Uncovering Hidden Phylogenetic Consensus

Nicholas D. Pattengale[1], Krister M. Swenson[23], Bernard M.E. Moret[4]

[1]Department of Computer Science, University of New Mexico, USA
[2]Department of Mathematics and Statistics, University of Ottawa, Canada
[3]LaCIM, Université du Québec à Montréal, Canada
[4]Laboratory for Computational Biology and Bioinformatics, EPFL, Switzerland

Abstract. Many of the steps in phylogenetic reconstruction can be confounded by "rogue" taxa, taxa that cannot be placed with assurance anywhere within the tree—whose location within the tree, in fact, varies with almost any choice of algorithm or parameters. Phylogenetic consensus methods, in particular, are known to suffer from this problem. In this paper we provide a novel framework in which to define and identify rogue taxa. In this framework, we formulate a bicriterion optimization problem that models the net increase in useful information present in the consensus tree when certain taxa are removed from the input data. We also provide an effective greedy heuristic to identify a subset of rogue taxa and use it in a series of experiments, using both pathological examples described in the literature and a collection of large biological datasets. As the presence of rogue taxa in a set of bootstrap replicates can lead to deceivingly poor support values, we propose a procedure to recompute support values in light of the rogue taxa identified by our algorithm; applying this procedure to our biological datasets caused a large number of edges to change from "unsupported" to "supported" status, indicating that many existing phylogenies should be recomputed and reevaluated to reduce any inaccuracies introduced by rogue taxa.

1 Introduction

Phylogenetic consensus methods are used for combining a set of trees defined on the same set of leaves into a single tree that summarizes the information found in the set. By their very nature, these methods discard information, typically structural elements not prevalent in the set. However, the most popular consensus methods (strict and majority rule) are susceptible to so-called *rogue taxa* [19]. That is, while the tree set may agree very strongly on the structure relating a large subset of the leaves, the remaining few leaves (the rogue taxa) can effectively prevent this underlying structure from appearing in the strict or majority consensus tree. In other words, these methods end up discarding structural elements that are, in fact, prevalent in the set.

Much work has been done on the problem of summarizing a set of trees and on the issue of rogue taxa in particular. The pioneering work of Wilkinson [19–21] addresses the problem by returning *sets* of trees, some of which are missing leaves, with the aim of conveying the prevalent structural elements in at least one of the returned trees. While theoretically satisfying, this approach suffers from computational complexity problems and, more importantly, from difficulties in interpretation. A problem closely related to both consensus and rogue taxa is the *Maximum Agreement Subtree (MAST).* A MAST on a set of input trees is the largest subtree common to all input trees. While the general problem of finding the MAST of three or more trees is $\mathcal{NP}$-hard [1], it can be solved efficiently when at least one of the input trees has bounded degree [6]. However, the MAST tends to be too conservative; most notably, there exist instances where the consensus tree (without dropping any leaves) has more internal structure than the MAST [12]. Cranston and Rannala recently presented a Markov Chain Monte Carlo (MCMC) method for identifying a version of rogue taxa in the context of Bayesian phylogenetic reconstruction [5]. Their approach identifies subsets of leaves for which the posterior distribution strongly supports the structure of the induced subtree—leaves left out can be viewed as rogue taxa, albeit in the narrow context of a sampling of trees in a Bayesian search, rather than in the general context of a consensus of trees. All of these approaches fall into the category of "leaf-dropping methods," in the terminology of Redelings [15]. In contrast, Redelings presents, again in the context of Bayesian phylogenetics, a method that returns a "multi-connected tree," which includes all leaves, but does not summarize the information through a single tree and thus again raises issues of interpretation – an issue plaguing all approaches producing non-trees [2, 4, 9, 10].

In this paper we contribute another leaf-dropping method, one based on a rigorous definition of the tradeoff involved between dropping leaves and uncovering additional consensus structure. Most existing measures and methods discard leaves in order to uncover *any* underlying structure; in contrast, our approach sets up a bicriterion problem, in which leaves should be discarded only if the gain in uncovered internal structure outweighs the loss incurred by discarding the leaves. We are not the first researchers to define some notion of relative information content for consensus trees [18], but our definition is the first to both *explicitly* take into account the loss incurred by dropping taxa, and generalize outside the setting of strict consensus. We provide an effective greedy heuristic to compute a good (if not necessarily optimal) set of rogue taxa and apply it to both pathological examples from the literature and a collection of large biological datasets that we used in a prior study of bootstrapping. As the presence of rogue taxa in a set of bootstrap replicates can lead to deceivingly poor support values, we propose a procedure to recompute support values in light of the rogue taxa identified by our algorithm; applying this procedure to our biological datasets caused a large number of edges to change from "unsupported" to "supported" status, indicating that many existing phylogenies should be recomputed and reevaluated to reduce any inaccuracies introduced by rogue taxa.

