DIMACS Series in Discrete Mathematics
and Theoretical Computer Science

A Classification of Consensus Methods for Phylogenetics

David Bryant

Abstract. A consensus tree method takes a collection of phylogenetic trees and outputs a single "representative" tree. The first consensus method was proposed by Adams in 1972. Since then a large variety of different methods have been developed, and there has been considerable debate over how they should be used.

This paper has two goals. First, we survey the main consensus tree methods used in phylogenetics. Second, we explore, pretty exhaustively, the links between the different methods, producing a classification of consensus tree methods.

1. Introduction

In a 1972 paper Edward N. Adams III presented "a new problem in the science of classification... along with its solution" [**1**]. The problem Adams discussed was how to combine information from rival trees into one representative tree. Adam's solution was the tree construction method that has since become known as the Adams consensus tree.

Adams' tree is now only one of many available consensus tree methods. My goal in this paper is to both survey and classify consensus methods for trees, not only reviewing the different methods but exploring connections between them.

1.1. Consensus methods and consensus trees. A consensus method summarises a collection of trees without boiling everything down to one measure or number. Fundamentally, it is nothing more than a function or algorithm that takes a collection of trees (on the same set of taxa) and returns a single tree (on the same set of taxa). Most methods identify common substructures in the input trees and represent these in the output tree. Hence, by exclusion, they also identify areas of conflict in the input trees.

There has been considerable debate over the use, or apparent abuse, of consensus tree methods. I would argue that the problem has not been so much in the decision to use consensus methods, but in the way that the consensus trees have been interpreted. For some, a consensus method is not merely a tool for representation but a tool for new phylogenetic inferences. This is problematic: the invention

1991 *Mathematics Subject Classification.* Primary 05C05, 92B10; Secondary 91B12, 92D15.
Key words and phrases. Consensus trees, phylogenetics, graph theory, parsimony.
This research was supported in part by NSERC grant #201888-00155-2000.

of most consensus tools has been guided by combinatorial properties, rather than phylogenetic inference criteria.

Nevertheless, consensus methods are a useful tool for phylogenetic inference, but only when used in conjuction with a model or paradigm. For example, Nelson [**32**] and Swofford [**42**] both observed that independently derived trees are unlikely to have clades (or clusters) in common. Thus, if we accept some basic distribution on trees, the clades appearing in most or all the input trees can be considered reliable.

Another example might be the use of consensus tree methods to help find improved trees within the parsimony paradigm. The intepretation and use of consensus trees is guided by an objective criterion (tree length).

More controversial is the combination of analyses from different data sets. Barrett *et al.*. [**4**] criticize consensus methods because the combined tree strict consensus returns is not the most parsimonious. However if the goal was just to optimize total parsimony score, one would be best to take several parsimonious, or almost parsimonious, trees from each data set and use a consensus method designed to optimize parsimony. If the goal is to represent conflict or agreement between the two data sets, then a standard consensus method is well suited.

1.2. The variety of consensus methods. What cannot be denied is that consensus trees have potential uses in a variety of different contexts, and different methods are better suited for different problems. There is now a considerable number of methods to choose from, and little detailed comparative work.

Hence the motivation for this survey and classification. My approach is descriptive rather than prescriptive: I survey the methods and discuss, relatively exhaustively, the connections between them. Section 2 describes the various consensus methods. The connections between the methods are summarised in figure 2 which is, in a literal sense, a classification of consensus methods. Proofs are relegated to the appendix.

I conclude the paper with a brief discussion on two variants of consensus tree methods: consensus subtrees and consensus supertrees. The later is particularly topical, given the current interest in constructing huge phylogenies. Supertrees are in essence not more than generalised consensus trees. Perhaps it would be judicious to reach a satisfactory consensus on the use of consensus trees before tackling their generalisation.

1.3. Terminology. The entities being classified are called *taxa*. A *group* is a subset of the set of taxa. A *rooted (phylogenetic) tree* is a rooted tree which has every leaf identified with a unique taxon and every node that is not a leaf has at least two children. A group is *monophyletic* on a tree if and only if it contains all the descendents of its most recent common ancestor. The monophyletic groups of a tree T are called *clusters* of T.

A collection of groups $\mathcal{C}$ is *compatible* if there is a rooted tree T such that every group in $\mathcal{C}$ is a cluster of T. Note that a collection of clusters $\mathcal{C}$ is compatible if and only if for each pair of clusters A and B in $\mathcal{C}$ either A is contained in B, or B is contained in A, or A and B are disjoint. We say that a cluster A is compatible with a tree T if it is compatible with every cluster of T.

A rooted tree T *refines* another rooted tree T' on the same set of taxa if every cluster of T' is a cluster of T. A rooted tree T is *binary* if every node that is not a leaf has exactly two children. A binary tree cannot be refined any further.

A *rooted triple* $ab|c$ denotes a grouping of a and b relative to c. We say that $ab|c$ is a rooted triple of T if the most recent common ancestor of a and b is a proper descendent of the most recent common ancestor of a, b, c. The set of all rooted triples of T is denote $r(T)$. Rooted triples are also called 3-taxon statements [**45**].

If X is a subset of the set of taxa and T is a rooted tree then $T|_X$ is the *restriction of* T *to* X. It is formed by replacing each cluster A of T with the intersection $A \cap X$, discarding empty clusters. The restriction of a collection $\mathcal{T} = \{T_1, \ldots, T_k\}$ of rooted trees to X equals $\{T_1|_X, T_2|_X, \ldots, T_k|_X\}$ and is denoted $\mathcal{T}|_X$.

An *unrooted (phylogenetic) tree* is a tree without a root. Every leaf is identified with unique taxon. Removing a branch (edge) of an unrooted tree divides the tree into two connected parts. If A is the group of taxa on one side of the branch and B is the group of taxa on the other, then $A|B$ is said to be the *split* corresponding to that branch. Splits are the unrooted equivalent of clusters.

A collection of splits $\mathcal{S}$ is *compatible* if there is an unrooted tree T such that every split in $\mathcal{S}$ is a split of T. One can show that $\mathcal{S}$ is compatible if and only if for every pair $A|B, C|D$ of splits in $\mathcal{S}$, at least one of $A \cap C$, $A \cap D$, $B \cap C$, $B \cap D$ is empty [**13, 31**]. We say that a split $A|B$ is compatible with a tree T if it is compatible with every split of T.

An unrooted tree T *refines* another unrooted tree T' on the same set of taxa if every split of T' is a split of T. An unrooted tree T is *binary* if every node that is not a leaf is adjacent to exactly three other nodes. A binary unrooted tree cannot be refined further.

A quartet $ab|cd$ indicates a separation of a and b from c and d. We say that $ab|cd$ is a quartet of an unrooted tree T if there is a split in T having a and b on one side and c and d on the other. The set of quartets of a tree is denoted $q(T)$.

A *weighted unrooted (rooted) tree* is an unrooted (rooted) tree with strictly positive branch lengths. Branch lengths can represent an estimate of the amount of change (e.g. number of mutations), or a level of confidence (e.g. bootstrap score). We define the *weight of a split* in a tree to be the length of the corresponding branch.

We will use the standard parenthesis (NEWICK) representation for trees in the examples. The non-singleton clusters correspond to nested matching parentheses so, for example, $((a, b), (c, d))$ denotes the rooted tree with clusters $\{a\}, \{b\}, \{c\}, \{d\}$, $\{a, b\}, \{c, d\}, \{a, b, c, d\}$. We use the same notation for unrooted trees, simply ignoring the position of the root.

2. A survey of consensus methods

2.1. Consensus methods based on splits and clusters.

2.1.1. *Strict consensus tree.* Perhaps the simplest of the all consensus methods is the strict consensus tree [**30**]. Given a collection of unrooted trees, the strict consensus tree contains exactly those splits common to all the trees in the collection. When the collection consists of rooted trees the strict consensus tree contains those clusters common to all the input trees.

EXAMPLE 2.1. Let $\mathcal{T}$ be the collection of rooted trees

$$\big\{((a,(b,c)),d),(((a,b),c),d)\big\}.$$

The clusters $\{a,b,c,d\}$ and $\{a,b,c\}$ appear in both trees, so the strict consenus tree is $((a,b,c),d)$.

Strict consensus trees have a natural generalisation to weighted trees. We compute the strict consensus as for an unweighted tree, and then assign to each branch the minimum weight of each of corresponding splits (clusters) in each of the input trees.

