Accurate Simulation of Large Collections of Phylogenetic Trees

Suzanne J. Matthews, *Member, IEEE, ACM*

Abstract—Phylogenetic analyses are growing at a rapid rate, producing increasingly large collections of trees. Scientists rarely share their tree collections, making it difficult for researchers to develop methods that anticipate and respond to this growth of data. While common methods for simulating phylogenetic trees focus on random topologies, the tree collections returned from phylogenetic search are rarely random and contain a high degree of topological similarity. In this paper, we introduce TreeSim, a software package that simulates large tree collections from published consensus trees. TreeSim implements our new simulation algorithm, the *combined consensus*. Our experimental results indicate that simulating trees based on the combined consensus produces collections whose topological diversity most closely resemble the trees returned from phylogenetic search. We expect that TreeSim will play a critical role in guiding the algorithmic development of new approaches that support the growth of phylogenetic data.

Index Terms—TreeSim, phylogeny, simulation, experimentation, tree collections, consensus

1 Introduction

Modern phylogenetic analyses are rapidly increasing in size and scope. The growing cheapness of "next-gen" sequencing techniques and ubiquity of high performance phylogenetic search packages such as MrBayes [1] enable scientists to analyze molecular data encompassing several hundreds of taxa, or organisms. The goal of all phylogenetic analysis is to produce a phylogenetic tree, a binary tree depicting the evolutionary relationships between a set of taxa.

Large phylogenetic searches produce tree collections that consist of tens to hundreds of thousands of trees. Scientists combine these tree collections into a single tree called the consensus, discarding the source trees in the process. Despite extensive research [2], [3], [4], [5] discussing the importance of such collections, scientists do not retain or publish the collections returned from phylogenetic search. Once discarded, the trees produced by a particular phylogenetic analysis are lost forever. Only 11 percent of the papers examined in a recent study [4] provide the data needed to re-run a phylogenetic search. Even if this data is available, re-running the heuristic search can take weeks to months to complete, and never guarantees the same set of trees.

In addition to impeding reproducibility, the sparsity of tree collections prevents the creation of new methods that can accommodate the growth of phylogenetic data. In the absence of real tree collections, scientists who design phylogenetic methods are forced to simulate tree collections. However, most of the available methods focus on generating random topologies. Random tree collections are a poor choice for algorithm writers, as the trees returned from phylogenetic search are rarely random, and contain a high degree of topological similarity to each other [3], [6], [7]. Fast software that produces collections that mimic the topological diversity found in real tree collections is essential to ensure that new and existing algorithms can handle the ever-growing datasets produced by modern phylogenetic analyses.

In this paper we describe TreeSim, an application that simulates tree collections from published consensus trees. These trees commonly appear in biological studies that conduct phylogenetic analysis, along with statistics on the number of trees outputted by the search heuristic and the number of taxa under study. TreeSim implements the *combined consensus* simulation algorithm, a novel method that simulates a collection based on inputted strict and majority consensus trees.

We use three large, diverse biological tree collections from published phylogenetic analyses to benchmark the different simulation algorithms. Robinson-Foulds distance [8], the most popular distance metric for comparing trees, is used to measure topological diversity in our collections under study. Our experimental results indicate that the collections produced by the combined consensus algorithm most closely reflect the topological diversity found in tree collections returned from phylogenetic search. Since the majority and strict consensus trees are usually published in biological studies, our method will allow researchers to simulate realistic collections when real datasets are unavailable. We anticipate that TreeSim will aid in the benchmarking and creation of novel methods for phylogenetic data analysis.