The rest of the paper is organized as follows. In Section 2 we define concepts and terminology. In Section 3 we define our measure of relative information content, formalize the bicriterion optimization problem for consensus and rogue taxa, and present some theoretical results that underlie our approach. In Section 4 we present an efficient greedy heuristic for our bicriterion problem. In Section 5 we present the results of our experiments.

2 Preliminaries

We use standard set and graph terminology and notation; in particular, $\cup$ refers to union, $\cap$ to intersection, $\setminus$ to set difference, and Δ to symmetric difference—i.e., $S \Delta T = (S \cup T) \setminus (S \cap T)$.

A *phylogenetic tree* represents the evolutionary relationships among a collection of living organisms. Homologous molecular sequences (one for each organism) are placed at the tips of the tree—hereafter called the *leaves*; the internal structure of the tree—its *edges* (sometimes also called branches)—represents the evolutionary relationships. The removal of an edge disconnects the tree and partitions the set of leaves into two subsets; thus each edge corresponds to a *bipartition* of the set of leaves. Every tree includes the same *trivial bipartitions*, which separate one leaf from all others; the other bipartitions are called *nontrivial* and correspond to an *internal* edge of a tree, that is, an edge not incident on a leaf. We can thus view a phylogenetic tree as a leaf-labeled tree $T = (L, B)$, where L is the set of leaves and B is its set of nontrivial bipartitions. To describe a bipartition, we list the two sets of leaves, separated by a $|$ symbol. To ensure an equivalence between nontrivial bipartitions and internal edges, we require that every internal node in a phylogeny have degree at least 3. The number $|B|$ of nontrivial bipartitions in a phylogeny is at most $|L| - 3$; when the two are equal, we say that the (binary) tree is *fully resolved*; otherwise, there must exist an internal node of degree at least 4 and any such node is known as a *polytomy.*

The *consensus* problem is given by a set $\mathcal{T}$ of m trees defined on a common set L of n taxa (leaves). The *bipartition profile* of $\mathcal{T}$ is the pair

$$\mathcal{P} = (B_{\mathcal{T}},\, \nu\colon B_{\mathcal{T}} \to 2^{\mathcal{T}})$$

where $B_{\mathcal{T}}$ is the set of all nontrivial bipartitions found across all m trees in the set and ν is a function mapping bipartitions to the trees in which they appear.

We denote the removal of leaves from trees through the *restriction* operator—which also uses the $|$ symbol. For example, $\mathcal{T}|L'$ refers to restricting each tree in the set $\mathcal{T}$ to the leaf subset $L' \subseteq L$, which corresponds to removing each leaf in $L \setminus L'$ from each tree, as well as removing any nodes of degree 2 created in the process. Individual trees, tree sets, and tree profiles can appear on the left-hand side of the restriction operator.

We focus on consensus methods based on bipartition *frequency*—see the excellent survey of Bryant [3] for a comprehensive treatment of consensus methods. Given a threshold parameter $\frac{m}{2} < t < m$, the t-consensus tree is composed of all of the bipartitions that occur in more than t trees. The *majority rule* consensus [11] is obtained by setting t to $\frac{m}{2}$, while the *strict* consensus is obtained by setting t to $m - 1$. We denote t-consensus methods by $\mathcal{C}_t$. Thus $\mathcal{C}_{m-1}(\mathcal{T})$ corresponds to taking the strict consensus tree of the set $\mathcal{T}$.