2.1.2. *Majority rule tree.* The majority rule tree contains exactly those clusters or splits that appear in more than half of the input trees [**5, 29, 30, 42**]. Thus every cluster (split) of the strict consensus tree will also be a cluster (split) of the majority rule tree, and the majority rule tree *refines* the strict consensus tree.

EXAMPLE 2.2. Let $\mathcal{T}$ be the collection of three rooted trees

$$\big\{((a,(b,c)),d),(((a,b),c),d),(((a,b),d),c)\big\}.$$

The clusters $\{a,b\}$, $\{a,b,c\}$ and $\{a,b,c,d\}$ appear in two out of three trees, so the majority rule tree is $(((a,b),c),d)$. Note that the strict consensus tree for this collection is (a,b,c,d).

EXAMPLE 2.3. Let T_1 and T_2 be the unrooted trees $((a,b,c),(d,e,f))$ and $((a,d),(b,e),(c,f))$. If $\mathcal{T}$ contains three copies of T_1 and two copies of T_2 then the majority rule tree of $\mathcal{T}$ equals T_1.

Barthélemy and McMorris [**5**] showed that the the majority rule tree is also a median tree. Given two trees T_1 and T_2 the *symmetric difference distance* $d(T_1,T_2)$ between T_1 and T_2 is the number of splits (clusters) appearing in one tree but not the other. The majority rule tree for $\mathcal{T}=\{T_1,T_2,\ldots,T_k\}$ minimizes

$$d(T,\mathcal{T})=\sum_{i=1}^{k} d(T,T_i) \tag{2.1}$$

making the majority rule a *median* of $\mathcal{T}$ with respect to the symmetric distance metric.

The majority rule tree can be generalised by assigning a weight to each tree and then taking the splits which appear in trees summing up to over half the total weight. For example, if the total weight of trees was 10 and the split $A|B$ appear in trees of weight 4 and 2 then $A|B$ would be in the majority rule tree. Jermiin *et al.* [**25**] propose weighting schemes based on likelihood scores.

2.1.3. *Loose consensus tree.* The loose consensus tree was originally called the combinable component tree or semi-strict consensus tree [**10, 42**]. We take its present name from [**6**]. The loose consensus tree for a collection of rooted trees $\mathcal{T}$ contains exactly those clusters that are compatible with every tree in $\mathcal{T}$. Similarly, the loose consensus of a collection of unrooted trees $\mathcal{T}$ contains exactly those splits that are compatible with every unrooted tree in $\mathcal{T}$. It follows that the loose consensus tree refines the strict consensus tree.

EXAMPLE 2.4. Let $\mathcal{T}$ be the collection of rooted trees $\{((a,b),(c,d)),((a,b,c),d)\}$. The cluster $\{a,b\}$ is compatible with both trees, however the cluster $\{c,d\}$ is not compatible with the cluster $\{a,b,c\}$. Hence the loose consensus tree for $\mathcal{T}$ is $((a,b),c,d)$. The strict consensus tree for this collection equals (a,b,c,d).

Note that if T is a binary unrooted tree and $A|B$ is a split compatible with T then $A|B$ must be a split of T, since T cannot be refined further. Hence if all the trees in the input collection $\mathcal{T}$ are binary then the loose consensus tree will be the same as the strict consensus tree. The same applies for the rooted case.

Queiroz [**16**] observed that the loose consensus tree can contain splits or clusters that appear in only one of the input trees. This was identified as a shortcoming when combining data because the splits or clusters appearing in more than one tree are likely to be reliable while those appearing in only one tree may or may not be. However the clusters and splits in the loose consensus tree are all *compatible* with all of the trees. It is unlikely for a cluster to be compatible with a random tree, so we can have some level of confidence in the reliability of these clusters.

2.1.4. *Greedy consensus tree.* Both PHYLIP [**18**] and PAUP [**43**] allow additional splits or clusters to be included in the majority rule tree. The splits (clusters) are selected using what is called a *greedy strategy*.

Suppose that $\mathcal{T}$ is a collection of unrooted trees. We write out the set of splits appearing in trees in $\mathcal{T}$ in order of frequency, with those splits appearing in the largest number of trees coming first in the order and ties broken arbitrarily. A collection of compatible splits $\mathcal{S}$ is built up step by step. The first split in the ordering is put in $\mathcal{S}$. The remaining splits are considered in order: if a split is compatible with all of the splits already in $\mathcal{S}$ then it is included in $\mathcal{S}$. At the end of the process we will obtain a collection of splits corresponding to some phylogenetic tree. This gives the greedy consensus tree for $\mathcal{T}$. The same process works for clusters.

One problem with the greedy approach is that if two clusters or splits appear the same number of times, then they could be put in either order: this decision made arbitrarily by the program. For example, in PHYLIP, the consensus of rooted trees $((a,b),c)$ and $((a,c),b))$ is either $((a,b),c)$ or $((a,c),b)$, depending on the order of the trees in the input file.

EXAMPLE 2.5. Let $\mathcal{T}$ be the collection of three rooted trees $((a,b,c),((d,e),f))$, $(((a,b,c),f),(d,e))$ and $(((a,b),c),(d,(e,f)))$. Then $\{a,b,c,d,e,f\}$ and $\{a,b,c\}$ appear in three trees, $\{d,e,f\}$ and $\{d,e\}$ appear in two trees, while $\{a,b,c,f\}$, $\{a,b\}$ and $\{e,f\}$ appear in only one tree. The greedy consensus tree for these trees is $(((a,b),c),((d,e),f))$. The strict consensus tree is $((a,b,c),d,e,f)$ and the majority rule tree is $((a,b,c),((d,e),f))$.

Every greedy consensus tree for a collection refines both the majority rule and strict consensus trees.

THEOREM 2.6. *The greedy selection method produces a consensus tree which refines both the majority rule tree and the loose consensus tree.*

2.1.5. *Nelson-Page and asymmetric median consensus trees.* The original definition of the Nelson consensus tree [**32**] was founded on the premise that one cladogram can be wrong, but not two cladograms. The assumption is that clusters

appearing in two or more trees are highly likely to be genuine. These clusters are called *replicated clusters*, or *replicated components*, and the Nelson consensus tree is defined to be the tree containing the replicated clusters together with the remaining clusters that are compatible with all the replicated clusters.

There are problems with this definition. The set of replicated clusters might not be compatible, so the method does not always give a tree [**30, 35, 10**]. As well, there may be several distinct groups of non-replicated clusters that are compatible with the replicated clusters and with themselves, and Nelson gives no indication as to how this indeterminacy is to be resolved. As Nelson remarked: "The pitfalls of philosophy are many, and some of them are deep." [**32**]

Page [**35**] goes some way towards addressing these shortcomings. Each cluster in the original trees is assigned a weight equal to one less than the number of times that cluster appears in the input collection. The *Nelson-Page Consensus Tree* is then taken to be the maximum weight compatible subset of this collection of clusters. If there is more than one maximum weight compatible subset then the Nelson Consensus tree equals the intersection of these maximal weight compatible sets. This version of the consensus tree will not contain any unreplicated clusters. These can easily be included, as suggested by Swofford [**42**]. Indeed, the approach adapts well to any weighting scheme for splits and clusters.

If the input collection $\mathcal{T}$ contains unrooted trees and the splits are weighted by the number of input trees they appear in, then the tree formed from a collection of compatible splits with maximum weight is also an *asymmetric median tree* [**36**]. The asymmetric median tree T minimizes

$$d_a(T, \mathcal{T}) = \sum_{i=1}^{k} d_a(T, T_i)$$

where $d_a(T, T_i)$ is the number of splits in T_i that are not in T.

One drawback of the maximum compatible subset approach to consensus is that finding a maximum compatible subset of characters is a computationally difficult problem [**15**]. If there are too many taxa, it may not be possible to locate a maximum weight compatible subset. In this case, a strategy such as the greedy consensus tree (Section 2.1.4) seems preferable.

2.2. Cluster intersection methods. The consensus methods in this section are only defined for rooted trees.

2.2.1. *Adams consensus tree.* Adams consensus was the first consensus method for trees and, perhaps as a consequence, is one of the most popular [**1, 2, 29, 30, 42, 45**]. The method is defined for rooted trees and has no analogue for unrooted trees [**39**]. It is perhaps most simply defined in terms of the algorithm for constructing it. Before that, we need to give three or four new definitions.