• *S.J. Matthews is with the Department of Electrical Engineering & Computer Science, United States Military Academy, West Point, NY, 10996. E-mail: suzanne.matthews@usma.edu*

2 Prior Work

Approaches for simulating tree collections typically require the creation of an n-taxa starting tree. Kuhner and Felsenstein [9] describe a method for generating a random starting tree with the goal of creating individual topologies with varying levels of reconstruction difficulty and realistic branch lengths [9]. A simpler random starting tree can be constructed as follows. Starting with a pool of n taxa, randomly partition it into two daughter sets, dropping edges between the parent and the children. Repeat the process recursively until each taxon occupies a single leaf in the tree. A fully resolved unrooted starting tree contains $n-3$ internal edges (or bipartitions), and requires $O(n)$ time to construct.

A collection of t identical trees over n taxa is constructed by duplicating a single n-taxa starting tree t times. A collection of t different trees is produced by generating t separate n-taxa random starting trees. The collections produced by these methods are either identical or maximally dissimilar. However, trees produced from phylogenetic search typically do not reside at either extreme, and contain a high degree of topological similarity, reflecting the close relationship between the produced trees.

2.1 Consensus resolution approaches

Sul and Williams [10] use the *consensus resolution* to simulate a collection of trees. The consensus resolution (r) is the percentage of bipartitions from a source tree collection ($\mathcal{T}$) that is included in the constructed consensus tree. The strict consensus resolution is the percentage of bipartitions in the consensus tree that are contained in every tree in $\mathcal{T}$. The majority consensus resolution is percentage of bipartitions that are contained in a majority ($> 50\%$) of the trees in $\mathcal{T}$. Note that the majority consensus necessarily must be at least as large as the strict consensus resolution.

For example, a majority consensus of 100 percent indicates that every bipartition in the consensus tree appears in a majority of the trees in $\mathcal{T}$. A strict consensus of 100 percent denotes that all the trees in the collection are identical. A strict consensus of 50 percent indicates that 50 percent of the bipartitions in the consensus tree appear in every tree in $\mathcal{T}$. A majority consensus of 0 percent indicates that no bipartition appears in the majority of the trees (a star topology), a result that implies that trees in $\mathcal{M}$ are very dissimilar to each other. The strict and majority consensus algorithms below closely resemble those originally described [10] by Sul and Williams. In the original algorithms, bipartitions were sampled from a provided random starting tree.

A collection of t trees over n taxa with a strict consensus resolution s is created by sampling s percent of the $n-3$ bipartitions of the starting tree, producing a "seed" topology which is assigned to each of the t output trees. At this point, each of the t output trees is an identical strict consensus tree with resolution s, implying that the remaining $(1-s) \times 100$ percent of the bipartitions in each tree are unresolved. Each output tree is fully resolved to a binary tree by randomly adding the remaining bipartitions.

To simulate a collection with majority consensus resolution m, we randomly sample m percent of the starting tree's bipartitions; call this set $\mathcal{R}$. For each bipartition $b \in \mathcal{R}$, we generate a random number p between 50 and 100 and assign b to p percent of the t output trees. We then iteratively build each tree with its assigned bipartitions, resolving each tree fully to a binary tree by adding random bipartitions as necessary.

2.2 Shortcomings

Both algorithms suffer from some shortcomings. The trees in a collection simulated from the strict consensus are biased toward a single core topology, and may have no other sources of similarity. A collection of trees simulated from the majority consensus rate enjoys a more even distribution of bipartitions. However, the bipartitions in a consensus tree with a majority resolution rate of 100 percent need only appear in 50 percent of the source trees. Therefore, up to 50 percent of the $n-3$ bipartitions in each tree are left to be randomly resolved. The resulting tree collection may contain too much topological diversity, despite having a high majority consensus rate.

Sul and Williams kindly make their code for the strict consensus and majority consensus algorithms available in packages `unresolverstr` [11] and `unresolvermaj` [12], respectively. Unfortunately, the published code produces malformed topologies on large datasets and requires a prohibitive amount of memory and run-time. In addition, the requirement of inputting resolution rates mandates that the user infers appropriate resolution parameters, an unclear process.