3 Relative Information Content, Consensus, Rogue Taxa

3.1 The measure and the problem

The general problem we study can be phrased as follows: given a set $\mathcal{T}$ of trees on a common leaf set L and given a frequency-based consensus method $\mathcal{C}_t$, we want to find a leaf subset L' that optimizes the relative information content of the consensus returned by $\mathcal{C}_t$ on the set of subtrees induced by L'. The crucial notion here is that of relative information content. Formally, if $\mathcal{C}_t(\mathcal{T}|L')$ yields $T' = (L', B')$, then the *relative information content* is

$$I(T', L, \mathcal{C}_t) = \frac{|L'| + |B'|}{|L| + (|L| - 3)} \tag{1}$$

This measure is the ratio of the total number of bipartitions (trivial and nontrivial) in the consensus tree derived on the reduced leaf set to the total number of bipartitions in an ideal, fully resolved tree on the original leaf set. By taking trivial bipartitions into account, we automatically penalize a method for removing many leaves, since the number of trivial bipartitions is simply the number of leaves. By adding the number of nontrivial bipartitions, we reward a method for preserving more internal edges, since the denominator is fixed to the number of such edges in an ideal tree.

We can now formulate our main problem, which we call *MISC*, for *Maximum-Information Subtree Consensus.*

Problem 1 (MISC). Given a set $\mathcal{T}$ of trees defined on a common leaf set L and a frequency-based consensus method $\mathcal{C}_t$, find a leaf subset L' that maximizes the relative information content $I(\mathcal{C}_t(\mathcal{T}|L'), L, \mathcal{C}_t)$.

Note that the MAST solution typically maximizes the $|B'|$ term at the expense of the $|L'|$ term—it has no direct penalty for dropping leaves; in contrast, consensus methods typically maximize $|L'|$ (in the case of majority and strict consensus, by forcing $L' = L$) at the expense of $|B'|$. MISC, on the other hand, combines the two aspects into a single formulation.

3.2 How bipartitions change under leaf deletion

We begin by studying the effect that dropping leaves has on a tree set profile. For any bipartition in the original profile, there are three cases. We illustrate these cases through a simple example, with an original leaf set of a, b, c, d, e, f and with leaves b and e dropped.

1. **merge:** If two bipartitions differ solely in (a subset of) the leaves being dropped, then those bipartitions get merged in the new profile. For example $ac|bdef$ and $abc|def$ merge into $ac|df$ and the ν set for the merged bipartition consists of the union of the two original bipartitions.
2. **disappear:** If dropping the leaves creates a bipartition with an empty side or makes the bipartition trivial, then the bipartition disappears. For example, both $acdf|be$ and $acd|bef$ disappear.

3. **no change:** Otherwise, a bipartition remains unchanged.

An important observation is that, for all $L'' \subseteq L' \subseteq L$, every nontrivial bipartition in $\mathcal{P}|L''$ and in $\mathcal{C}_t(\mathcal{T}|L'')$ arises as a result of a "no change" of a single bipartition or a "merge" of two or more bipartitions in $\mathcal{P}|L'$. Unfortunately this observation does not suggest an efficient algorithm.

3.3 Finding subsets of leaves to drop

Given two bipartitions b_1 and b_2 of L, we can easily identify all leaf subsets L' of minimum cardinality such that dropping L' from L merges b_1 and b_2. If we have $b_1 = A|B$ and $b_2 = C|D$, then the *dropset* L' is the smaller of the two following sets (or either set in case they have the same size):

$$(A \Delta C) \cup (B \Delta D) \text{ or } (A \Delta D) \cup (B \Delta C) \qquad (2)$$

This concept is exploited in Algorithm 1. Observe that, in the terminology of [16], the dropset of b_1 and b_2 is the largest partial X-split such that b_1 and b_2 both extend it.

Theorem 2. *Algorithm 1 computes the minimum cardinality dropset for any pair of bipartitions of L.*

Proof. That the dropset causes the two partitions to merge is evident. We establish that the dropset has minimum cardinality by contradiction. Consider that there exists a smaller dropset merging the two bipartitions. Then there is at least one leaf ℓ in the dropset returned by our algorithm that is not in the smaller dropset. This leaf must be on the same side of the partition in both b_1 and b_2, since otherwise our dropset would not merge the two. But our algorithm uses the symmetric difference of these two sides in computing the dropset, so it could not have chosen ℓ, a contradiction. □

Algorithm 1 Find minimum cardinality leaf-dropset that renders $b_1 = b_2$

```
Input: two bipartitions on the same leaf set
Output: the dropset (or dropsets if there are two)
 1: function BIPARTITION-PAIR-DROPSET($b_1 = A|B$, $b_2 = C|D$)
 2:     $S_0 \leftarrow A \Delta C \cup B \Delta D$
 3:     $S_1 \leftarrow A \Delta D \cup B \Delta C$
 4:     if $|S_0| < |S_1|$ then
 5:         return $[S_0]$
 6:     else if $|S_1| < |S_0$ then
 7:         return $[S_1]$
 8:     else
 9:         return $[S_0, S_1]$
10:     end if
11: end function
```

