that, if $|L_0| = 2$, then the problem of deciding whether Q is consistent is NP-complete. The instances of the problem in our proof have at least a constant times $|L|$ occurrences of each label in L_0, so restrictions of the problem which avoid this are potentially polynomial time solvable.

Two alphabets are used for leaf labels in the following lemma to emphasise their different roles in the proof of Theorem 1. In particular, we use $L_0 = \{x, y\}$. The labels a, b and c

will be used in a betweenness ordering that is central to the proof of Theorem 1, while the labels θ and ψ are extraneous labels used in quartets to ensure the betweenness ordering.

Lemma 1. *Let $Q = \{x\theta|by, xb|\psi y, xa|c\theta, yc|a\psi\}$. Then the only trees on the label set $\{x, y, a, b, c, \theta, \psi\}$ that realise Q are the two caterpillars $xa|\theta b\psi|cy$ and $c\theta|xby|a\psi$ shown in Figure 2.*

Figure 2: Two caterpillars

Proof. There is only one tree on the label set $\{x, y, b, \theta, \psi\}$ that is compatible with the quartets $x\theta|by$ and $xb|\psi y$, namely, the caterpillar $x\theta|b|\psi y$. If the leaves a and c are added to this tree in such a way as to be compatible with $xa|c\theta$, then since x and θ are adjacent leaves, either a must be added right next to x, or c next to θ. Similarly, considering $yc|a\psi$, we need c next to y or a next to ψ. Hence there are exactly two possibilities: c can be added next to y and a next to x (giving the first tree), or c next to θ and a next to ψ (giving the second). ∎

From Section 4 of [6], we have the following result.

Lemma 2. *If a set of xy-caterpillars is consistent, then there exists an xy-caterpillar that realises the set.*

The topic of this section is the following decision problem.

BI-ROOTED QUARTET CONSISTENCY

INSTANCE: A set Q of quartets each of which includes a leaf labelled by x or by y.

QUESTION: Is Q consistent?

Theorem 1. *Bi-rooted quartet consistency is NP-complete.*

Proof. The problem is clearly in NP, for, given a tree T that realises Q this consistency can be verified by checking each quartet in Q against T, and this checking can be done in polynomial time. We next describe a transformation from the following problem, which is NP-complete (Garey and Johnson [3], p. 279, problem MS1).

BETWEENNESS

INSTANCE: A finite set A and a collection I of ordered triples (a, b, c) of distinct elements from A (we may assume that each element of A occurs in at least one triple from I).

QUESTION: Is there a betweenness ordering f of A for I, that is, a one-to-one function $f : A \longrightarrow \{1, 2, \ldots, |A|\}$ such that for each $(a, b, c) \in I$, either $f(a) < f(b) < f(c)$ or $f(c) < f(b) < f(a)$?

Given an instance $I = \{(a_i, b_i, c_i); i = 1, \ldots, k\}$ of BETWEENNESS, we let a_i, b_i, c_i, θ_i and ψ_i $(i = 1, \ldots, k)$ be $5k$ labels, x and y two other labels, $Q_i = \{x\theta_i|b_iy, xb_i|\psi_iy, xa_i|c_i\theta_i, yc_i|a_i\psi_i\}$ and $Q(I) = \cup_{i=1}^{k} Q_i$. We note that each quartet in $Q(I)$ has a leaf labelled by x or by y. Clearly, the transformation can be done in polynomial time. We shall now show that $Q(I)$ is consistent if and only if I allows a betweenness ordering on the set $A = \cup_{i=1}^{k}\{a_i, b_i, c_i\}$.

Suppose that $Q(I)$ is consistent and T is a tree that realises $Q(I)$. Consider, for each $i, t_i := T|_{\{x,y,a_i,b_i,c_i\}}$. By Lemma 1, t_i is $xa_i|b_i|c_iy$ or $xc_i|b_i|a_iy$. Now the set $S = \{t_i; i = 1, \ldots, k\}$ is consistent since it is realised by T. By Lemma 2, there exists an xy-caterpillar T' which realises S. Then the order of the labels in A along T' provides the required betweenness ordering of A for I, since the label set of T' is $A \cup \{x, y\}$.