3 TreeSim

The goal of TreeSim is to produce high quality simulated collections of trees. We accomplish this by using the consensus trees provided by a particular phylogenetic analysis. Unlike sequence alignments and tree collections, consensus trees commonly appear in the text of the paper, along with the number of trees produced by the analysis and the number of taxa. Increasingly, researchers deposit their consensus trees in TreeBASE [13] and other databases. Software like TreeRipper [14] can also be used to extract phylogenetic trees directly from the text of a paper.

3.1 Combined consensus collection

The hallmark of TreeSim is our novel combined consensus simulation algorithm. Unlike the majority or strict consensus simulation algorithms, the combined consensus uses both a majority consensus resolution m and a strict consensus resolution s, where $m \geq s$. The end result is that the tree collections have topological diversity levels that more closely resemble real collections. Tree collections outputted from phylogenetic search have multiple dimensions of similarity, and typically have non-zero strict and majority consensus rates.

The combined consensus simulation algorithm requires an n-taxa majority consensus tree, and the corresponding strict consensus tree. We start by reading in each tree, calculating the resolutions s and m in the process. Let $\mathcal{S}$ denote the set of bipartitions in the strict consensus tree, and $\mathcal{R}$ be the set of bipartitions from the majority consensus tree. Note that $\mathcal{S} \subseteq \mathcal{R}$. The set S is assigned to each of the t output trees. Next, each bipartition $b \in \mathcal{R} - \mathcal{S}$ is assigned to some p percent of the output trees, where $50 \leq p \leq 100$.

As an example, consider a 13-taxa tree (10 bipartitions) with a majority and strict consensus resolutions of 90 percent and 50 percent, respectively. The majority consensus tree has 9 bipartitions ($\mathcal{R}$), and the strict consensus tree has 5 bipartitions ($\mathcal{S}$). We assign the 5 bipartitions in $\mathcal{S}$ to each of the t output trees, while the remaining 4 bipartitions in $\mathcal{R}$ are assigned to p percent of the t trees, where $p \geq 50$. Each tree is then fully resolved to a binary tree.

3.2 Strengths

The sampling of the strict consensus bipartitions from the majority consensus bipartitions guarantees that s percent of the bipartitions in the starting tree are contained in all the output trees. Adding the remaining bipartitions in $\mathcal{R}$ to a majority of the trees guarantee that there is another dimension of similarity in the output trees. In addition to the combined consensus, TreeSim also implements updated forms of the majority and strict consensus algorithms from `unresolverstr` and `unresolvermaj` so that strict and majority consensus bipartitions are taken from consensus trees, and not sampled from a random topology. A benefit of using consensus trees is that the consensus resolution no longer needs to be specified; it can be calculated directly from the input tree.

Users can also simulate collections using any of the described algorithms even if consensus trees are not available, assuming consensus resolutions are provided. In this latter case, all algorithms will start from a random starting tree. Our implementation builds upon and extends Sul's and Williams' work, fixing critical bugs that allow the algorithms to efficiently and correctly build arbitrarily large collections of trees.

4 Experimental Methodology

TreeSim is written C++ and is compiled with g++ 4.6.3 with the -03 option, and is accessible from https://github.com/suzannejmatthews/treesim. Experiments were conducted on a 64-bit Intel machine running Ubuntu Linux 12.04 with two quad-core 1.2Ghz processors and 32 GB of RAM.

4.1 Description of Datasets

We experiment on three topologically diverse tree collections obtained from previously published phylogenetic studies:

