Using Constrained-INC for Large-Scale Gene Tree and Species Tree Estimation

Thien Le, Aaron Sy, Erin K. Molloy, Qiuyi Zhang, Satish Rao, and Tandy Warnow

Abstract—Incremental tree building (INC) is a new phylogeny estimation method that has been proven to be absolute fast converging under standard sequence evolution models. A variant of INC, called Constrained-INC, is designed for use in divide-and-conquer pipelines for phylogeny estimation where a set of species is divided into disjoint subsets, trees are computed on the subsets using a selected base method, and then the subset trees are combined together. We evaluate the accuracy of INC and Constrained-INC for gene tree and species tree estimation on simulated datasets, and compare it to similar pipelines using NJMerge (another method that merges disjoint trees). For gene tree estimation, we find that INC has very poor accuracy in comparison to standard methods, and even Constrained-INC(using maximum likelihood methods to compute constraint trees) does not match the accuracy of the better maximum likelihood methods. Results for species trees are somewhat different, with Constrained-INC coming close to the accuracy of the best species tree estimation methods, while being much faster; furthermore, using Constrained-INC allows species tree estimation methods to scale to large datasets within limited computational resources. Overall, this study exposes the benefits and limitations of divide-and-conquer strategies for large-scale phylogenetic tree estimation.

Index Terms—Phylogeny, gene tree, species tree, maximum likelihood, divide-and-conquer

1 Introduction

The estimation of gene trees and species trees is a basic part of many biological analysis pipelines; gene trees have implications for trait evolution and the prediction of protein function and structure (as well as other applications), while species trees are needed to understand how species adapt to their environments, to date speciation events, etc. The estimation of both gene trees and species trees are based on statistical models of evolution, with gene trees based on a single locus within the genome of the different species, and species trees based on multiple loci.

Gene tree estimation is generally formulated as a statistical inference problem in which the sequences given as input are assumed to have been generated on an unknown (but fixed) model tree, equipped with a stochastic model of evolution. Species tree estimation in turn assumes that the input is a set of multiple sequence alignments (one for each locus), the different loci have gene trees that have evolved within a common species tree, and the sequences for a given locus evolve down the gene tree for that locus. Because gene trees can differ from the species tree due to various biological processes (e.g., gene duplication and loss, horizontal gene transfer, and incomplete lineage sorting), statistically consistent estimation of species trees from multiple loci requires methods that address gene tree heterogeneity [1].

One of the fundamental questions in phylogenetic tree estimation is whether a method is guaranteed to converge to the model tree as the amount of data increases, and methods that have this property are said to be "statistically consistent". Indeed, statistical consistency is widely regarded as a necessary property of any statistical estimator (see discussion in [2], [3], and references therein), and this is also true in phylogenetics. It is now well known that maximum likelihood, distance-based methods such as neighbor joining and balanced minimum evolution (BME), and many other methods are statistically consistent under the standard statistical models of sequence evolution, such as Jukes-Cantor [4] and the Generalised Time Reversible (GTR) models [5].

Although statistical consistency is important, the "sample complexity" (which evaluates the amount of data that a method needs to return the true tree with high probability) is perhaps more important than statistical consistency, since datasets are of finite length. In [6], [7], [8], the concept of "Absolute Fast Convergence" was proposed, which we describe in the context of the Jukes-Cantor [4] site evolution model. A Jukes-Cantor (JC) model tree is a pair (T, Θ), where T is a rooted binary tree and Θ is a set of "branch lengths", where l_e denotes the expected number of times each site will change on the edge e. The nucleotide at the root of the tree is drawn uniformly at random from $\{A, C, T, G\}$, and substitutions on an edge (if they occur) are made to the other nucleotides with equal probability. All the sites in the sequence evolve down this JC model tree identically and independently (*i.i.d.*). We will say that the phylogeny estimation

- *T. Le is with the Department of EECS, Massachusetts Institute of Technology, Cambridge, MA 02139 USA. E-mail: steven.le.thien@gmail.com.*
- *A. Sy is with YouTube, San Bruno, CA 94066 USA. E-mail: raaronsy@berkeley.edu.*
- *Q. Zhang is with Google Brain, Mountain View, CA 94043 USA. E-mail: qiuyizhang@gmail.com.*
- *S. Rao is with the Department of EECS, University of California, Berkeley, Berkeley, CA 94720 USA. E-mail: satishr@cs.berkeley.edu.*
- *E.K. Molloy and T. Warnow are with the Department of Computer Science, University of Illinois at Urbana-Champaign, Champaign, IL 61801 USA. E-mail: {emolloy2, warnow}@illinois.edu.*

