Optimal agreement supertrees

David Bryant

LIRMM, 161 rue Ada, 34392 Montpellier, France Cedex 5

Abstract

An *agreement supertree* of a collection of unrooted phylogenetic trees $\{T_1, T_2, \ldots, T_k\}$ with leaf sets $\mathcal{L}(T_1)$, $\mathcal{L}(T_2), \ldots, \mathcal{L}(T_k)$ is an unrooted tree T with leaf set $\mathcal{L}(T_1) \cup \cdots \cup \mathcal{L}(T_k)$ such that each tree T_i is an induced subtree of T. In some cases, there may be no possible agreement supertrees of a set of trees, in other cases there may be exponentially many. We present polynomial time algorithms for computing an optimal agreement supertree, if one exists, of a bounded number of binary trees. The criteria of optimality can be one of four standard phylogenetic criteria: binary character compatibility; maximum summed quartet weight; ordinary least squares; and minimum evolution. The techniques can be used to search an exponentially large number of trees in polynomial time.

1 Introduction

The general phylogenetic supertree problem is how to combine phylogenies for different sets of species into one large "super"-phylogeny that classifies all of the species. The problem is motivated by physical, computational, and statistical constraints on the construction of large scale phylogenies.

In this paper we present a supertree method for combining a bounded number of binary, unrooted, phylogenetic trees. We search for supertrees such that every input tree is a restriction of the supertree to a subset of the set of species. When multiple supertrees exist we choose the supertree that is optimal with respect to one of four standard phylogenetic optimization criteria. Even though the number of possible supertrees can be exponentially large, we can determine an optimal supertree in polynomial time.

1.1 Agreement supertrees

We begin with a few definitions. An **unrooted phylogenetic tree** T is a finite, acyclic connected graph with no internal vertices of degree two and degree one vertices (leaves) labelled uniquely by members of a finite leaf set $\mathcal{L}(T)$. A tree T' is an **induced subtree** of T if T' can be obtained from T by deleting vertices and edges and supressing vertices of degree two. Intuitively T' represents the restriction of T to a subset of the leaf set. If T' is an induced subtree of T with leaf set $L' \subseteq \mathcal{L}(T)$ then we write $T' = T|_{L'}$. An **agreement supertree** of a collection of trees $\{T_1, T_2, \ldots, T_k\}$ with leaf sets $\mathcal{L}(T_1), \mathcal{L}(T_2), \ldots, \mathcal{L}(T_k)$ is an unrooted tree T with leaf set $\mathcal{L}(T_1) \cup \cdots \cup \mathcal{L}(T_k)$ such that each tree T_i is an induced subtree of T.

The agreement supertree is the dual of the **agreement subtree**, recalling that S is an agreement subtree of trees $T_1, \ldots, T_k$ if S is an induced subtree of each of $T_1, \ldots, T_k$ (c.f [3, 6]). A collection of trees $T_1, \ldots, T_k$ on leaf set L are agreement supertrees of a collection of trees $S_1, \ldots, S_l$ on subsets of L if and only if $S_1, \ldots, S_l$ are agreement subtrees of $T_1, \ldots, T_k$, and every leaf in L appears in at least one S_i.

There can be exponentially many agreement supertrees for a collection of trees $\mathcal{T}$, even when the number of trees in $\mathcal{T}$ is bounded [4]. When there is more than one agreement supertree we select one that is optimal according to one of four standard phylogenetic optimization criteria. The choice of the four optimization criteria is determined by algorithmic constraints - we discuss the possible extension to other criteria in section 6. Note that the optimization criteria require *global* information for the entire set of taxa present in the input trees. Hence the algorithms are better tailored to divide and conquer applications, rather than the assembly of different data sets.

Though the number of agreement supertrees can be exponentially large, the algorithms determine an optimal agreement supertree in polynomial time, provided that the number of input trees is bounded. The agreement supertree methods presented here are practical for a small number of large input trees, rather than a large number of small input trees.

Finally we note that agreement supertree methods do not handle conflict in the set of input trees. If two input trees resolve the relationship between species in a different way then there will be no agreement supertrees for the collection. There are numerous possible *ad hoc* solutions to this shortcoming (e.g. removing leaves involved in conflicts, modifying input trees). However the more general problem of how to systematically resolve conflict in supertrees is still hotly debated within phylogenetics. At this early stage we present our results as tools to be incorporated into practical supertree methods once some of the fundamental systematic questions have been answered.

Despite the false modesty, the algorithmic results presented in this paper have several direct applications, a few of which we discuss in section 5. In particular we stress the ability to search over an exponentially large number of trees in polynomial time.