- `freshwater`: A collection of 20,000 trees over 150 freshwater taxa produced from two runs of Bayesian analysis [15], and consisting of $1,168$ unique bipartitions. The collection has strict and majority consensus resolutions of 34.01% and 85.71% respectively. The strict consensus resolutions for runs 0 and run 1 were 37.42% and 34.69%, while the majority consensus resolutions for runs 0 and 1 were 89.12% and 90.47% respectively.
- `angiosperms`: A collection of 33,306 trees over 567 angiosperm taxa produced from 12 runs of Bayesian analysis [16], and consisting of $2,444$ unique bipartitions. The strict and majority consensus rates for this collection are 51.77% and 92.55% respectively. Each run has distinct strict and majority consensus rates. The strict consensus rates for each run ranges between 61.70% and 67.20%. The majority consensus rates for each run ranges between 91.31% and 95.74%.
- `insects`: An collection of 150,000 trees over 525 insect taxa produced from 5 runs of Parsimony analysis [17]. A total of 573 unique bipartitions exist in this dataset. The collection's strict consensus rate and majority consensus rate are 90.04% and 99.23%, respectively. Each run's consensus rates are identical to the collection as a whole.

4.2 Performance Metrics

Our goal is to produce tree collections that contain similar levels of topological diversity to those outputted by phylogenetic search. To this end, we use popular bipartition-based methods for assessing topological diversity. We first count the average number of unique bipartitions (internal edges) produced in our simulated datasets and compare it to the bipartition counts in our real datasets.

In the next set of experiments, we simulate each run of our datasets using the different consensus resolution algorithms. We split each real tree collection into its respective runs and calculate the majority and strict consensus tree for each run. A $t \times t$ RF distance matrix is produced for each run, where each cell (i, j) in the matrix represents the RF distance between trees T_i and T_j. The RF distance [8] between any pair of trees T_i and T_j is defined as:

$$RF(T_i, T_j) = \frac{|B(T_i) - B(T_j)| + |B(T_j) - B(T_i)|}{2}$$

where $B(T)$ represents the set of bipartitions in tree T.

The normalized RF distance (RF rate) is calculated as $RR(T_i, T_j) = \frac{RF(T_i,T_j)}{n-3}$. An RF distance (RF rate) of 0 indicates that the two trees are identical, while an RF distance of $n-3$ (RF rate of 1) indicates that the two trees are maximally dissimilar.

Simulation Algorithm	Data Sets		
	freshwater	angiosperms	insects
Random	$1,797,952.3$	$12,491,544.0$	$49,159,860.6$
Strict	$496,916.0$	$2,100,514.0$	$125,592$
Majority	$47,461.0$	$221,954.33$	$180,511$
Combined	$18,314.0$	$60,198.0$	$2,521.0$
Real Data	$1,168$	$2,444$	573

TABLE 1
Number of unique bipartitions contained in our simulated collections contrasted with our biological datasets.

Simulation Algorithm	Data Sets		
	freshwater	angiosperms	insects
Random	7.24	53.65	223.49
Strict	9.49	72.35	807.65
Combined	23.48	199.96	$1,077.35$
Majority	26.05	262.44	$3,284.41$

TABLE 2
Time (in seconds) required to simulate our biological datasets.

5 RESULTS

We calculate statistics for our real datasets by using the HashCS [3], HashRF(p,q) [18], and TreeZip [19] software packages. The strict and majority consensus trees returned for each biological dataset are used to seed the simulation algorithms. We confirm that the simulated datasets had consensus resolutions that either matched or closely resembled the inputted parameters. We simulate three collections per algorithm per run and/or dataset of interest. The numbers reported in the remainder of this section represent the average performance of each algorithm over its three simulated collections.

5.1 Bipartition Diversity

In our first experiment, we are imagining the scenario when a single strict and/or majority consensus tree was generated for the entire dataset. Table 1 compares the average number of unique bipartitions in the collections produced by our simulation algorithms contrasted with our real tree collections. Our experiments indicate that combined consensus simulation algorithm produces up to 71.6 (49.81) times less unique bipartitions than the majority (strict) consensus simulation algorithm. Of all the simulation algorithms, the combined consensus produces the number of unique bipartitions closest to the real datasets.