First, suppose that $\pi_1, \pi_2, \ldots, \pi_k$ are all partitions of the set of taxa. The *product* of these partitions is the partition π for which two taxa a and b are in the same block if and only if they are in the same block for each of $\pi_1, \pi_2, \ldots, \pi_k$. For example, the product of $ab|cde$ and $ac|bde$ is $a|b|c|de$. The elements d and e are in the same block of both partitions $ab|cde$ and $ac|bde$.

Second, the maximal clusters of a rooted input tree T_i are the largest proper clusters in T. The *maximal cluster partition* for T_i is the partition $\pi(T_i)$ of the set of taxa with blocks equal to the maximal clusters of T_i.

The Adams tree is formed by recursively forming partitions π for $\mathcal{T}$ and restrictions of the trees of $\mathcal{T}$.

Procedure AdamsTree($T_1, \ldots, T_k$)
  If T_1 contains only one leaf **then**
    return T_1
  else
    Construct $\pi(\mathcal{T})$, the product of $\pi(T_1), \pi(T_2), \ldots, \pi(T_k)$.
    For each block B of $\pi(\mathcal{T})$ **do**
      Construct AdamsTree($T_1|_B, T_2|_B, \ldots, T_k|_B$)
    Attach the roots of these trees to a new node v.
    return this tree.
end

Example 2.7. Let T_1 be the rooted tree $(((((a, b), c), d), e), f)$ and let T_2 be the rooted tree $(((((d, e), f), a), b), c)$. The maximal cluster partition of T_1 is $abcde|f$, the maximal cluster partition of T_2 is $def ab|c$ and the product of these two partitions is $abde|c|f$. Hence the Adams tree has three maximal clusters $\{a, b, d, e\}$, $\{c\}$ and $\{f\}$. The restriction of T_1 and T_2 to $\{a, b, d, e\}$ is $(((a, b), d), e)$ and $(((d, e), a), b)$. The maximal cluster partitions of these restrictions are $adb|e$ and $ade|b$ which have product $ad|b|e$. Hence the Adams tree for T_1 and T_2 equals $(((a, d), b, e), c, f)$.

In response to the criticism that the Adams consensus tree has little intuitive justification, Adams showed that the tree preserves all nesting information common to all the input trees. Given two groups A and B and a rooted tree T we say that A *nests* in B if the smallest cluster containing A is a proper subset of the smallest cluster containing B. If T represents ancestor-descendent relationships then this is the same as saying that the most recent common ancestor of A is a strict descendent of the most recent common ancestor of B. Adams showed that if A nests in B in all of the trees in $\mathcal{T}$ then A nests in B in the Adams consensus tree for $\mathcal{T}$.

Note that the Adams tree can contain nestings and clusters that are not in any of the input trees (as in the example). However Adams showed that if A and B are two clusters in the Adams tree such that $A \subseteq B$ then A nests in B in every input tree T_i [**2**]. A consequence of the nesting property is that the Adams consensus tree also preserves the rooted triple information shared by the input trees. The Adams consensus tree also introduces no new rooted triple information:

Theorem 2.8. *Let T^{AD} be the Adams consensus tree for the collection $\mathcal{T} = \{T_1, T_2, \ldots, T_k\}$. Then*

$$\bigcap_{i=1}^{k} r(T_i) \subseteq r(T^{AD}) \subseteq \bigcup_{i=1}^{k} r(T_i).$$

2.2.2. *Cluster height methods.* This class of consensus methods contains the Neumann consensus tree and its various extensions: the Durschnitt consensus tree [**33**], the cardinality rule consensus tree [**33**], and the s-consensus trees [**40, 41**]. Each cluster in every tree in the collection is assigned a height, and this height increases or decreases monotonically with respect to set inclusion. For any given value h and any tree T_i in the collection there is a partition of the leaf set given

by those maximal clusters with height less than h (or sometimes minimal clusters with height greater than h). The Neumann consensus tree is constructed by taking, for each value of h, the partition product of these partitions, where the partition product is as defined for the Adams consensus tree. The variations on the Neumann consensus tree each stem from use of a different height function.

The construction of the s-consensus tree is a little more complicated. A Neumann consensus tree is built using the cardinality rule, and then clusters are removed from this tree if they have too little consensus strength, this measure being defined in [**40, 41**].

2.3. Consensus methods based on subtrees. All of the consensus methods discussed so far have used clusters or splits as the basic unit of information. Now I describe a handful of consensus methods based on rooted triples and quartets. There are several advantages of this approach, the main being that the set of quartets (or rooted triples) common to two trees typically contains a lot more information than the set of common splits or clusters. Two trees can share a lot of rooted triples but not share a single cluster.

The consensus methods of Sections 2.3.1 and 2.3.3 originate from Kannan *et al.* [**26**], with some modifications. Kannan *et al.* discuss a number of additional consensus methods based on triples. However these are not defined for all possible input collections, so I omit them from the survey.

2.3.1. *Local consensus tree.* Let $\mathcal{T}$ be a collection of rooted trees and let $R = \bigcap_{i=1}^{k} r(T_i)$ be the set of rooted triples appearing in all trees T_i of $\mathcal{T}$. The set R is *compatible*, which for rooted triples means that there is a tree T such that $R \subseteq r(T)$.

Given any set of compatible rooted triples R, we can use the algorithm of Aho *et al.* [**3**] to construct a tree T such that $R \subseteq r(T)$ (see also [**12, 14, 23, 34**]). The *local consensus tree* for $\mathcal{T}$ is the tree constructed by Aho *et al.*'s algorithm when given the set R of rooted triples appearing in every tree of $\mathcal{T}$. Aho *et al.*'s algorithm is outlined in appendix A.

Example 2.9. We again consider the two trees $T_1 = (((((a, b), c), d), e), f)$ and $T_2 = (((((d, e), f), a), b), c)$. There are exactly two rooted triples common to both trees: $ab|c$ and $de|f$. When we apply Aho *et al.*'s supertree algorithm to these two rooted triples we obtain the local consensus tree $((a, b), c, (d, e), f)$.

The local consensus tree is an example of an RVII consensus tree under the terminology of Kannan *et al.* [**26**]. They describe an algorithm for constructing an RVII tree in $O(n^2)$ for two trees. The algorithm is identical to that for constructing the Adams consensus tree. However the Adams tree is neither equal to the local consensus tree nor is it an RVII tree. For example, the local consensus tree of $((a, b, c), d)$ and $(a, (b, c, d))$ is (a, b, c, d) while the Adams consensus tree is $(a, (b, c), d)$.

The link between the local consensus tree and the Adams tree is less direct:

Theorem 2.10. *Let $\mathcal{T}$ be a collection of rooted trees and let R be the set of rooted triples present in all of the trees in $\mathcal{T}$. Then the local consensus tree for $\mathcal{T}$ equals the Adams consensus tree for the collection of all trees T such that $R \subseteq r(T)$.*

Note that there exists no analogue of the Adams consensus tree for unrooted trees and quartets [**39**].

2.3.2. *Prune and regraft tree.* The prune and regraft tree is, as the name suggests, constructed in two steps. First we prune leaves and taxa from all the trees in the collection. Second we graft the pruned leaves back on to create a new consensus tree. The method was invented by Gordon [**20**] and only applies to collections of rooted trees. Our presentation differs considerably from that of [**20**] or [**19**], though the tree constructed is the same.

An *agreement set* for a collection $\mathcal{T} = \{T_1, T_2, \ldots, T_k\}$ is a subset X of taxa such that removing all taxa not in X from every tree in $\mathcal{T}$ gives a collection of identical trees. Equivalently,

$$T_1|_X = T_2|_X = \cdots = T_k|_X,$$

where $T_i|_X$ denotes the restriction of T_i to X defined in Section 1.3. The tree $T_i|_X$ is called an *agreement subtree* (or common pruned tree) of $\mathcal{T}$.

The first step in constructing the prune and regraft tree is to determine a maximum size agreement set for $\mathcal{T}$. Farach *et al.* [**17**] describe an efficient algorithm for computing maximum size agreement sets. Another is implemented in PAUP* [**43**].