Theorem 3. *The cardinalities of the dropsets returned by Algorithm 1 define a metric on the space of bipartitions of L.*

Proof. Three properties characterize a metric: it must be positive definite and symmetric, and it must obey the triangle inequality. The first two properties are trivial in this case. Suppose we have bipartitions b_1, b_2, and b_3; we want to show that the cardinality of the dropset of b_1 and b_3 cannot exceed the sum of the cardinalities of the dropsets of b_1 and b_2 and of b_2 and b_3. Note that removing both of these dropsets from both b_1 and b_3 merges the two bipartitions, thereby establishing an upper bound on the distance between these two bipartitions in our space; but the distance is the size of the dropset of b_1 and b_3, so that the triangle inequality holds. □

4 The Algorithm

We describe the algorithm at a conceptual level, leaving a more formal specification to inset text. First, we build the bipartition profile for the given tree set. Next, we compute the dropset for each pair of bipartitions in the profile such that neither bipartition in the pair appears in the consensus tree, but the pair would appear if merged. For each unique dropset we accumulate the list of bipartition pairs yielding that dropset. These last two parts are formalized in Algorithm 2. We then compute the *impact* of each dropset as the number of bipartition pairs giving rise to that dropset minus the size of the dropset itself. This score corresponds roughly to the difference between the number of edges that will be created and the number of leaves that will be lost should that dropset be used. The dropset of largest impact is then used, the profile updated, the impacts updated, and the process repeated until there does not remain any dropset with a nonnegative impact. This greedy overall framework is formalized in Algorithm 3.

Algorithm 2 Find potential dropsets by examining all pairs in a profile

```
Input: A bipartition profile $\mathcal{P} = (L, B_{\mathcal{T}}, \nu : B_{\mathcal{T}} \to 2^{\mathcal{T}})$
Input: A frequency-only consensus method $\mathcal{C}_t$ with threshold $t$
Output: An object mapping dropsets to lists of bipartition pairs
 1: function POTENTIAL-PROFILE-DROPSETS($\mathcal{P}$, $\mathcal{C}_t$)
 2:     $\Gamma \leftarrow \{b \mid b \in B_{\mathcal{T}}$ and $|\nu(b)| \le t\}$
 3:     for all pairs of bipartitions $b_1$,$b_2$ in $\Gamma$ do
 4:         if $|\nu(b_1) \cup \nu(b_2)| > t$ then
 5:             $L \leftarrow$ BIPARTITION-PAIR-DROPSET($b_1, b_2$)
 6:             for $d \in L$ do
 7:                 $\delta[d] \leftarrow \delta[d] \cup \{(b_1, b_2)\}$
 8:             end for
 9:         end if
10:     end for
11:     return $\delta$
12: end function
```

Algorithm 3 Our top level iterative heuristic for finding dropsets

```
Input: A tree set $\mathcal{T}$
Input: A frequency-only consensus method $C$ with threshold $t$
Output: A set of leaves to drop, composed of the union of dropsets
 1: function SELECT-AND-REMOVE-DROPSETS($\mathcal{T}$)
 2:     $d^* \leftarrow d_{greedy} \leftarrow \emptyset$
 3:     repeat
 4:         $\mathcal{P} \leftarrow$ BUILD-BIPARTITION-PROFILE($\mathcal{T}|(L - d^*)$)
 5:         $\delta \leftarrow$ POTENTIAL-PROFILE-DROPSETS($\mathcal{P}, \mathcal{C}_t$)
 6:         $maximpact = 0$
 7:         $d_{greedy} = \emptyset$
 8:         for all $d \in \delta$'s domain do
 9:             if $|d| - |\delta[d]| \geq maximpact$ then
10:                 $d_{greedy} = d$
11:                 $maximpact = |d| - |\delta[d]|$
12:             end if
13:         end for
14:         $d^* = d^* \cup d_{greedy}$
15:     until $d_{greedy} = \emptyset$
16:     return $d^*$
17: end function
```

The impact measure ignores disappearing edges and dropsets that are subsets of another—the latter because a superset with deceivingly poor score is likely to get chosen in a subsequent round. The overall algorithm is a greedy heuristic, but does well in practice and on hard instances, as we demonstrate in the next two sections.