Manuscript received 29 Aug. 2019; revised 19 Mar. 2020; accepted 24 Apr. 2020. Date of publication 14 May 2020; date of current version 3 Feb. 2021. (Corresponding author: Tandy Warnow.) Digital Object Identifier no. 10.1109/TCBB.2020.2990867

method Φ is *absolute fast converging (AFC)* for the JC model if there is a polynomial $p(n)$ such that for all positive ϵ, f, g the probability of recovering the unrooted topology of the true tree T given sequences of length $p(n)$ is at least $1-\epsilon$ for all JC model trees (T, Θ) where T has n leaves and all edges e satisfy $f \leq l_e \leq g$, where l_e is defined as before. The first provably AFC methods were presented in [6], [7], and were distance-based methods that operated by computing quartet trees and then combining them. By restricting the quartet trees that were used to a subset (called the "short quartets") of the full set of quartet trees, it was shown that the true tree could be constructed with high probability from polynomial length sequences. Since then, many other methods have been established to be AFC under sequence evolution models, including maximum likelihood [9].

Recently, a new polynomial time method, INC, was presented by Zhang *et al.* [10] and proven to be statistically consistent under GTR. Zhang *et al.* also presented a variant of the method called INC-NJ that uses neighbor joining [11] (NJ) on small subsets, and proved that INC-NJ is also AFC and has low degree polynomial time. Finally, Zhang *et al.* presented a generic technique called constrained-INC that allows the user to provide a set of disjoint constraint trees, and then uses INC to combine the constraint trees into a tree on the full dataset. However, no implementation of INC was developed, and so INC and its variants were not explored with respect to empirical accuracy on data.

This paper reports on an extensive study of INC and its variants, as described in [10], and also explores modifications to INC to improve accuracy. We explore the design space of divide-and-conquer strategies using INC for the purpose of gene tree estimation, and compare the best of these methods to standard phylogeny estimation methods on simulated data. We also evaluate variants of Constrained-INC for multi-locus species tree estimation, where gene trees can differ from the species tree due to incomplete lineage sorting (ILS), an event that can arise when allelic polymorphism is present in a population and persists even after speciation events [1].

We compare the best variants of Constrained-INC to NJMerge [12], another recent method that combines disjoint trees, on a collection of simulated datasets. We find that Constrained-INC and NJMerge have similar accuracy (with a slight advantage to NJMerge for species tree estimation), but Constrained-INC is much faster. We find that maximum likelihood using RAxML version 8.2.12 [13] has the best accuracy for single gene tree estimation, and that the divide-and-conquer approaches tend to reduce accuracy in this case. However, we also find that divide-and-conquer pipelines using both Constrained-INC and NJMerge generally maintain accuracy for the leading species tree estimation methods for multi-locus datasets when gene tree discordance due to ILS is present, while being much faster. Thus, although NJMerge and Constrained-INC are not beneficial for gene tree estimation, they provide advantages for multi-locus species tree estimation.

Our open-source implementation is available on Github [14], along with the commands needed to reproduce the study. All gene tree estimation datasets used in this study are available on the Illinois Data Bank [15]. Species tree estimation datasets are available at [16].

2 Methods

2.1 The Incremental Tree Building (INC) Method

We briefly describe the INC method of Zhang *et al.* [10], and direct the reader to the original paper for details. The input to INC is a set of n sequences S in a multiple sequence alignment $\mathcal{A}$ and an $n \times n$ matrix d computed on the alignment. Given the matrix d, a minimum spanning tree (MST) on the sequences in S is computed, and then the MST is used to define an ordering on the sequences so that each added sequence is adjacent within the spanning tree to exactly one sequence that precedes it in the ordering. The taxa are added into a growing tree t, which begins with the first three taxa in the ordering and then adds each new taxon into t. To add a new taxon q into t, a set of quartet trees is computed, and these quartet trees vote on where to place q into t. The approach in [10] uses the Four Point Method (FPM) [17], [18] to compute quartet trees and then allows only "valid quartets" (which are quartets with sufficiently low interleaf distances, according to the input matrix d) to vote. Furthermore, all quartet trees have the same voting power (i.e., each vote has the same weight). When all the taxa have been inserted, the final tree is returned. Because of the incremental nature of the approach, the algorithm is called Incremental Tree Building, or INC.

2.2 Disjoint Tree Merger (DTM) Methods

Zhang *et al.* [10] showed that INC can be used to combine a given set of disjoint trees (i.e., constraint trees). This approach, called Constrained-INC, takes as input a set of leaf-disjoint trees, and merges the trees together, using an auxiliary distance matrix. Specifically, Constrained-INC operates using nearly the same incremental taxon-addition technique used in INC, but it makes sure never to add a taxon into the growing tree in such a way that would violate any of the input constraint trees.