5.2 Topological Diversity

In our second set of experiments, we are imagining a scenario where consensus trees are provided for every run in a dataset. After partitioning each biological collection into their respective runs, we use HashCS to determine the strict and majority consensus trees for each run. We use the consensus trees, number of output trees, and number of taxa to seed the combined, strict and majority consensus tree collection algorithms, producing simulated runs. For every pair of runs in a particular dataset, we compute a $t \times t$ RF distance matrix for the trees between the runs, and compute the average RF rate.

We visualize each dataset as a heatmap in Figure 1 through Figure 3. Each cell (i,j) in the heatmap represents the average RF rate between the trees in run i compared to the trees in run j. The heatmap is colored to highlight areas of similarity, ranging from red to purple. "Hotter" (reddish) cells indicate runs where the trees are closer to each other, while "cooler" (blue/purple) cells indicate runs where the trees are further apart. Each dataset shows very different patterns of similarity, reflecting their diversity. In this set of experiments, our goal is to determine which algorithm best captures the topological characteristics of the real datasets. The random algorithm consistently produced RF rates of over 0.99 for each dataset, yielding heatmaps of pure purple. For brevity, they are not shown here.

For each of our datasets, the combined consensus algorithm produced tree collections with levels of topological diversity that most closely resembled the datasets returned from phylogenetic search. In all cases, the trees in the simulated collections tend to be more diverse than those in the real collections. For example, the angiosperms dataset contains a very diverse set of trees. Overall, the runs are very similar to each other, with average RF rates ranging between 0.099 and 0.184. The trees produced by our combined consensus simulation algorithm produces trees with RF rates ranging between 0.120 to 0.227. The trees contained in the insects dataset contained highly similar trees, with RF rates ranging from 0.035 to 0.037. On this dataset, the combined consensus algorithm simulated a collection containing RF rates that varied between 0.032 and 0.044, reflecting the very high level of similarity of the trees.

The strict and majority consensus algorithms consistently performed poorer than the combined consensus algorithm in these experiments. The performance of the strict consensus simulation algorithm varied with the strict consensus resolution for the collection. For example, in the freshwater dataset, the strict consensus resolution for each run was under 40 percent. This relatively low resolution caused the strict consensus simulation algorithm to do poorly on this dataset. While it did better with the higher strict consensus rates in the angiosperms and insects datasets, the trees produced had a much higher level of topological diversity than the combined consensus. The majority consensus algorithm consistently produced runs with average RF rates that ranged from 0.329 to 0.448.

5.3 Run-time analysis

We perform two run-time studies. In the first, we measure how quickly each of TreeSim's simulation algorithms can generate a collection of t trees over n taxa. One way we imagine TreeSim being used is to simulate

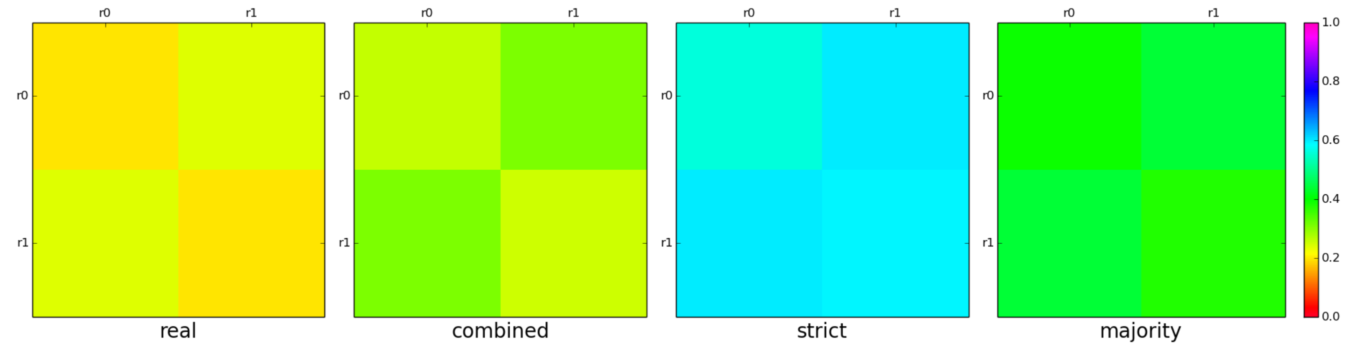

Fig. 1. Comparison of `freshwater` dataset with simulated datasets produced by combined, strict, and majority consensus algorithms (view electronically).