There remains the issue, as with all leaf-dropping methods, of what to do with the dropped leaves. The staying power of consensus methods argues for producing a single tree and our method does that. For the rogue taxa, we provide an intriguing strategy that is applicable in some settings in Section 5.3.

5 Experimental Results

We have implemented our approach as a standalone Python-based prototype. Our current implementation is suitable for datasets of up to a thousand trees on a thousand leaves. Scaling up to 10,000 trees on 10,000 leaves is simply a matter of reimplementing our approach as part of RAxML [17] so as to leverage the efficient bipartition manipulation routines therein. In the following, we present results on artificial datasets constructed to cause difficulties to various consensus methods, followed by results on biological datasets that we used in previous work on bootstrapping. We then discuss implications of our results on the interpretation of phylogenetic reconstruction. We conclude by a smaller study on biological datasets using a slight modification of our algorithm to maximize the number of nontrivial bipartitions in the result.

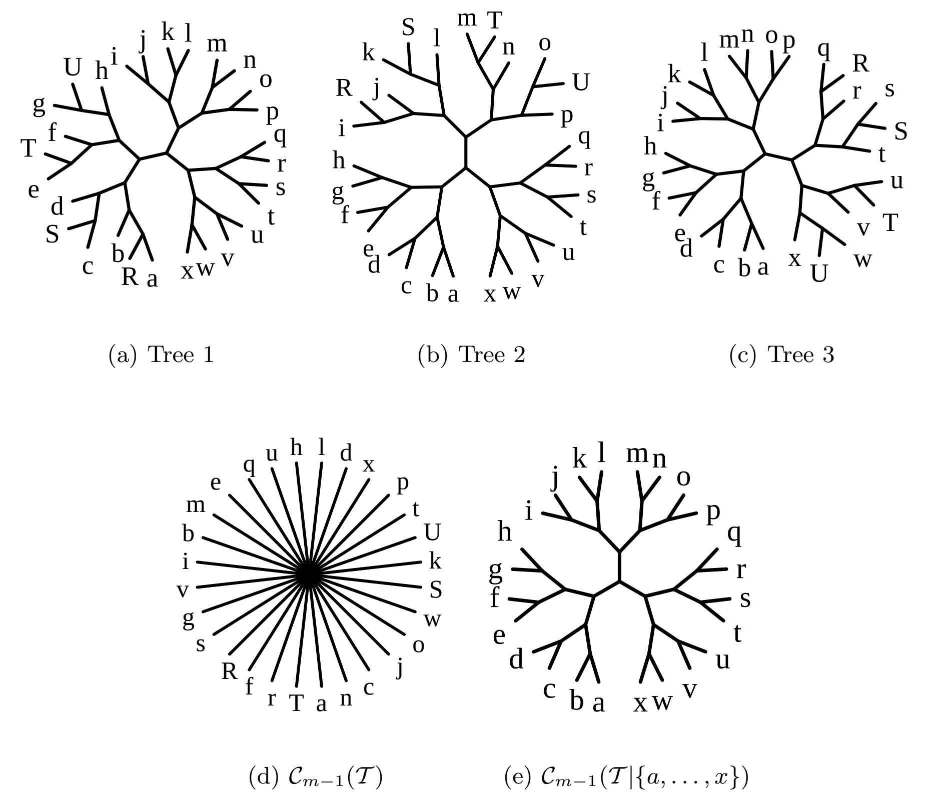
(a) Tree 1
(b) Tree 2
(c) Tree 3
(d) $\mathcal{C}_{m-1}(\mathcal{T})$
(e) $\mathcal{C}_{m-1}(\mathcal{T}|\{a, \ldots, x\})$

Fig. 1. *A simple, yet starkly contrasting, example (top) for which the strict consensus returns a star tree, but for which our algorithm correctly identifies the rogue taxa and produces a fully resolved tree (bottom).*

5.1 Difficult instances

Our algorithm is particularly well suited to the so-called "pathological" instances used in the literature to critique the strict or majority consensus. A classic example is an instance where the trees share a common subtree of $n-k$ leaves, but where the remaining k leaves destroy resolution in the consensus. The example we present below is rather simple and space limits prevent us from given more examples; suffice it to say that our algorithm has no problem in overcoming most of the pathological cases encountered in the literature.