2 Four standard optimization criteria

In this section we describe the phylogenetic criteria used for optimization. Discussion of the motivations, use, and characteristics of the various criteria can be found, for example, in [10]. Here we give only sufficient detail to describe the problems.

2.1 Binary character weights

Given a phylogenetic tree T, removing an edge of T induces a bipartition of the leaf set of T, called a **split** of T. The set of all splits of T is denoted $splits(T)$, and a single split is written as $A|B$, using the standard notation for partitions.

A split can be viewed as a **binary character** where each leaf is assigned a value of 0 or 1 depending on which block of the split it is in. Given a weighted collection of binary characters we can score a tree T by summing the weight of the characters corresponding to splits in T. The total summed weight is called the **binary character compatibility score** of T. We wish to find T that maximizes the binary character compatibility score.

We will assume that all binary characters have non-negative weights.

2.2 Quartet criteria

A (resolved) **quartet** is an unrooted binary tree on four leaves. There are three possible quartets for each set of four leaves. For any tree T the **quartet set** of T, denoted $q(T)$, is the set of all quartets that are induced subtrees of T.

Suppose that we have assigned a weight $w(ab|cd)$ for every possible quartet $ab|cd$ such that $\{a, b, c, d\} \subseteq \mathcal{L}(T)$. The **quartet score** of a tree T is the sum of the weights $w(ab|cd)$ over all $ab|cd \in q(T)$. We want to find a tree with maximum quartet score.

We will assume that all quartets have non-negative weights.

2.3 Least squares (OLS)

Suppose that T is an unrooted tree and the edges of T are given real valued weights. The **leaf to leaf distance** between any two leaves of T is the sum of the edge weights along the path between the two leaves. In this way we can construct a distance matrix p containing all of the leaf to leaf distances between leaves in T. The **sum of squares** distance between p and a given distance matrix d for $\mathcal{L}(T)$ is defined

$$\|p - d\|^2 = \sum_{x \in \mathcal{L}(T)} \sum_{y \in \mathcal{L}(T)} (d_{xy} - p_{xy})^2.$$

For a fixed tree T the edge weights for T that give a distance matrix p minimizing $\|p - d\|^2$ are called the **OLS edge length estimates** for T.

The **ordinary least squares (OLS) score** of a tree T is the value $\|p - d\|^2$ given by the OLS edge length estimates. We want to find a tree T with minimum OLS score.

2.4 Minimum evolution (ME)

The minimum evolution criteria is closely related to OLS. Given an unrooted tree T and a distance matrix d we first calculate the OLS edge estimates for T from d. The **minimum evolution (ME) score** for T is then the sum of these edge weights. The minimum evolution score is usually only evaluated for binary trees[1]. We want to find a tree with minimum ME score.

3 Split constrained phylogenetic optimization

In this section we describe the results of [1, 2, 5, 7] that provide the machinery for the optimal agreement supertree algorithms. The results all follow the same theme: the introduction of split based constraints that make phylogenetic reconstruction polynomial time solvable. More formally we have:

INSTANCE: Set $\mathcal{S}$ of splits on leaf set L. Degree bound d.
PROBLEM: Is there a tree T with degree bound d such that $splits(T) \subseteq \mathcal{S}$? If so, find the tree(s) with degree bound d and $splits(T) \subseteq \mathcal{S}$ with

1. maximum binary compatibility score (with respect to a given weighting of splits in $\mathcal{S}$). [1]
2. maximum quartet score (for a given quartet weighting). [7]
3. minimum OLS score (for a given distance matrix). [5]
4. minimum ME score (for a given distance matrix). [2, 5]

All of these versions of the problem can be solved in polynomial time (in the number of splits and leaves), when d is bounded by a constant.

4 Optimal agreement supertree algorithms

4.1 Split set constraints

We show how the split constrained optimization algorithms can be used to construct optimal agreement supertrees.

Let $\mathcal{T} = \{T_1, \ldots, T_k\}$ be a collection of trees. Put $L_i = \mathcal{L}(T_i)$ for each i and $L = L_1 \cup \cdots \cup L_k$. Define

$$\mathcal{S}(\mathcal{T}) = \left\{ A_1 \cup \cdots \cup A_k | B_1 \cup \cdots B_k : \begin{array}{l} A_i|B_i \in splits(T_i) \cup \{\emptyset|L_i\}, i = 1, 2, \ldots, k \\ (A_1 \cup \cdots \cup A_k) \cap (B_1 \cup \cdots B_k) = \emptyset \end{array} \right\}$$