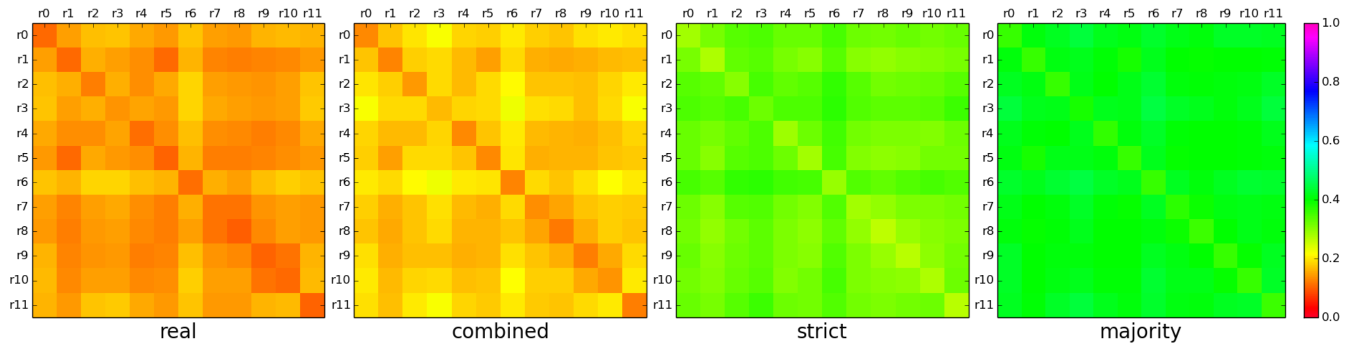

Fig. 2. Comparison of `angiosperms` dataset with simulated datasets produced by combined, strict, and majority consensus algorithms (view electronically).

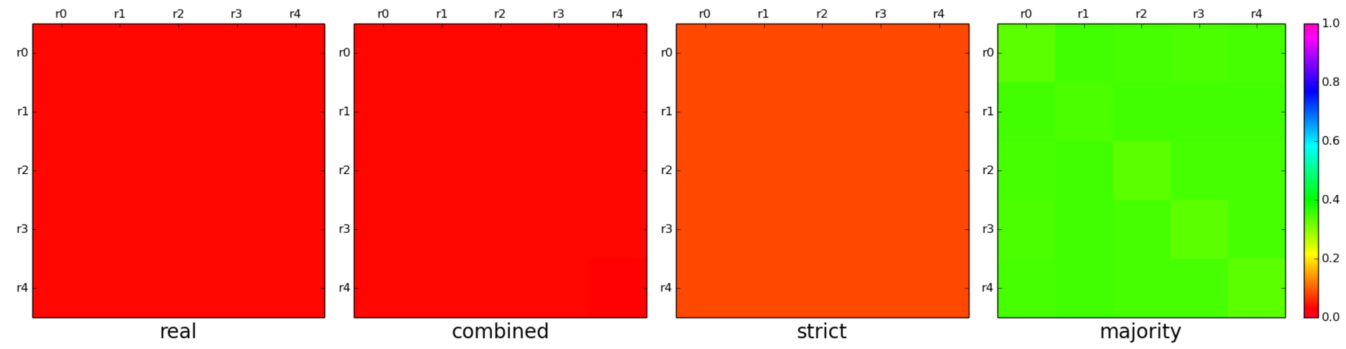

Fig. 3. Comparison of `insects` dataset with simulated datasets produced by combined, strict, and majority consensus algorithms (view electronically).

larger collections of trees based on some existing consensus resolution rates. We study the scalability of TreeSim for this approach in our second set of experiments, where we study how an increase in the number of trees affects the time required to simulate each dataset.