Our example uses the strict consensus. An instance consists of just three trees, defined on the 28-leaf set $\{a, b, \ldots, x, R, S, T, U\}$. The common backbone consists of the 24 taxa $\{a, b, \ldots, x\}$, as illustrated in Figure 1(e)). The rogue taxa form the set $\{R, S, T, U\}$; they vary in position on the backbone as indicated in Figures 1(a), 1(b), and 1(c). The strict consensus tree of the three trees is shown in Fig. 1(d): it is just a star, with no nontrivial bipartition (no internal tree edge) and its relative information content is $I(\mathcal{T}, L, \mathcal{C}_{m-1}) = \frac{28+0}{28+25} = \frac{28}{53} \approx 0.53$. Our algorithm correctly identifies the rogue set, however, so that its strict consensus tree on the remaining set of leaves is the backbone, with an relative information content of $I(\mathcal{T}|\{a, \ldots, x\}, L, \mathcal{C}_{m-1}) = \frac{24+21}{28+25} = \frac{45}{53} \approx 0.85$.

5.2 Results on biological data

We applied our method to the datasets we used in an earlier study of bootstrapping methods [13] and available at http://lcbb.epfl.ch/BS.tar.bz2. There are 10 datasets of single-gene and multi-gene DNA sequences, with anywhere from 125 to 994 taxa. For each dataset we generated 1,000 bootstrap replicates and applied our algorithm to the resulting trees using both $\mathcal{C}_{\frac{m}{2}}$ and $\mathcal{C}_{m-1}$. Our algorithm found rather diverse dropset sizes across the 10 datasets. The results are depicted in Figure 2, where a quartet of histogram bars are shown for each dataset with a nonempty dropset. The first histogram bar (a negative quantity) denotes how many leaves were dropped, while the second bar (a positive quantity) denotes how many nontrivial bipartitions were uncovered. The third bar is the sum of the first two, simply depicting the net (non-normalized) contribution to relative information content. The final bar is discussed in Section 5.3.

5.3 Biological interpretation

It is common practice to conduct and interpret a maximum likelihood phylogenetic analysis as follows. First the reconstruction proper is performed, which yields the “best tree.” Next, some number of bootstrap replicate trees are generated, say 500 of them. For each bipartition b in the best tree, its support value is calculated as a normalized count of the number of replicates in which b appears. Researchers tend to consider edges with support lower than 75% as unreliable [7].

If, however, rogue taxa are at work in the replicate set, the support values for certain bipartitions can be deceivingly depressed. To remedy this problem, we propose that Algorithm 3 be applied to the replicate set in order to identify rogue taxa. If a dropset of nonzero size is found, this dropset is then to be removed from each tree in the replicate set. Finally, the (modified) support value is calculated as a normalized count of the replicates in which b' appears such that, if we have $b = A|B$, then, without loss of generality, we have $b' = A' \subseteq A|B' \subseteq B$. In this way, support values in the “best tree” are less susceptible to the deceiving influence of rogue taxa. This approach offers one possible solution to the data display problem of leaf-dropping methods. We still return a single tree on the original leaf set (the “best tree” as reconstructed by an ML method), but support values for individual bipartitions more accurately reflect the underlying replicate data.

In our datasets, recomputing support values as suggested above yields very intriguing and promising results. All but two of the identified dropsets succeeded in pushing at least one previously hidden edge in the “best tree” over the 75%