Let X be a maximum size agreement subset for $\mathcal{T}$. Consider each cluster A of $T_1|_X$ in turn. (Choosing any other tree $T_i \in \mathcal{T}$ gives the same outcome.) Form a new cluster A_G from A by adding all taxa a such that $A \cup \{a\}$ is a cluster in the strict consensus of $\mathcal{T}|_{X\cup\{a\}}$. Recall that $\mathcal{T}|_{X\cup\{a\}}$ is the restriction of every tree in $\mathcal{T}$ to the set of taxa $X \cup \{a\}$. The prune and regraft tree is built up from all these clusters A_G (one for each cluster in $T_1|_X$) as well as all of the clusters in the strict consensus tree for $\mathcal{T}$. Finden and Gordon proved that these cluster do actually form a tree [**19**].

Example 2.11. Once again consider the two trees $T_1 = (((((a, b), c), d), e), f)$ and $T_2 = (((((d, e), f), a), b), c)$. The trees have two maximum agreement subsets, $\{a, b, c\}$ and $\{d, e, f\}$. Taking the first set, we have $T_{1|X} = ((a, b), c)$, a tree with two clusters. Put $A = \{a, b\}$. Since $\{a, b, d\}$ is not a cluster of T_1 and T_2 restricted to $X \cup \{d\} = \{a, b, c, d\}$ the taxon d is not in A_G. The same applies for e and f. Hence the prune and regraft tree corresponding to $X = \{a, b, c\}$ is $((a, b), c, d, e, f)$.

Example 2.12. The unique prune and regraft tree for

$$\mathcal{T} = \{(((a, e), b, f), (c, d)), ((a, (b, e)), (c, d, f))\}$$

is $((a, b, e), (c, d), f)$.

Both the Adams consensus tree and local consensus tree have the property that a rooted triple appearing in all trees will appear in the consensus tree. This property does not hold for the prune and regrafted tree (cf. example 2.11), though the complementary property does hold:

Theorem 2.13. *If $ab|c$ is a rooted triple in a prune and regraft tree for $\mathcal{T}$ then there is at least one tree $T \in \mathcal{T}$ such that $ab|c \in r(T)$, that is, the prune and regraft tree is co-Pareto with respect to rooted triples.*

2.3.3. *Q^* and R^* consensus trees.* The Q^* and R^* methods complement the local consensus tree. Whereas the local consensus method constructs a tree that *contains* all of the rooted triples in a set, the Q^* and R^* consensus methods construct trees with all quartets or rooted triples that are *contained* within a given set.

We will consider the unrooted case first. Let $\mathcal{T}$ be a collection of unrooted trees. Let $f(ab|cd)$ be the number of trees in $\mathcal{T}$ that have $ab|cd$ in their quartet sets. Construct the set of quartets $ab|cd$ such that $f(ab|cd) > f(ac|bd)$ and $f(ab|cd) > f(ad|bc)$ and denote this set by Q_{maj}. In other words, Q_{maj} is the set of quartets that appear in more trees than the quartets they disagree with. It is clear that for each set of four leaves $\{a, b, c, d\}$ at most one of $ab|cd$, $ac|bd$, $ad|bc$ is in Q.

We use the set of quartets Q_{maj} as input to the Q^* method of Berry and Gascuel [**9**]. This constructs the maximum refined tree T such that $q(T) \subseteq Q_{maj}$. We call this tree the *Q^* (Q star) consensus tree* for $\mathcal{T}$.

The R^* consensus method for rooted trees is defined in a similar fashion. Let R_{maj} be the set of rooted triples $ab|c$ that appear in more trees than their conflicting triples $ac|b$ or $bc|a$. The strong cluster algorithm of [**8**] can be used to construct the maximum refined rooted tree T such that $r(T) \subseteq R_{maj}$. We call this the *R^* (R star) consensus tree* for $\mathcal{T}$. When $\mathcal{T}$ contains exactly two trees the R^* consensus tree is identical to the RVIII consensus tree of [**26**].

Though the Q^* and R^* consensus trees are defined using quartets and rooted triples, they have a close relationship with split and cluster based consensus trees.

THEOREM 2.14. *Let $\mathcal{T}$ be a collection of unrooted (rooted) trees. Every split (cluster) in the majority rule tree for $\mathcal{T}$ and every split (cluster) in the loose consensus tree for $\mathcal{T}$ is in the Q^* consensus tree (R^* consensus tree) for $\mathcal{T}$.*

2.4. Consensus methods based on recoding. The next three consensus methods all use the same strategy: the input trees are converted into another kind of data (sequences, distances), and these data are subsequently re-analysed using a phylogenetic tree construction method.

2.4.1. *Matrix representation with parsimony.* Matrix representation with parsimony (MRP) was devised independently by Baum [**7**] and Ragan [**37**] as a method for combining phylogenies from different data sets. Here we discuss only the special case when all the input trees have the same taxa, though the method extends to unequal taxa sets.

Let $\mathcal{T}$ be a collection of unrooted trees. For each split $A|B$ in each tree T_i we construct a binary character that assigns a zero to each element in A and a one to each element in B. It does not matter if we assign ones to A and zeros to B instead. The set of binary characters is then used as input to Fitch parsimony. (Refer to [**44**] for a review of parsimony criteria. Fitch parsimony corresponds to finding Steiner trees with respect to Hamming distances.) The *MRP consensus tree* is the strict consensus tree of the set of maximum parsimony trees for this data set. The MRP consensus tree for a collection of rooted trees can be constructed using an outgroup (zero-row).

EXAMPLE 2.15. Let T_1, T_2, T_3, T_4 be the unrooted trees $((a, b, c), (d, e, f))$, $((a, d, e), (b, c, f))$, $((b, d, e), (a, c, f))$, $((c, d, e), (a, b, f))$. Let $\mathcal{T}$ be the collection with four copies of tree T_1 and one copy each of T_2, T_3, T_4. This gives a binary character matrix

a	1	1	1	1	1	0	0
b	1	1	1	1	0	1	0
c	1	1	1	1	0	0	1
d	0	0	0	0	1	1	1
e	0	0	0	0	1	1	1
f	0	0	0	0	0	0	0

PAUP* [**43**] finds three maximum parsimony trees. They have a strict consensus tree of $((a, b, c), (d, e), f)$ which is therefore the MRP consensus tree for $\mathcal{T}$. In contrast, the majority rule tree, greedy consensus tree, Nelson-Page tree and asymmetric median tree all equal T_1.

Despite the quite different mechanism for constructing consensus trees, the MRP consensus tree is still closely related to the other consensus trees.

THEOREM 2.16. *The MRP consensus tree contains all of the splits (clusters) of the loose consensus tree, and therefore all of the splits (clusters) of the strict consensus tree.*

The parsimony length of an MRP tree can be viewed as a measure of the fit of the tree to the input collection. Given two trees, the length of the first tree compared to the matrix encoding of the second tree gives an indication of how well the first tree fits the second. What MRP does is generalise this notion to multiple trees. Let $l(T, T_i)$ denote the fit of T to the matrix encoding of tree T_i. Then the score of a tree with respect to the combined matrix encoding used in MRP is exactly

$$\sum_{i=1}^{k} l(T, T_i).$$

Thus, as a consensus method, MRP finds the tree that best fits the input trees.

2.4.2. *Average consensus tree.* The average consensus tree of Lapointe and Cucumel [**28**] can be viewed as a distance based version of MRP. Distance matrices, rather than binary encodings, are combined.

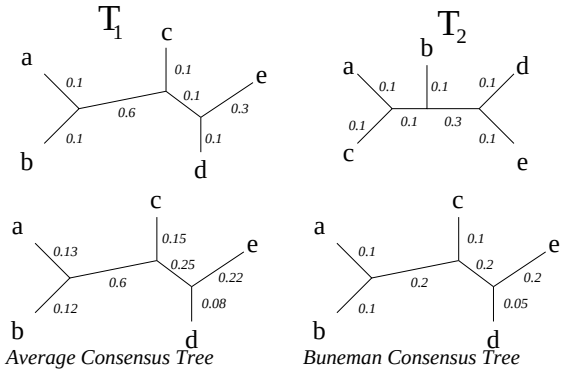

FIGURE 1. An example of an Average consensus tree and Buneman consensus tree. Branch lengths on the consensus trees are given to two decimal places.

The method is used for weighted trees: unrooted trees with branch lengths. The *path length* between two taxa a and b on a weighted tree is the sum of the branch lengths along the path between a and b. The *path length distance matrix* of a tree is the matrix of path lengths between the taxa. There is a one-to-one correspondence between weighted trees and path length distance matrices[1] [**22, 13**].