Table 2 shows the amount of time it takes to simulate each of our different biological datasets using the algorithms implemented in TreeSim. It is fastest to simulate a random collection of trees, requiring between 7.24 seconds and 3.62 minutes on average. The majority simulation algorithm typically takes the longest, requiring 54.7 minutes to simulate the `insects` dataset. The combined consensus simulation method in contrast requires only 17.95 minutes to simulate `insects`, as the number of bipartitions in $\mathcal{R} - \mathcal{S}$ for this collection is low.

In our second set of experiments, we artificially increase the number of trees in each real dataset, while keeping all other parameters in the datasets the same. Generating a tree collection in this manner allows a scientist to artificially increase the number of trees in a collection, while maintaining a similar level of topological diversity. Our experimentation shows that a collection of

$100,000$ trees over 567 taxa requires up to 13.21 minutes to compute. Increasing the number of output trees to $500,000$ drastically increases the run-time to 3.86 hours. Note that generating 5 separate runs of $100,000$ trees yields a similarly sized tree collection in only 66 minutes. This suggests that run-time is closely tied to the number of output trees.

6 Conclusions

Scientists lack the habit of publishing tree collections outputted from phylogenetic analysis, creating a data vacuum that hinders the development of methods that can react to the explosive growth of phylogenetic data. The most popular methods for simulating collections generate random trees. Yet, the trees returned from phylogenetic search are rarely random, and are topologically similar to each other.

We develop TreeSim, a novel software package that enables researchers to simulate tree collections based on on available consensus information. TreeSim's key novelty is the *combined consensus* simulation algorithm. Our experimental results strongly indicate that the combined consensus algorithm is the best at emulating the levels of topological diversity found in tree collections returned from phylogenetic search, far outperforming the other consensus simulation algorithms and the random approach. TreeSim's implementation of the majority and strict consensus algorithms effectively serve as a replacement for the `unresolvermaj` and `unresolverstr` packages. Future work will concentrate on improving TreeSim's ability to simulate weighted trees.

TreeSim requires the majority and strict consensus trees along with the number of taxa and trees to be maximally effective. Our experimentation indicates that for best results, each run of phylogenetic analysis should be separately simulated. In the absence of consensus trees, researchers can simulate tree collections based on published consensus rates. Researchers can easily reproduce the collections described here using the consensus trees posted on TreeSim's download page.

In the short term, scientists can use the consensus trees we provide to reproduce our results. In the long term, it would be beneficial for scientists to provide the majority and strict consensus trees for each run of phylogenetic analysis. We are not alone in this call [6], [20], as several other researchers have discussed the limitations of single consensus trees.

We caution however that simulation algorithms can only go so far; in cases where the majority and strict consensus resolutions are very high, it becomes difficult to emulate the topological nuances of a tree collection. Ultimately, there is no substitution for real data, and we as a community must move forward to enforcing stricter data sharing requirements.

Acknowledgments

We are very grateful to S.-J. Sul and T.L. Williams for making the code for `unresolverstr` and `unresolvermaj` available and for encouraging this work. The opinions in this paper are those of the author and do not necessarily reflect the opinions of the U.S. Military Academy, or the U.S. Army.

References

[1] F. Ronquist and J. P. Huelsenbeck, "Mrbayes 3: Bayesian phylogenetic inference under mixed models." *Bioinformatics*, vol. 19, no. 12, pp. 1572–1574, August 2003.

[2] S. J. Matthews, "Efficient algorithms for comparing, storing, and sharing large collections of evolutionary trees," Ph.D. dissertation, Texas A&M University, 2012.

[3] S.-J. Sul, S. J. Matthews, and T. L. Williams, "Using tree diversity to compare phylogenetic heuristics," *BMC Bioinformatics*, vol. 10, no. Suppl 4, p. S3, 2009.