Note that we will assume $A_i|B_i \in splits(T_i)$ implies $B_i|A_i \in splits(T_i)$. Put $n = |L|$. There are at most $2n-3$ splits in each tree so $|\mathcal{S}(\mathcal{T})|$ is $O(2^k n^k)$. Furthermore

Theorem 1 *Let T be an unrooted phylogenetic tree with leaf set L. If each tree $T_i \in \mathcal{T}$ is an induced subtree of T then $splits(T) \subseteq \mathcal{S}(\mathcal{T})$.*

Proof
Suppose that each tree $T_i \in \mathcal{T}$ is an induced subtree of T. Thus $T_i = T|_{L_i}$ and

$$\begin{aligned} splits(T_i) &= splits(T|_{L_i}) && (1) \\ &= \{A \cap L_i | B \cap L_i : A|B \in splits(T)\} - \{L_i|\emptyset\}. && (2) \end{aligned}$$

Hence for each $A|B \in splits(T)$ we have

$$A \cap L_i | B \cap L_i \in splits(T_i) \cup \{L_i|\emptyset\}.$$

It follows that

$$A|B = (A \cap L_1) \cup \cdots \cup (A \cap L_k) | (B \cap L_1) \cup \cdots \cup (B \cap L_k)$$

[1]It would be interesting to determine whether or not optimal minimum evolution trees can be assumed to be binary

and $A|B$ belongs to $\mathcal{S}(\mathcal{T})$. □

We now take advantage of the fact that the input trees are all binary.

Lemma 2 *If $\mathcal{T}$ contains only binary trees and there is an agreement supertree T of $\mathcal{T}$ then there is a binary agreement supertree T' of $\mathcal{T}$ such that $splits(T) \subseteq splits(T')$.*

Proof
Suppose that T is an agreement supertree of $\mathcal{T}$ that is not binary. Clearly there exists a binary tree T' such that $splits(T) \subseteq splits(T')$. We will show that T' is also an agreement supertree for $\mathcal{T}$.

For each $T_i \in \mathcal{T}$ we have $T_i = T|_{L_i}$ and, since $splits(T) \subseteq splits(T')$, we also have $splits(T|_{L_i}) \subseteq splits(T'|_{L_i})$. The tree T_i is binary, so $splits(T_i) = splits(T|_{L_i}) \subseteq splits(T'|_{L_i})$ implies $T_i = T'|_{L_i}$, completing the proof. □

We now have all the machinery needed for the main result.

Theorem 3 *Let $\mathcal{T} = \{T_1, \ldots, T_k\}$ be a collection of binary trees with leaf sets $L_1, \ldots, L_k$. We can determine whether an agreement supertree of $\mathcal{T}$ exists in $O(4^k n^{2k+1})$ time. If there is an agreement supertree for $\mathcal{T}$ then we can find the agreement supertree with optimum*

1. *binary character compatibility score (with respect to a given weighting of splits) in $O(4^k n^{2k+1})$ time.*
2. *summed quartet weight (for a given quartet weighting) in $O(2^k n^{k+4} + 4^k n^{2k+1})$ time.*
3. *least squares score (for a given distance matrix) in $O(8^k n^{3k})$ time.*
4. *ME score (for a given distance matrix) in $O(8^k n^{3k})$ time.*

Proof
By Theorem 1 we have that any agreement supertree T for $\mathcal{T}$ satisfies $splits(T) \subseteq \mathcal{S}(\mathcal{T})$. With all of the optimization criteria (except perhaps ME, which is usually defined only for binary trees) a tree T' with $splits(T) \subseteq splits(T')$ will have at least as good score as T. Hence, using Lemma 2, we can assume that the optimal agreement supertrees are binary.

The problem therefore reduces to finding an optimal binary tree contained within a given collection of splits. Applying the methods stated in section 3 is is possible to obtain the given complexities. □

We note that, when the number of trees k is bounded by a constant, we have *polynomial time* algorithms for determining optimal agreement supertrees. When k is unbounded, the problem is NP-complete (by a reduction from QUARTET COMPATIBILITY [9]).

5 Applications to optimization

To illustrate how these methods may be used during optimization we describe two applications.

5.1 Divide and conquer

Given two (or more) binary trees we now have a method for combining them in a way that optimizes phylogenetic criteria. The most obvious application of these algorithms is as the basis of a divide and conquer algorithm.

There are several possibilities for ways to subdivide the data set. The DCM method [8] uses distance data to divide the sequences into closely related clusters. However the merge algorithms we describe here will work well even if all of the input trees contain leaves that are widely scattered throughout the combined tree. For this reason, the subdivision process appears less critical. We propose the use of a random subdivision, biased to produce subsets that have close to equal sizes.