The *least squares difference* between two weighted trees T_1 and T_2 is defined

$$\Delta(T_1, T_2) = \sum_a \sum_b [d_1(a,b) - d_2(a,b)]^2$$

where d_1 and d_2 are the path length distances for T_1 and T_2. The summation ranges over the complete set of taxa. Given a collection $\mathcal{T} = \{T_1, T_2, \ldots, T_k\}$ of unrooted trees with branch lengths the *average consensus tree* is the unrooted tree T_c (with branch lengths) that minimizes

$$\Delta(T, \mathcal{T}) = \sum_{i=1}^{k} (T_c, T_i).$$

Equivalently, the average consensus tree is the tree with path length distances d_c minimizing

$$\sum_a \sum_b [d_c(a,b) - \overline{d}(a,b)]^2$$

where $\overline{d}$ is

$$\overline{d}(a,b) = \frac{1}{k} \sum_{i=1}^{k} d_i(a,b),$$

the average of the path length distance matrices of the input trees T_i.

EXAMPLE 2.17. Let T_1 and T_2 be the two weighted unrooted trees in figure 1. The additive distance matrices, and average matrix are

Taxon	T_1					T_2					average				
a	0					0					0				
b	0.2	0				0.3	0				0.25	0			
c	0.8	0.8	0			0.2	0.3	0			0.5	0.55	0		
d	0.9	0.9	0.3	0		0.6	0.5	0.6	0		0.75	0.7	0.45	0	
e	1.1	1.1	0.5	0.4	0	0.6	0.5	0.6	0.2	0	0.85	0.8	0.65	0.3	0

PAUP* [**43**] finds exactly one optimal least squares tree. It has the same shape as T_1 but different branch lengths (Figure 1).

The average consensus method is one of the few consensus methods for phylogenetic trees that incorporates branch length information into the computation of the consensus tree. However it has a number of drawbacks. First, like the MRP tree, there is no efficient algorithm for constructing the average consensus tree. Second, it is not known whether or not the consensus method is even *Pareto*, i.e., whether a split that appears in every tree also appears in the consensus tree.

[1] under the assumption that all branches have strictly positive length and all internal nodes have degree at least three.

2.4.3. *Buneman consensus tree.* The Buneman tree consensus method is a distance based analogue of the Q^* consensus method (Section 2.3.3). It is a hybrid between the average consensus tree and the tree construction method introduced by Buneman [**13**]. As with the average consensus tree, we first compute the average distance matrix

$$\overline{d}(a,b) = \frac{1}{k}\sum_{i=1}^{k} d_i(a,b)$$

where d_i is path length matrix for T_i. We then construct the Buneman tree for $\overline{d}$. This tree contains exactly those splits $A|B$ such that $\overline{d}(a,a') + \overline{d}(b,b') < \overline{d}(a,b) + \overline{d}(a',b')$ for all $a,a' \in A$ and $b,b' \in B$. An efficient algorithm for constructing the Buneman tree was given by Berry and Bryant [**8**]. This tree is called the *Buneman consensus tree* for $\mathcal{T}$.

EXAMPLE 2.18. The Buneman consensus tree for the two weighted trees T_1 and T_2 in example 2.17 equals tree T_1, with branch lengths given in figure 1.

The Buneman consensus tree avoids many of the disadvantages of average consensus tree. It can be constructed quickly and is Pareto.

THEOREM 2.19. *Every split in the strict consensus tree for $\mathcal{T}$ will be a split in the Buneman consensus tree for $\mathcal{T}$.*

3. A Classification of Consensus Methods

We have established that there are a lot of consensus methods to choose from, but which one is best? What information does each method retain and how do we interpret the output trees? In a sense we face a classification problem: we need to produce a classification of consensus methods.

A consensus method is usually introduced by comparing the amount of information it retains to the amount of information retained by the strict consensus tree. The appeal of this approach is that the strict consensus tree is almost always poorly resolved. Here we extend the comparison to classify all consensus methods.

Figure 2 is a classification of consensus trees. An arrow from consensus method X to consensus method Y means that for all collections $\mathcal{T}$, the consensus tree for $\mathcal{T}$ made using method Y always refines the consensus tree for $\mathcal{T}$ made using method X. We have indicated which methods are co-Pareto on splits or clusters (every split in the consensus tree appears in at least one input tree), co-Pareto on rooted triples (every rooted triple in the consensus tree appears in at least one input tree), and Pareto on rooted triples (every rooted triple appearing in all input trees appears in the consensus tree).

There are two complementary facets to the classification. First, the inclusion results. These are either straightfoward or follow from Theorems in the text. Second, the non-inclusion results. If there is no arrow from one consensus method to another then we have an example collection for which the consensus tree produced by the first method is not refined by the consensus tree produced by the second method. There are considerably more non-inclusion results than inclusion results. Many are straightfoward, only involving trees with three or four taxa. Most of the less trivial counter-examples are given as examples in the text above.

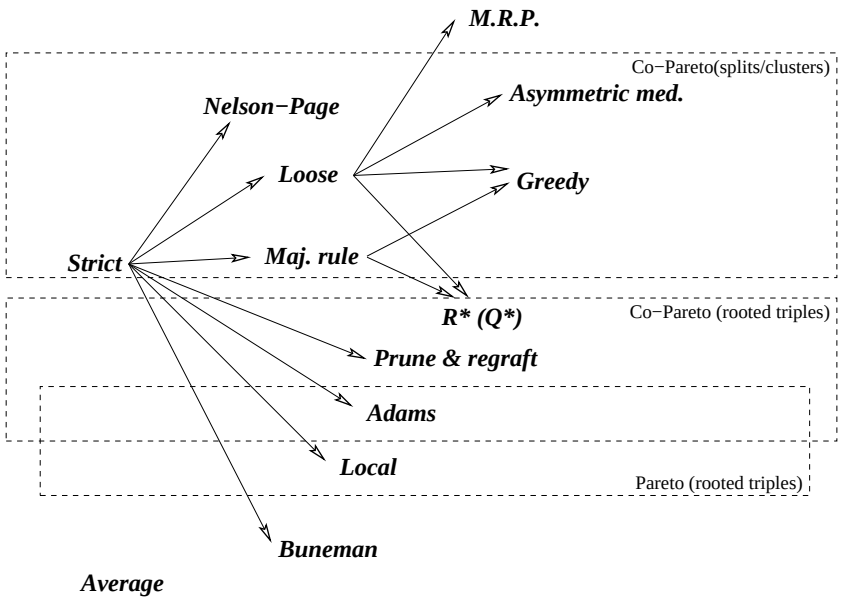

FIGURE 2. A classification of consensus methods. There is an arrow from one method to another if every split in the consensus tree produced by the first method is contained in every consensus tree produced by the second method.

4. Beyond consensus: subtrees and supertrees

All of the consensus methods that we have discussed satisfy:

(1) All the input trees have equal taxa sets;
(2) The output tree has the same taxa set as the input trees.

In application, one or both of the properties may not be suitable. In this case we must look beyond the consensus tree to subtree and supertree methods.

A *consensus subtree method* constructs a consensus tree for which (2) does not necessarily hold. There are some cases when the removal of a taxon leads to a far better representation of the branching structure shared by two or more trees. One clear example is when a single distantly related taxa appears in different positions on otherwise identical trees, a situation that can occur when using outgroups. The wandering taxon is best removed from a consensus tree.

The most widely used consensus subtree method is the agreement subtree. An agreement subtree, or common pruned tree, for a collection $\mathcal{T}$ is a tree T on a subset X of the taxa set such that $T = T_i|_X$ for all $i = 1, \ldots, k$ [**21**]. Here $T = T_i|_X$ means that T is the restriction of T_i to X (see section 2.2.1). The problem of constructing maximum size agreement subtrees became popular with algorithm designers, generating a flurry of papers with faster and faster algorithms. The current record for two trees is held by [**27**] and for arbitrarily many (with bounded degree) by [**17**]. The algorithm of [**11**] has been implemented as part of PAUP*.

Several alternative subtree techniques have been proposed by Wilkinson [**45**].

A *consensus supertree* method can construct a consensus tree when property (1) is violated, that is, when the input trees contain different collections of taxa [**21**]. There are several important situations when this problem occurs. Supertrees are used to combine analyses of several data sets, each of which contains information for different groups of taxa. Supertrees are also used as part of divide and conquer strategies for constructing large phylogenies.

Several consensus methods have been adapted to the supertree problem, including the strict consensus tree [**24**], Adams consensus tree [**38**], MRP [**7, 37**], and the average consensus tree [**28**]. Wilkinson and Thorley [**46**] have developed a method producing supertrees on a subset of the total collection of taxa: a subtree supertree method.