[4] B. T. Drew, R. Gazis, P. Cabezas, K. S. Swithers, and J. Deng, "Lost branches on the tree of life," *PLoS Biol*, vol. 11, no. 9, p. e1001636, 2013.

[5] A. F. Magee, M. R. May, and B. R. Moore, "The dawn of open access to phylogenetic data." *PLoS ONE*, vol. 9, no. 10, 2014.

[6] D. Maddison, "The discovery and importance of multiple islands of most parsimonous trees," *Syst. Bio.*, vol. 42, no. 2, pp. 200–210, 1991.

[7] S.-J. Sul, S. Matthews, and T. L. Williams, "New approaches to compare phylogenetic search heuristics," in *IEEE International Conference on Bioinformatics and Biomedicine (BIBM'08)*, 2008, pp. 239–245.

[8] D. F. Robinson and L. R. Foulds, "Comparison of phylogenetic trees," *Mathematical Biosciences*, vol. 53, no. 1-2, pp. 131–147, 1981.

[9] M. Kuhner and J. Felsenstein, "A simulation comparison of phylogeny algorithms under equal and unequal evolutionary rates." *Molecular Biology and Evolution*, vol. 11, no. 3, pp. 459–468, 1994.

[10] S.-J. Sul and T. L. Williams, "An experimental analysis of consensus tree algorithms for large-scale tree collections," in *Proceedings of the 5th International Symposium on Bioinformatics Research and Applications (ISBRA'09)*. Springer-Verlag, 2009, pp. 100–111.

[11] ——, "unresolverstr," Internet Website, last accessed, March 2015, https://bitbucket.org/sulsj/unresolverstr.

[12] ——, "unresolvermaj," Internet Website, last accessed, March 2015, https://bitbucket.org/sulsj/unresolvermaj.

[13] W. Piel, "TreeBASE: A database of phylogenetic knowledge," Internet Website, last accessed, January 2012, http://www.treebase.org/.

[14] J. Hughes, "Treeripper web application: towards a fully automated optical tree recognition software," *BMC bioinformatics*, vol. 12, no. 1, p. 178, 2011.

[15] L. A. Lewis and P. O. Lewis, "Unearthing the molecular phylodiversity of desert soil green algae (chlorophyta)," *Systematic Biology*, vol. 54, no. 6, pp. 936–947, 2005.

[16] D. E. Soltis, M. A. Gitzendanner, and P. S. Soltis, "A 567-taxon data set for angiosperms: The challenges posed by bayesian analyses of large data sets," *International Journal of Plant Sciences*, vol. 168, no. 2, pp. 137–157, 2007.

[17] A. D. Molin, S. Matthews, S.-J. Sul, J. Munro, J. B. Woolley, J. M. Heraty, and T. L. Williams, "Large data sets, large sets of trees, and how many brains? — Visualization and comparison of phylogenetic hypotheses inferred from rdna in chalcidoidea (hymenoptera)," Entomological Society of America (ESA) Annual Meeting: Student Competition for the Presidents Prize (poster), December 2009.

[18] S.-J. Sul, G. Brammer, and T. L. Williams, "Efficiently computing arbitrarily-sized robinson-foulds distance matrices," in *Workshop on Algorithms in Bioinformatics (WABI'08)*, ser. Lecture Notes in Computer Science, vol. 5251. Springer-Verlag, 2008, pp. 123–134.

[19] S. J. Matthews and T. L. Williams, "An efficient and extensible approach for compressing phylogenetic trees," *BMC Bioinformatics*, vol. 12, no. Suppl 10, p. S16, 2011.

[20] C. Stockham, L. S. Wang, and T. Warnow, "Statistically based postprocessing of phylogenetic analysis by clustering," in *Proceedings of the 10th International Conference on Intelligent Systems for Molecular Biology (ISMB'02)*, 2002, pp. 285–293.