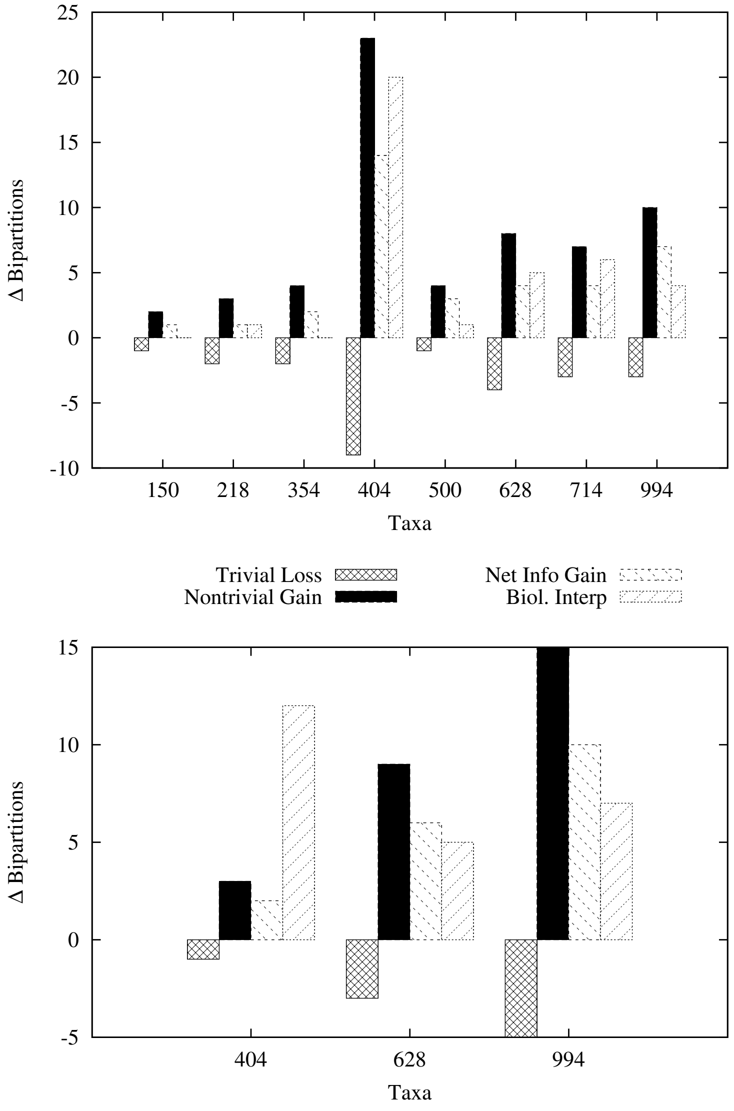

Fig. 2. The performance of Algorithm 3 in terms of how much "hidden" consensus is uncovered in biological data sets. The top plot is for majority consensus, the bottom for strict consensus. The tree sets each consist of 1,000 bootstrap replicates generated by the RAxML 7.2.5 Rapid Bootstrap Algorithm.

threshold. The number of edges uncovered by this application of our technique is displayed in the fourth histogram bar in Figures 2(a) and 2(b). In the dataset with 404 taxa, 20 edges were uncovered in this manner, pointing to a need for reevaluation of the phylogeny.

5.4 Increasing resolution

Our algorithm can easily be modified to maximize nontrivial bipartitions, that is, to remove taxa so as to increase resolution. With such a setting, our algorithm matches the goal of Cranston and Rannala [5], so we analyzed the same dataset with our technique to compare our results to theirs. The data set consists of 85 species of Canformia Carnivora [8]. We obtained the sequence data from TreeBASE (http://www.treebase.org, Study Accession # S1532) and reconstructed a tree using RAxML-7.2.5 [17] under the GTRCAT approximation. Additionally, RAxML was used to generate 350 bootstrap replicates (the number chosen by RAxML's bootstopping algorithm). Analyzing these 350 trees with our modified Algorithm 3 and using majority consensus generated fully resolved trees with 50 to 55 taxa, a value consistent with the size of the agreement subtrees observed by Cranston and Rannala [5].

6 Conclusions and Future Work

We have presented a novel framework to define rogue taxa so as to maximize the relative information present in a consensus tree computed after removing these rogue taxa. This framework defines a bicriterion problem, MISC, that is the first to balance explicitly loss of taxa with gain in resolution. We have also provided an effective greedy heuristic to find a good set of such rogue taxa. This algorithm was tested on both pathological cases from the literature and a variety of biological data. The changes in the consensus tree can be parlayed into more accurate bootstrap scores, which in turn can lead to the reevaluation of phylogenetic trees, as we showed on our biological datasets.

Further algorithmic work includes a characterization of the computational complexity of the MISC problem, as well as improved algorithms for it, including approximation algorithms with known performance guarantees. Generalizing our approach to support consensus methods other than frequency-based ones is another algorithmic problem worth investigating. Finally, there is certainly room to extend and apply our techniques in different domains, most notably in Bayesian phylogenetics (as suggested in Section 5.4) and for the subtree mergers used in the Disk-Covering Methods (as suggested in [14]). On the bioinformatics side, our preliminary findings indicate that existing phylogenies can be significantly refined by applying our approach to the recomputation of bootstrap support.