NJMerge [12], TreeMerge [19], and Guide Tree Merger (GTM) [20], are other methods that are designed to construct a tree from a set of leaf-disjoint constraint trees. Each uses auxiliary information, with NJMerge and TreeMerge using a distance matrix computed on the full set of species and GTM using a guide tree computed on the full set of species. These three methods have been tested only for multi-locus species tree estimation, where the three methods displayed similar accuracy (with a slight advantage of NJMerge over TreeMerge [19]). Because these methods combine disjoint trees, they are referred to as "Disjoint Tree Merger" (DTM) methods.

2.3 Divide-and-Conquer Pipelines

DTM methods can be used within divide-and-conquer pipelines for tree construction: the input taxon set is divided into disjoint subsets, trees are constructed on subsets (using a selected tree estimation method), and then the subset trees are combined using the DTM method into a tree on the full dataset.

In [12], [19], [20], the following divide-and-conquer pipeline was used for multi-locus species tree estimation. First, a starting tree is computed from the multi-locus dataset, so that each leaf in the starting tree corresponds to a species in the input set. The starting tree can be computed from a distance matrix computed on the multi-locus dataset or using

some other technique. We then decompose the species set into subsets using the recursive centroid edge decomposition strategy (see SATé-II [21] and PASTA [22]) that operates by by pulling out a "centroid edge" (i.e., an edge that splits the leafset into two sets of roughly equal size), and then recursing until all the subsets are small enough. The leafsets for the different subtrees then define the subsets of species on which a constraint tree will be computed. A selected base method is then used to construct a species tree for each subset of species, using all the available data for that species set, and the set of these constraint trees is then passed to the DTM method (NJMerge, TreeMerge, or GTM), which also computes the auxiliary information it needs directly from the input data.

2.4 Statistical Consistency

Here we discuss statistical consistency of the methods we examine under different evolutionary models, focusing on the estimation of the unrooted true tree.

Recall that the GTR site evolution model describes how a single site evolves down a model tree, and is extended to sequence evolution by assuming that all the sites evolve identically and independently (*i.i.d.*). Similarly, the MSC +GTR model [23] describes how a multi-locus dataset evolves: there is a rooted binary species tree T with branch lengths (representing time in coalescent units), gene trees evolve down the species tree under the multi-species coalescent (MSC) model, and then sequences evolve down each gene tree under the GTR model.

For a given model of evolution Φ (i.e., Φ could be the GTR sequence evolution model or the MSC+GTR hierarchical model) and tree estimation method M, we will say that M is "statistically consistent under Φ" if, for all model trees (T, Θ) under Φ, the probability that M returns the unrooted version of T converges to 1 as the amount of data increases.

For GTR and other sequence evolution models, by saying "as the amount of data increases" we mean the number of sites increase, while for MSC+GTR we mean that the number of sites and the number of genes both increase.

2.4.1 Statistical Consistency Under the GTR Model

Maximum likelihood (ML) is statistically consistent under the GTR model (and all its submodels) if solved exactly [24]. Since ML is NP-hard [25], heuristics, such as RAxML [13], IQtree [26], PhyML [27], and FastTree2 [28], are used to search for local optima. Neighbor joining [11], balanced minimum evolution, and other distance-based methods are also statistically consistent under the GTR model if used with properly corrected distances, such as the log-det distance [29]. Specifically, for each of these methods, if they are given a dissimilarity matrix as input that is close enough under the L_∞ metric to an additive matrix for the model tree, then they are guaranteed to return the model tree topology (see [30] for the theorem regarding Neighbor Joining and [31] for a general discussion).

2.4.2 Statistical Consistency Under the MSC+GTR Model

Many methods are statistically consistent under the MSC +GTR model. For example, ASTRAL [32], [33], ASTRID [34], and several other summary methods (i.e., methods that combine gene trees to estimate the species tree) are statistically consistent. Methods that estimate the species tree from site patterns in the concatenated alignment, such as SVDquartets [35], SVDquest [36], and SNAPP [37], are also statistically consistent. Finally, some methods, such as StarBEAST [38], co-estimate gene trees and species trees from the multi-locus dataset, and are also statistically consistent. However, standard maximum likelihood analyses applied to the concatenated alignments (whether partitioned [39] or unpartitioned [23]) are not statistically consistent.