Appendix A. The algorithm of Aho et al.

Here we outline the supertree algorithm OneTree based on the algorithm of Aho *et al.* [**3**]. The algorithm takes a collection of rooted triples R and a set of leaves L and constructs a tree T with leaf set L such that $R \subseteq r(T)$, provided such a tree exists.

Given a set or rooted triples R and set of leaves S define the graph $[R, S]$ as follows: take the leaves in S to be the vertices of the graph; add an edge between two vertices a and b if there are any triples in R of the form $ab|c$, where $a, b, c \in S$.

For example, consider the graph $[R, S]$ when

$$R = \{ab|c, be|d, be|c, af|g, ef|b, bf|a, bf|c, cd|a, cd|f, cg|b\}$$

and $S = \{a, b, c, d, e, f\}$. The triples $af|g$ and $cg|b$ are ignored, because $g \notin S$. The graph has six vertices and five edges. Note that the edges $(b, f), (b, e)$ and (c, d) correspond to more than one rooted triple (Figure 3).

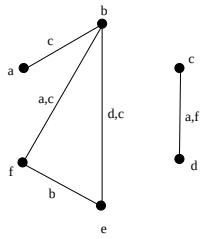

FIGURE 3. The graph $[R, S]$ for $R = \{ab|c, be|d, be|c, af|g, ef|b, bf|a, bf|c, cd|a, cd|f, cg|b\}$ and $S = \{a, b, c, d, e, f\}$.

Two vertices are in the same *component* of $[R, S]$ if there is a path from one two the other. The algorithm computes the different components of $[R, S]$ and then recurses. To determine whether there is a tree T with leaf set L such that $R \subseteq r(T)$ we would call OneTree(R,L).

Procedure OneTree(R,S)
1. **If** $n = 1$ **then return** a single vertex labelled by x_1.
2. **If** $n = 2$ **then return** a tree with two leaves labelled x_1 and x_2.
3. Otherwise, construct $[R, S]$ as described.
4. **If** $[R, S]$ has only one component **then return** 'No Tree'.
5. **For** each component S_i of $[R, S]$ **do**
6. **If** OneTree(R,S_i) returns a tree **then** call it T_i **else return** 'No Tree'.
7. **end(for)**
8. Construct a new tree T by connecting the roots of the trees T_i to a new root r.
9. **return** T.

end.

Appendix B. Proofs

Theorem 2.6. *The greedy selection method produces a consensus tree which refines both the majority rule tree and the loose consensus tree.*

Proof. The algorithm for constructing the greedy consensus tree examines splits in decreasing order of frequency. Hence those splits or clusters that appear in more than half of the trees will be considered first for inclusion in $\mathcal{S}$. These splits (clusters) form the majority rule and are compatible, so will all be included in $\mathcal{S}$. The greedy consensus tree therefore refines the majority rule tree.

Let $A|B$ be a split in the loose consensus tree, then $A|B$ is compatible with all splits in the input collection. When we consider $A|B$ during the construction of the greedy consensus tree, we will necessarily have that $A|B$ is compatible with all of the splits already in $\mathcal{S}$. Hence $A|B$ will always be included in $\mathcal{S}$ and become a split of the greedy consensus tree. □

Theorem 2.8. *Let T^{AD} be the Adams consensus tree for the collection $\mathcal{T} = \{T_1, T_2, \ldots, T_k\}$. Then*

$$\bigcap_{i=1}^{k} r(T_i) \subseteq r(T^{AD}) \subseteq \bigcup_{i=1}^{k} r(T_i).$$

Proof. Suppose that $ab|c$ is a rooted triple in every input tree T_i. Then $\{a, b\}$ nests in $\{a, b, c\}$ for every tree T_i and so by the main result of [**2**] we have that $\{a, b\}$ nests in $\{a, b, c\}$ in T_{AD}. Thus $ab|c \in r(T_{AD})$.

For the other inclusion, choose $ab|c \in r(T^{AD})$. There is some set S such that the product partition given by the maximal subtrees of $T_1|_S, \ldots, T_k|_S$ has a and b in one block and c in another. Hence there is some tree T_i such that a and b are in one maximal subtree of $T_i|_S$ and c is in another. It follows that $ab|c \in r(T_i)$. □

Theorem 2.10. *Let $\mathcal{T}$ be a collection of rooted trees and let R be the set of rooted triples present in all of the trees in $\mathcal{T}$. Then the local consensus tree for $\mathcal{T}$ equals the Adams consensus tree for the collection of all trees T such that $R \subseteq r(T)$.*

Proof. Let $\langle R \rangle$ denote the set of trees T such that $R \subseteq r(T)$. Both the Aho *et al.* tree and the Adams tree are defined recursively, starting with the maximal clusters and working down towards the leaves. If we can show that the partition π_1 given by the maximal subtrees of the Aho *et al.* tree for R is the same as the

partition π_2 given by the maximal subtrees of the Adams consensus tree for $\langle R \rangle$ then the result follows by recursion.

The Aho *et al.* tree is in $\langle R \rangle$ so the partition π_2 equals the partition product of π_1 together with all the other partitions from trees in $\langle R \rangle$. Hence π_2 is a refinement of π_1.

Let S be the set of all leaves in R. If a and b are in the same block of π_1 then they are in the same component of the graph $[R, S]$ defined in Section A. Any two leaves connected by an edge in $[R, S]$ will be in the same maximal subtree of any tree in $\langle R \rangle$. Hence any two leaves in the same component of $[R, S]$ will also be in the same maximal subtree in any tree in $\langle R \rangle$. Therefore a and b are in the same maximal subtree in any tree in $\langle R \rangle$ and they are in the same block of π_2.

The two partitions π_1 and π_2 are refinements of each other, so $\pi_1 = \pi_2$. □

THEOREM 2.13. *If $ab|c$ is a rooted triple in a prune and regraft tree for $\mathcal{T}$ then there is at least one tree $T \in \mathcal{T}$ such that $ab|c \in r(T)$, that is, the prune and regraft tree is co-Pareto with respect to rooted triples.*

PROOF. Let T_G be a prune and regraft tree for $\mathcal{T}$ and let $ab|c$ be a rooted triple in $r(T_G)$. Let X be the agreement set that gave T_G and let C be the a cluster of T_G such that $a, b \in C$ and $c \notin C$. By the construction of T_G, the set $C \cap X$ is non-empty.

We have that $(C \cap X) \cup \{a\}$ is in the strict consensus tree for $\mathcal{T}$ restricted to $X \cup \{a\}$ and $(C \cap X) \cup \{b\}$ is in the strict consensus tree for $\mathcal{T}$ restricted to $X \cup \{b\}$. Since $c \notin C$, we have that $(C \cap X) \cup \{c\}$ is not in the strict consensus tree for $\mathcal{T}$ restricted to $X \cup \{c\}$ and so there is $T \in \mathcal{T}$ such that $C \cap X \cup \{c\}$ is not a cluster of $T|_{X \cup \{c\}}$.

Let A be a cluster of T such that $A \cap (X \cup \{a\}) = (C \cap X) \cup \{a\}$. Let B be a cluster of T such that $B \cap (X \cup \{b\}) = (C \cap X) \cup \{b\}$. The clusters A and B are compatible and both contain $C \cap X$, so either $A \subseteq B$ or $B \subseteq A$. W.l.o.g. $A \subseteq B$. Hence $a, b \in B$.

Now $B \cap X = C \cap X$, so if $c \in B$ then

$$\begin{aligned} B \cap (X \cup \{c\}) &= (B \cap X) \cup (B \cap \{c\}) \\ &= (C \cap X) \cup \{c\} \end{aligned}$$

and so $(C \cap X) \cup \{c\}$ is a cluster of $T|_{X \cup \{c\}}$, a contradiction. Thus $c \notin B$ and $ab|c \in T$. □

THEOREM 2.14. *Let $\mathcal{T}$ be a collection of unrooted (rooted) trees. Every split (cluster) in the majority rule tree for $\mathcal{T}$ and every split (cluster) in the loose consensus tree for $\mathcal{T}$ is in the Q^* consensus tree (R^* consensus tree) for $\mathcal{T}$.*

PROOF. For each quartet $ab|cd$ let $f(ab|cd)$ be the number of trees T_i such that $ab|cd \in q(T_i)$.