Acknowledgements

NDP thanks Alexandros Stamatakis for useful feedback regarding experimental design as well as for initially collating the biological datasets used in Section 5.

References

1. A. Amir and D. Keselman. Maximum agreement subtree in a set of evolutionary trees. *SIAM Journal on Computing*, 26:758–769, 1994.

2. H. Bandelt and A. Dress. Split decomposition: A new and useful approach to phylogenetic analysis of distance data. *Molecular Phylogenetics and Evolution*, 1(3):242–252, September 1992.
3. D. Bryant. A classification of consensus methods for phylogenetics. In *Bioconsensus*, volume 61 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 163–184. American Math. Soc. Press, 2002.
4. D. Bryant and V. Moulton. Neighbor-Net: An Agglomerative Method for the Construction of Phylogenetic Networks. *Mol Biol Evol*, 21(2):255–265, 2004.
5. K. A. Cranston and B. Rannala. Summarizing a Posterior Distribution of Trees Using Agreement Subtrees. *Syst Biol*, 56(4):578–590, 2007.
6. M. Farach, T. M. Przytycka, and M. Thorup. On the agreement of many trees. *Information Processing Letters*, 55(6):297–301, 1995.
7. J. Felsenstein. *Inferring Phylogenies*. Sinauer Associates, Inc., 2004.
8. T. L. Fulton and C. Strobeck. Molecular phylogeny of the arctoidea (carnivora): Effect of missing data on supertree and supermatrix analyses of multiple gene data sets. *Molecular Phylogenetics and Evolution*, 41(1):165–181, October 2006.
9. O. Gauthier and F.-J. Lapointe. Seeing the Trees for the Network: Consensus, Information Content, and Superphylogenies. *Syst Biol*, 56(2):345–355, 2007.
10. D. Huson. SplitsTree: analyzing and visualizing evolutionary data. *Bioinformatics*, 14(1):68–73, 1998.
11. T. Margush and F. R. McMorris. Consensus n-trees. *Bulletin of Mathematical Biology*, 43:239–244, 1981.
12. N. D. Pattengale. *Efficient Algorithms for Phylogenetic Post-Analysis.* PhD thesis, University of New Mexico, 2010.
13. N. D. Pattengale, M. Alipour, O. R. P. Bininda-Emonds, B. M. E. Moret, and A. Stamatakis. How many bootstrap replicates are necessary? In S. Batzoglou, editor, *RECOMB*, volume 5541 of *Lecture Notes in Computer Science*, pages 184–200. Springer, 2009.
14. N. D. Pattengale, K. M. Swenson, M. M. Morin, and B. M. E. Moret. Higher fidelity subtree merging for disk-covering methods. Poster, Algorithmic Biology, 2006. http://www.calit2.net/events/algorithmicbio/files/PattengaleAlgoBio2006.pdf.
15. B. Redelings. Bayesian phylogenies unplugged: Majority consensus trees with wandering taxa. http://www4.ncsu.edu/∼bdredeli/wandering.pdf.
16. C. Semple and M. Steel. Tree reconstruction via a closure operation on partial splits. In *Gascuel, M.-F. Sagot (Eds.), Proceedings of Journes Ouvertes: Biologie, Informatique et Mathmatiques, Lecture Notes in Computer Science*, pages 129–134. Springer-Verlag, 2001.
17. A. Stamatakis. RAxML-VI-HPC: maximum likelihood-based phylogenetic analyses with thousands of taxa and mixed models. *Bioinformatics*, 22(21):2688–2690, 2006.
18. J. L. Thorley, M. Wilkinson, and M. Charleston. The information content of consensus trees. In A. Rizzi, M. Vichi, and H. Bock, editors, *Studies in Classification, Data Analysis, and Knowledge Organization*, Advances in Data Science and Classification, pages 91–98. Springer, 1998.
19. M. Wilkinson. Common Cladistic Information and its Consensus Representation: Reduced Adams and Reduced Cladistic Consensus Trees and Profiles. *Syst Biol*, 43(3):343–368, 1994.
20. M. Wilkinson. More on reduced consensus methods. *Syst. Biol.*, 44:435–439, 1995.
21. M. Wilkinson. Majority-rule reduced consensus trees and their use in bootstrapping. *Mol Biol Evol*, 13(3):437–444, 1996.