Conversely, suppose I allows a betweenness ordering on A. Let T' be one of the associated xy-caterpillars, obtained by ordering the labels in A along T' according to the betweenness ordering. We need to attach $2k$ additional labels $\{\theta_i, \psi_i; i = 1, \ldots, k\}$ to T' to obtain a tree compatible with $Q(I)$. For $i = 1, \ldots, k$, proceed as follows: If $T'|_{\{x,y,a_i,b_i,c_i\}} = xa_i|b_i|c_iy$, then attach θ_i and ψ_i to the xy-path of the tree so far constructed so that θ_i is between a_i and b_i, and ψ_i is between b_i and c_i. The resultant tree restricted to $\{x, y, a_i, b_i, c_i, \theta_i, \psi_i\}$ will be the caterpillar $xa_i|\theta_i b_i \psi_i|c_i y$. On the other hand, if $T'|_{\{x,y,a_i,b_i,c_i\}} = xc_i|b_i|a_iy$, then attach θ_i to the edge incident with the leaf labelled c_i, and attach ψ_i to the edge incident with the leaf labelled a_i. The resultant tree when restricted to $\{x, y, a_i, b_i, c_i, \theta_i, \psi_i\}$ will be the second tree specified in Lemma 1. In this way, we obtain a tree T which realises Q_i, for $i = 1, \ldots, k$ and hence realises $Q(I)$. ∎

Comments:

1. In the above proof, half of the quartets in $Q(I)$ have both labels x and y. One may ask the question: what is the complexity of the bi-rooted quartet consistency problem, if no quartet in Q has both labels x and y? The answer is that it is still NP-complete, as we can replace each $x\theta_i|b_iy$ by two quartets $x\theta_i|b_i\alpha_i$ and $b_i\theta_i|\alpha_iy$ where α_i is a new label. These quartets imply the quartet $x\theta_i|b_iy$. A similar replacement can be done for $xb_i|\psi_iy$.

2. The above theorem shows that the consistency problem is NP-complete for general (non-binary) trees for both types of compatibility described in the introduction.

3 Roots of distance 3

As mentioned in the Introduction, the problem with two roots is equivalent to the rooted tree case if the roots are prescribed to have distance 2. To answer this question, if any quartet has x and y of distance 3 then the answer is "No", and otherwise replace all labels y by x in the quartets, throw away all quartets with two x's, and solve the resulting single-root problem.

In the rest of this section we consider the analogous problem when the roots are prescribed as having distance 3.

DISTANCE 3 BI-ROOTED QUARTET CONSISTENCY

INSTANCE: A set Q of quartets each of which includes a leaf labelled by x or by y.

QUESTION: Is there a binary tree T compatible with all the trees in Q, such that the distance from x to y in T is 3?

Theorem 2. *DISTANCE 3 BI-ROOTED QUARTET CONSISTENCY can be answered correctly by a polynomial-time algorithm.*

Proof. In fact, assuming such a tree exists (call it T_0), we give an algorithm for finding a tree T (possibly different from T_0) that is also compatible with Q.

Let the x, y-path in T include the vertices x' and y', where x' is adjacent to x and y' to y. Recalling that we seek a binary tree, consider the two branches that diverge from the x, y-path at x' and at y'. The set of leaves in the branch at x' (i.e., leaves other than x which are closer to x' than to y') we denote by S_x, and the leaves in the branch at y' we denote by S_y. The problem is now broken down into two tasks:

(a) Determine sets S_x and S_y.

(b) Construct the branches of T at x' containing the leaves in S_x, and those at y' containing the leaves in S_y.