Let $A|B$ be a split in the loose consensus tree for T. Let $aa'|bb'$ be a quartet such that $a, a' \in A$ and $b, b' \in B$. By the definition of the loose consensus tree there is at least one tree T_i with split $A|B$. It follows that $aa'|bb' \in q(T_i)$ and so $f(aa'|bb') \geq 1$. Suppose that, for example, $f(ab|a'b') > 0$. Then there is one tree T_j such that $ab|a'b' \in q(T_j)$, and so there must be a split $C|D$ of T_j such that $a, b \in C$ and $a', b' \in D$. But then $a \in A \cap C$, $a' \in A \cap D$, $b \in B \cap C$, and $b' \in B \cap D$ so the two splits $A|B$ and $C|D$ are not compatible. This contradicts the definition of the loose consensus tree because $A|B$ is compatible with all of the trees in $\mathcal{T}$.

Thus $f(ab|a'b') = 0$ and, by the same argument $f(ab'|a'b) = 0$, which means that $aa'|bb' \in Q_{maj}$.

Hence all the quartets $aa'|bb'$ such that $a, a' \in A$ and $b, b' \in B$ are in Q_{maj} and so $A|B$ is in the Q^* tree for Q_{maj} and therefore in the Q^* consensus tree for $\mathcal{T}$.

Now suppose that $A|B$ is in the majority rule tree for $\mathcal{T}$. Then $A|B$ is a split in over half of the trees in $\mathcal{T}$ and so any quartet $aa'|bb'$ such that $a, a' \in A$ and $b, b' \in B$ is present in over half the trees in $\mathcal{T}$. Since $f(aa'|bb') + f(ab|a'b') + f(ab'|a'b) = k$ and $f(aa'|bb') > k/2$ we have that $aa'|bb' \in Q_{maj}$. It follows that $A|B$ is in the Q^* consensus tree for $\mathcal{T}$.

The proof for the R^* rooted consensus tree follows the same lines. □

In the following, we identify binary characters with splits. Thus a split is compatible with a binary character if and only if it is compatible with the corresponding splits.

LEMMA B.6. *Let $\mathcal{C}$ be a collection of binary characters. If $A|B$ is compatible with every character in $\mathcal{C}$ then there is a maximum parsimony tree for $\mathcal{C}$ containing split $A|B$. Furthermore, if $A|B$ corresponds to a character in $\mathcal{C}$ then every maximum parsimony tree for $\mathcal{C}$ has split $A|B$.*

PROOF. Suppose that $A|B$ is compatible with every character in $\mathcal{C}$.

Since each character in $\mathcal{C}$ is compatible with $A|B$ every character must be constant over all of B or over all of A. Let $\mathcal{C}_1$ be the set of characters that are constant over A and let $\mathcal{C}_2$ be the remaining characters. Thus all the characters in C_2 are constant over B.

Choose an arbitrary taxa a from A and b from B. Put $A' = A \cup \{b\}$ and $B' = B \cup \{a\}$. Let $\mathcal{C}'_1$ be the set of characters in $\mathcal{C}_1$ restricted to taxa in B' and let $\mathcal{C}'_2$ be the set of characters in $\mathcal{C}_2$ restricted to A'.

Let T_1 be a maximum parsimony tree for $\mathcal{C}'_1$. It has taxa set B'. Let T_2 be a maximum parsimony tree for $\mathcal{C}'_2$. It has taxa set A'. Let v_1 be the node adjacent to a in T_1 and v_2 the node adjacent to b in T_2. Let T be the tree obtained by removing a from T_1 and b from T_2 then joining T_1 and T_2 using a branch from v_1 to v_2. The tree T contains the split $A|B$. We claim that T is a maximum parsimony tree for $\mathcal{C}$.

Let $l(T_1, \mathcal{C}'_1)$ be the length of T_1 with respect to $\mathcal{C}'_1$ and let $l(T_2, \mathcal{C}'_2)$ be the length of T_2 with repsect to $\mathcal{C}'_2$. Let C be a character in $\mathcal{C}_1$ and let C' be the character in $\mathcal{C}'_1$ that is the restriction of C to B'. By the choice of $\mathcal{C}_1$, the character C is constant over all of A. Hence a parsimonious assignment of internal states for C on T will be constant over the subtree of T with taxa set A, that is, the subtree derived from T_2. It follows that the length of C on T is the same as the length of C' on T_1 and, by symmetry, the length of T equals the length $l(T_1, \mathcal{C}'_1)$ of T_1 plus the length $l(T_2, \mathcal{C}'_2)$ of $\mathcal{C}_2$.

Now suppose that $\overline{T}$ is some other tree with the same taxon set as T. Let $\overline{T}_1$ be $\overline{T}$ restricted to B' and let $\overline{T}_2$ be $\overline{T}$ restricted to A'. If we remove taxa from a tree we cannot increase the length of the tree. Hence

$$l(\overline{T}_1, \mathcal{C}'_1) \leq l(\overline{T}, \mathcal{C}_1)$$

and

$$l(\overline{T}_2, \mathcal{C}'_2) \leq l(\overline{T}, \mathcal{C}_2).$$

The length of $\overline{T}$ with respect to $\mathcal{C}$ is $l(\overline{T}, \mathcal{C}_1) + l(\overline{T}, \mathcal{C}_2)$. Since the trees T_1 and T_2 are maximum parsimony trees for $\mathcal{C}_1'$ and $\mathcal{C}_2'$ we have

$$\begin{aligned} l(T, \mathcal{C}) &= l(T_1, \mathcal{C}_1') + l(T_2, \mathcal{C}_2') \\ &\leq l(\overline{T}_1, \mathcal{C}_1') + l(\overline{T}_2, \mathcal{C}_2') \\ &\leq l(\overline{T}, \mathcal{C}_1) + l(\overline{T}, \mathcal{C}_2) \\ &= l(\overline{T}, \mathcal{C}). \end{aligned}$$

Thus the length of any other tree $\overline{T}$ with respect to $\mathcal{C}$ is at least as much as the length of T, and T is a maximum parsimony tree for T. This proves the first part.

For the second part, suppose that $A|B$ corresponds to a character in $\mathcal{C}$. Let $\mathcal{C}'$ be the set of characters in $\mathcal{C}$ that do not correspond to $A|B$. By the first part, there is a maximum parsimony tree T for $\mathcal{C}'$ containing the split $A|B$. Each character in $\mathcal{C} - \mathcal{C}'$ corresponds to the split $A|B$ so has length one on T and $l(T, \mathcal{C}) = l(T, \mathcal{C}') + |\mathcal{C} - \mathcal{C}'|$. On any tree $\overline{T}$ that does not contain the split $A|B$ the character corresponding to $A|B$ has length at least two. Thus

$$\begin{aligned} l(\overline{T}, \mathcal{C}) &= l(\overline{T}, \mathcal{C}') + l(\overline{T}, \mathcal{C} - \mathcal{C}') \\ &\geq l(T, \mathcal{C}') + l(\overline{T}, \mathcal{C} - \mathcal{C}') \\ &\geq l(T, \mathcal{C}') + 2|\mathcal{C} - \mathcal{C}'| \\ &> l(T, \mathcal{C}). \end{aligned}$$

Hence all maximum parsimony trees contain the split $A|B$. □

THEOREM 2.16. *The MRP consensus tree contains all of the splits (clusters) of the loose consensus tree, and therefore all of the splits (clusters) of the strict consensus tree.*

PROOF. If $A|B$ is in the loose consensus tree then $A|B$ is compatible with every split of every tree in $\mathcal{T}$. Hence $A|B$ is compatible with the binary psuedo-characters given by splits of trees in $\mathcal{T}$. By Lemma B.6, $A|B$ is in every MRP tree. □

THEOREM 2.19. *Every split in the strict consensus tree for $\mathcal{T}$ will be a split in the Buneman consensus tree for $\mathcal{T}$.*