5.2 Testing stability

The standard technique for phylogenetic optimization is to conduct a tree search: an initial tree is constructed; Neighbouring trees are examined, where 'neighbouring' means within one branch swap or NNI exchange of the original tree; If a better tree is found, this is chosen as the next tree and the search continues. At each step at most $O(n)$ or $O(n^2)$ trees are searched.

We can use the merge algorithms presented in this paper to search an exponentially large number of trees without a dramatic increase in complexity. We randomly divide the set of leaves into two parts, calculate the two induced subtrees, and then calculate an optimal agreement supertree for these two trees. An agreement supertree will always exist since the leaf sets of the subtrees are disjoint. If we have found a global optimum then this procedure will always produce the original tree. If it finds a better tree we take this new tree and repeat the process.

6 Discussion

In this abstract we have outlined a method for constructing an optimal agreement supertree of a set of binary trees, if such an supertree exists. The algorithm can search an exponentially large collection of trees (the possible agreement supertrees) in polynomial time. Our results lead to several directions of further research:

6.1 Extension to other criteria

It would be useful to determine whether the algorithms can be extended to optimize over other phylogenetic criteria such as parsimony score or likelihood. In both cases we are pessimistic. With parsimony, the structure of one part of a tree can alter how best to resolve the structure of another part of the tree, making parsimony less suitable for dynamic programming algorithms such as those we use here. With maximum likelihood it is unknown whether the likelihood of a *single* tree can be computed in polynomial time! In both cases, the above results can be used to make a rough search of the tree space that would precede a more detailed, and computationally expensive, local search.

6.2 Extension to non-binary trees

The next direction of future research would be to determine if the algorithms can be extended to handle non-binary trees. The problem is that we require a degree bound in order to use the algorithms outlined in section 3. We suspect that an appropriate modification of the algorithms will give a polynomial time algorithm (for a bounded number of trees) even without a degree bound. We have derived the required modifications for the special case when the leaf sets of the input trees are disjoint, but the the complexity of the general problem remains open.

6.3 Trees with no agreement supertree

At the moment the algorithms will simply terminate if there is no agreement supertree possible. This occurs when, for example, there are conflicts between the input trees. It would be interesting to extend the algorithm to derive a consensus supertree method capable of handling input trees with conflicts. First, however, we must address the question of how to handle conflict between different trees. The subtree merger algorithm of [8] simply contracts edges causing problems. This will, in general, lead to poorly resolved trees. Once suitable criteria for combining subtrees are defined, it should be possible to extend the optimal agreement supertree algorithms above to give a general optimal subtree merger technique.

Acknowledgements

We wish to thank an anonymous referee for their helpful comments and suggestions. The paper was written while the author was a postdoctoral fellow based in the laboratory of Olivier Gascuel.

E-mail: `bryant@lirmm.fr`

References

[1] Bryant, D. (1996): Hunting for trees in binary character sets. *Journal of Computational Biology* **3**(2), 275-288.

[2] Bryant, D. (1997): *Hunting for trees, building trees and comparing trees: theory and method in phylogenetic analysis.* Ph.D. thesis, Dept. Mathematics, University of Canterbury.

[3] Finden, C.R. and Gordon. A.D. (1986) Obtaining common pruned trees. *Journal of Classification*, 2:255–276.

[4] Gordon, A.D. (1986) Consensus supertrees: the synthesis of rooted trees containing overlapping sets of leaves. *Journal of Classification*, 3:335–348.

[5] Bryant, D. (2000): Fast algorithms for constructing optimal trees with respect to OLS and minimum evolution criteria. In preparation.

[6] Goddard, W., Kubicka, E., Kubicki, G., and McMorris, F.R.. (1994). The agreement metric for labelled binary trees. *Mathematical Biosciences*, 123:215–226.

[7] Bryant, D. and Steel, M. (1999): Fast algorithms for constructing optimal trees from quartets. *Proc. ACM-SIAM Symposium on Discrete Algorithms* 10, 147-155. An extended version has been accepted for publication in *Journal of Algorithms.*

[8] D.H. Huson, S. Nettles, and T. Warnow. (1999): Obtaining highly accurate topology estimates of evolutionary trees from very short sequences. *Proc. 3rd Annual Int. Conf. Comp. Mol. Biol. (RECOMB)*, pp. 198–209.

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

[10] Swofford, D. L., G. J. Olsen, P. J. Waddell and D. M. Hillis. (1996): Phylogenetic Inference. Pp. 407–514 *in* D. M. Hillis, C. Moritz, and B. K. Mable, eds. Molecular Systematics, second edition. Sinauer, Sunderland, Mass.