We have to perform (a) in such a way that if T_0 exists, then (b) can be performed such that the final tree T realises Q. Doing (b) is easy, since S_x can be treated by restricting the quartets in Q to those containing four labels in $S_x \cup \{x, y\}$, and solving this as a distance 2 problem as described above. The resulting tree T' (if Q is consistent) determines the branch at x'. The branch at y' is obtained in a similar fashion and inserted into T' on the edge incident with y.

It remains only to discuss how to deal with (a). For this, we can build up sets of leaf labels which must be in the same set (of S_x and S_y). Start with all leaf labels (except x and y) in separate sets, and repeatedly amalgamate the two sets containing labels p and q for any quartet in Q of the form $xt|pq$ or $yt|pq$ where t may equal x, y or any other label. After this, we have sets of labels $B_1, \ldots, B_k$ for some k. Note that all labels in the same set B_i must be in the same S_z in T_0. Next, for each quartet in Q of the form $xp|yq$, put the labels in the set B_i containing p all into S_x, and those in the set containing q into S_y. (If this causes a conflict, it may be deduced that the tree T_0 does not exist.) The labels in any set B_i which are not now in S_x or in S_y are placed into a set R of remainders. (This includes all labels not appearing in any quartets in Q.)

If T_0 exists, the leaves now in S_x must be in the branch at x', and those in S_y in the branch at y', and hence step (b) can indeed be performed as described above, to obtain a tree T'' containing all labels not in R. Finally, note that the only quartets which can contain labels in R are of the form

(i) $xy|pq$ with p, $q \in R$, or

(ii) $xp|rs$ with $p \in R$ and r, s either both in R or both not in R, or

(iii) $xt|pq$ with p, $q \in R$, or

(iv) quartets as in (ii) or (iii) with x replaced by y.

If T_0 exists, let T_1 be the subtree of T_0 induced by the leaves with labels in $R \cup \{x, y\}$. Delete y and its adjacent vertex from T_1, to obtain T_2. Then T_2 satisfies all the quartets listed above, with y replaced by x (ignoring the ones containing both x and y). So we can use the single rooted tree algorithm to find a tree T_2' compatible with the quartets listed above (with y replaced by x). Attach T_2' to the tree T'' by gluing the leaf x of T_2' to the middle of the edge of T'' in the branch at x' (thus forming a new vertex of degree 3). The resulting tree is T. It is clear that T is compatible with all the quartets listed above, and hence all quartets in Q. ∎

Comment: Another variation of the bi-rooted quartet consistency problem that is solvable in polynomial time is when all but r of the quartets in Q contain a common label, say, x. Let Q_x be the set of quartets that contain x and $Q' = Q \setminus Q_x$. The quartets in Q' need not have any labels in common. One can attach x to each quartet in Q' to obtain a set Q_x' of five-leafed binary trees. There are five ways to attach x to a quartet and 5^r ways of generating Q_x'. The consistency of Q can now be checked by running the polynomial-time algorithm for rooted trees (Aho et al. [1]) on $Q \cup Q_x'$ and let Q_x' range over all 5^r possibilities. If one of the possibilities is consistent, then Q is consistent.

4 Forbidding subtrees

In this section, we consider the complexity of constructing a tree that is not compatible with any of a given set of subtrees. It will be shown that the problem is NP-complete even for rooted trees.

We consider the following decision problem.

FORBIDDEN SUBTREES

INSTANCE: A collection S of rooted binary trees whose leaf sets are subsets of a label set L.

QUESTION: Is there a leaf-labelled rooted binary tree T with label set L having no subtree containing the root homeomorphic to a tree in S?

Our main result in this section is the following.

Theorem 3. *The decision problem FORBIDDEN SUBTREES is NP-complete.*

Comments:

1. If we drop the word "binary" from both the instance and the question, then the resulting problem is still NP-complete, by polynomial transformation from FORBIDDEN SUBTREES. This is because the output tree in FORBIDDEN SUBTREES can be forced to be binary by including appropriate trees in the input which forbid all vertices of degree at least 4.