PROOF. If $A|B$ is a split of a weighted tree T with strictly posistive edge lengths and pathlength distance matrix d then

$$d(a, a') + d(b, b') < d(a, b) + d(a', b')$$

for all choices of $a, a' \in A$ and $b, b' \in B$ [**13**]. Let $d_1, d_2, \ldots, d_k$ be the path length matrice for $T_1, T_2, \ldots, T_k$. Since $A|B$ is a split in every tree T_i we have

$$\overline{d}(a, a') + \overline{d}(b, b') = \frac{1}{k} \sum_{i=1}^{k} (d_i(a, a') + d_i(b, b')) \tag{B.1}$$

$$< \frac{1}{k} \sum_{i=1}^{k} (d_i(a, b) + d_i(a', b')) \tag{B.2}$$

$$= \overline{d}(a, b) + \overline{d}(a', b') \tag{B.3}$$

for all $a, a' \in A$ and $b, b' \in B$. Hence $A|B$ is a split in the Buneman tree for $\overline{d}$ and therefore a split in the Buneman consensus tree for $\mathcal{T}$. □

References

1. E.N. Adams, *Consensus techniques and the comparison of taxonomic trees*, Syst. Zool. **21** (1972), 390–397.
2. ———, *N-trees as nestings: complexity, similarity, and consensus*, J. Classif **3** (1986), 299–317.
3. A.V. Aho, T.G. 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), no. 3, 405–421.
4. M. Barrett, M. Donoghue, and E. Sober, *Against consensus*, Syst. Zool. **40** (1991), no. 4, 486–493.
5. J. P. Barthélemy and F. R. McMorris, *The median procedure for n-trees*, J. Classif. **3** (1986), 329–334.
6. J. P. Barthélemy, F. R. McMorris, and R. C. Powers, *Dictatorial consensus functions on n-trees*, Math. Soc. Sci. **25** (1992), 59–64.
7. B.R. Baum, *Combining trees as a way of combining data sets for phylogenetic inference, and the desirability for combining phylogenetic trees*, Taxon **41** (1992), 3–10.
8. V. Berry and D. Bryant, *Faster reliable phylogenetic analysis*, Proc. 3rd international conference on computational molecular biology (RECOMB), vol. 3, 1999, pp. 59–68.
9. Vincent Berry and Olivier Gascuel, *Inferring evolutionary trees with strong combinatorial evidence*, Theor. Comput. Sci. **240** (2000), no. 2, 271–298.
10. K. Bremer, *Combinable component consensus*, Cladistics **6** (1990), 369–372.
11. D. Bryant, *Building trees, hunting for trees, and comparing trees*, Ph.D. thesis, University of Canterbury, Dept. Mathematics, 1997.
12. D. Bryant and M. Steel, *Extension operations on sets of leaf-labelled trees*, Adv. Appl. Math. **16** (1995), 425–453.
13. P. Buneman, *The recovery of trees from measures of dissimilarity*, Mathematics in the Archaeological and Historical Sciences (F.R. Hodson, D.G. Kendall, and P. Tautu, eds.), Edinburgh University Press, Edinburgh, 1971, pp. 387–395.
14. M. Constantinescu and D. Sankoff, *An efficient algorithm for supertrees*, J. Classif. **12** (1995), no. 1, 101–112.
15. W.H.E. Day and D. Sankoff, *Computational complexity of inferring phylogenies by compatibility*, Syst. Zool. **35** (1986), no. 2, 224–229.
16. A. de Queiroz, *For consensus (sometimes)*, Syst. Biol. **42** (1993), no. 3, 368–372.
17. M. Farach, T. Przytycka, and M. Thorup, *On the agreement of many trees*, Inform. Process. Lett. **55** (1995), 297–301.
18. J. Felsenstein, *Phylip*, http://evolution.genetics.washington.edu/phylip.html, 1993.
19. C.R. Finden and A.D. Gordon, *Obtaining common pruned trees*, J. Classif. **2** (1985), 255–276.
20. A. Gordon, *On the assessment and comparison of classifications*, Analyse de Données et Informatique (R. Tomassine, ed.), INRIA, Le Chesnay, 1980, pp. 149–160.
21. A. D. Gordon, *Consensus supertrees: the synthesis of rooted trees containing overlapping sets of labeled leaves*, J. Classif. **3** (1986), 335–348.
22. J.A. Hartigan, *Representation of similarity matrices by trees*, J. Am. Stat. Assoc. **62** (1967), 1140–1158.
23. M.R. Henzinger, V. King, and T. Warnow, *Constructing a tree from homeomorphic subtrees, with applications to computational evolutionary biology*, Algorithmica **24** (1999), 1–13.
24. D. Huson, S. Nettles, and T. Warnow, *Disk-covering, a fast converging method for phylogenetic tree reconstruction*, J. Comput. Biol. **6** (1999), no. 3/4, 369–386.
25. L. Jermiin, G.J. Olsen, K.L. Mengersen, and S. Easteal, *Majority-rule consensus of phylogenetic trees obtained by maximum-likelihood analysis*, Mol. Biol. Evol. **14** (1997), no. 12, 1296–1302.
26. S. Kannan, T. Warnow, and S. Yooseph, *Computing the local consensus of trees*, SIAM J. Comput. **27** (1998), no. 6, 1695–1724.
27. M.Y. Kao, T.W. Lam, W.K. Sung, and H.F. Ting, *A faster and unifying algorithm for comparing trees*, Combinatorial Pattern Matching, Lecture notes in computer science, vol. 1848, Springer-Verlag, 2000, pp. 241–254.
28. J.F. Lapointe and G. Cucumel, *The average consensus procedure: combination of weighted trees containing identical or overlapping sets of taxa*, Syst. Biol. **46** (1997), no. 2, 306–312.

29. T. Margush and F. R. McMorris, *Consensus n-trees*, B. Math. Biol. **43** (1981), no. 2, 239–244.
30. F. R. McMorris, D. B. Meronk, and D. A. Neumann, *A view of some consensus methods for trees*, Numerical Taxonomy (J. Felsenstein, ed.), Springer-Verlag, 1983, pp. 122–125.
31. F.R. McMorris, *On the compatibility of binary qualitative taxonomic characters*, B. Math. Biol. **39** (1977), 133–138.
32. G. Nelson, *Cladistic analysis and synthesis: principles and definitions, with a historical note on Adanson's* Familles des Plantes (1763-1764), Syst. Zool. **28** (1979), 1–21.
33. D. A. Neumann, *Faithful consensus methods for n-trees*, Math. Biosci. **63** (1983), 271–287.
34. M.P. Ng and N.C. Wormald, *Reconstruction of rooted trees from subtrees*, Discrete Appl. Math. **69** (1996), no. 1-2, 19–31.
35. D. M. Page, *Tracks and trees in the antipodes: A reply to Humphries and Seberg*, Syst. Zool. **39** (1990), no. 3, 288–299.
36. C. Phillips and T.J. Warnow, *The assymetric median tree - a new model for building consensus trees*, Discrete Appl. Math. **71** (1996), 311–335.
37. M.A. Ragan, *Phylogenetic inference based on matrix representation of trees*, Mol. Phylogenet. Evol. **1** (1992), 53–58.
38. C. Semple and M. Steel, *A supertree method for rooted trees*, Discrete Appl. Math. **105** (2000), 147–158.
39. M. Steel, S. Böcker, and A. Dress, *Some simple but fundamental limits for supertree and consensus tree methods*, Syst. Biol. **42** (2000), no. 2, 363–368.
40. R. Stinebrickner, *s-Consensus trees and indices*, B. Math. Biol. **46** (1984), 923–935.
41. ———, *s-Consensus index method: an additional axiom*, J. Classif. **3** (1986), 319–327.
42. D.L. Swofford, *When are phylogeny estimates from molecular and morphological data incongruent?*, Phylogenetic analysis of DNA sequences (M. M. Miyamoto and J. Cracraft, eds.), Oxford University Press, 1991, pp. 295–333.
43. ———, *Paup*. phylogenetic analysis using parsimony (*and other methods)*, vol. 4, Sinauer Associates, Sunderland, Massachusetts, 1998.
44. D.L. Swofford, G.J. Olsen, P.J. Waddell, and D.M. Hillis, *Phylogenetic inference*, Molecular Systematics (D.M. Hillis, C. Moritz, and B.K. Mable, eds.), Sinauer, 2nd ed., 1996, pp. 407–514.
45. M. Wilkinson, *Common cladistic information and its consensus representation: reduced Adams and reduced cladistic consensus trees and profiles*, Syst. Biol. **43** (1994), no. 3, 343–368.
46. M. Wilkinson and J. Thorley, *Reduced consensus supertrees*, Trends Ecol. Evol. **13** (1998), 283.

School of Computer Science and Department of Mathematics and Statistics, McGill University, Montréal, Québec. http://www.math.mcgill.ca/bryant/

E-mail address: bryant@math.mcgill.ca