2. A more general problem can be formulated as follows. Suppose we are given a function $f : S \to [0,1]$. Then f may be considered as a measure of "confidence" or "probability" of the subtrees in S. We wish to know whether f "lifts" to a probability distribution on the set $R(L)$ of all rooted binary trees with leaf set L. That is, is there a function $\hat{f} : R(L) \to [0,1]$ with $\sum_{T \in R(L)} \hat{f}(T) = 1$ and such that, for all $t \in S$, $f(t)$ is the sum of $\hat{f}(T)$ over all T in $R(L)$ that are compatible with t. If $\hat{f}$ exists, then our beliefs represented by f are "coherent"; otherwise, not. The special case that $f(t) = 0$ for all $t \in S$ has answer yes if and only if FORBIDDEN SUBTREES has answer yes and is therefore NP-hard. On the other hand, the special case that $f(t) = 1$ for all $t \in S$ has answer yes if and only if S is consistent, so there is a polynomial time algorithm for this special case (Aho et al. [1], Ng and Wormald [5], Henzinger et al. [4]).

Proof of Theorem 3 Consider an instance I of BETWEENNESS and let A be the set of all labels i, j, k with $(i, j, k) \in I$. Let $L = A \cup \{z\}$ where $z \notin A$, and construct S as the union of the following four sets. Here $a(bc)$ denotes the rooted tree with b and c on one branch at the root and a on the other, and $a(b(cd))$ denotes the rooted tree with $b(cd)$ on one branch at the root and a on the other.

$$\begin{aligned}
S_1 &= \{z(xy) : x, y \in A\} \\
S_2 &= \{j(ik) : (i, j, k) \in I\} \\
S_3 &= \{i(k(jz)) : (i, j, k) \in I\} \\
S_4 &= \{k(i(jz)) : (i, j, k) \in I\}
\end{aligned}$$

Suppose that there is a leaf-labelled tree T as required in FORBIDDEN SUBTREES. Then the absence of the subtrees in S_1 forces T to be a caterpillar, with the leaves having labels from A to be attached along the path from the root to z in some linear order. Next, forbidding the rest of S forces the ordering to be a betweenness ordering for I. Conversely, if I has a betweenness ordering, then that ordering gives a permissible ordering of those leaves along the path from the root to z. The transformation implicitly described here takes polynomial time, and therefore FORBIDDEN SUBTREES is NP-complete. ∎

Open problem

In view of the second comment after Theorem 3, we ask for the complexity of the following problem, where c denotes a pre-chosen "confidence level", $0 < c \leq 1$:

c-EXPECTED SUBTREES

INSTANCE: A collection S of rooted binary trees whose leaf sets are subsets of a label set L and a function $f : S \to [0, 1]$ with $f(t) \geq c$ for all $t \in S$.

QUESTION: Does f "lift" to a probability distribution on the set $R(L)$ of all rooted trees labelled from L?

Of course, for $c = 1$ this has a polynomial time algorithm.

Acknowledgement The authors are grateful to the referees for their very helpful comments.

References

[1] A.V. Aho, Y. Sagiv, T.G. Szymanski and J.D. Ullman, Inferring a tree from lowest common ancestors with an application to the optimization of relational expressions, SIAM J. Comput. 10 (1981) 405–421.

[2] M. Constantinescu and D. Sankoff, An efficient algorithm for supertrees, Journal of Classification 12 (1995) 110–112.

[3] M. R. Garey and D. S. Johnson, Computers and Intractability, Bell Telephone Laboraties, Ltd, New Jersey, 1979.

[4] M.R. Henzinger, V. King and T. Warnow, Constructing a tree from homeomorphic subtrees, with applications to computational molecular biology, Proceedings ACM/SIAM Symposium on Discrete Algorithms (1996) 330-340.

[5] M.P. Ng and N.C. Wormald, Reconstruction of rooted trees from subtrees, Discrete Applied Math. 69 (1996) 19–31.

[6] M. Steel, The complexity of reconstructing trees from qualitative characters and subtrees, Journal of Classification 9 (1992) 91–116.