2.5 Our Experimental Study

2.5.1 Overview

We performed a sequence of experiments to evaluate methods for gene tree and species tree estimation. Our first experiments evaluated INC and INC-NJ for gene tree estimation. Our next experiments explored the design space of divide-and-conquer strategies using Constrained-INC for gene tree experiment. After this, we compared the best variants of Constrained-INC to standard gene tree estimation methods, and to NJMerge using the same divide-and-conquer strategies. Then we turned to evaluating Constrained-INC for multi-locus species tree estimation, where gene trees can differ from the species tree due to incomplete lineage sorting. In summary, we have the following experiments:

- Experiment 0: Evaluating INC and INC-NJ for gene tree estimation
- Experiment 1: Exploring the design space for Constrained-INC for gene tree estimation using divide-and-conquer
- Experiment 2: Comparing the best variants of Constrained-INC to leading gene tree estimation methods
- Experiment 3: Comparing the best variants of Constrained-INC to NJMerge for gene tree estimation in divide-and-conquer strategies
- Experiment 4: Evaluating Constrained-INC for multi-locus species tree estimation in divide-and-conquer strategies

Each experiment uses simulated datasets, some of which are for single genes and others which are for multi-locus datasets where gene trees differ from the species tree due to ILS.

The Robinson-Foulds (RF) error rate is the most commonly used method for evaluating phylogeny estimation methods, as it reflects the proportion of the monophyletic groups (i.e., clades) in the true tree that are not recovered in the estimated tree. However, when the true tree or the estimated tree is not fully resolved, then the RF error rate is inappropriate, for reasons that we describe below. Furthermore, for such cases, the use of the False Negative (FN) error rate is preferable. We define both these metrics here, and explain why we focus on the FN error rate.

Every edge e in a tree T with leafset S defines a bipartition π_e on S, produced by deleting the edge (but not its endpoints) from T. Furthermore, each tree T is uniquely identified by its set of bipartitions, which is denoted by $C(T) = \{\pi_e : e \in E(T)\}$. Then, the Robinson-Foulds distance, also referred to as the bipartition distance, between

two trees T and T', both on the same leafset, is given by $|C(T) \triangle C(T')|$ (i.e., the symmetric difference between the sets of bipartitions). Furthermore, given the model tree T and an estimated tree $\hat{T}$, the RF error rate is the RF distance between T and $\hat{T}$ divided by $2(n-3)$, where n is the number of leaves in the trees. However, in some cases the estimated tree may not be fully resolved, which makes the RF distance inappropriate. Another case where the RF distance can be inappropriate is where the model tree is either not binary or may has zero-event branches (meaning, branches on which no changes occur), so that recovering those branches is not possible. The "potentially inferrable model tree" (PIMT) is the model tree with all zero-event branches collapsed. In this study, our estimated trees in general are generally fully resolved, but some of the PIMTs may differ (in a given replicate) from the model tree (due to very short branches). For these reasons, instead of reporting the RF error rate with respect to the model tree, we report the missing branch rate, also referred to as the false negative rate, with respect to the PIMT: this is the fraction of bipartitions that appear in the PIMT that do not appear in the estimated tree. Note that when the PIMT and estimated tree are both fully resolved, then the FN error and the RF error are identical.

We note that the FN error rate can be inappropriately high when the dataset includes rogue taxa (i.e., taxa that can be attached to the tree in many different locations, which can occur when the taxon is attached to the tree by a very long branch). For such conditions, other metrics, such as the quartet tree error rate (i.e., the normalized quartet distance, which is the proportion of sets of four leaves on which the two trees induce different quartet trees) are more suitable than the RF error rate. Although our experimental conditions do not include rogue taxa, for the sake of an alternative view, we also examine quartet tree error rates, using software described in [40], for Experiments 1 and 4.

Finally, we also record the running time used to estimate trees. All analyses for all methods and datasets were performed on Blue Waters (a supercomputer at the National Center for Supercomputing Applications) and were limited to 48 hours.

2.5.2 Gene Tree Datasets for Experiments 0-3

We analyzed several collections of aligned sequence datasets under a range of model conditions so that they varied in terms of difficulty. We explored datasets that evolved under model trees with a strict molecular clock (i.e., the expected number of changes is proportional to time, so that the model tree is ultrametric) and others where the evolutionary process deviates from the strict molecular clock. We also varied the number of sequences, sequence length, substitution model, branch lengths, whether or not the sequence evolution model has insertions and deletions ("indels"), and overall rate of evolution. This last issue (overall rate of evolution) is reflected in the average and maximum p-distances (i.e., normalized Hamming distances, which is the proportion of sites in which two aligned sequences differ, see Table 1), with the most difficult datasets being the ones with the largest average and maximum p-distances.

Furthermore, three of the model conditions have many very short internal branches, making them difficult to infer correctly. Some of the model trees come from prior publications and others were generated by us; see [14], [15] for more information. In all cases, we used the true alignment.

TABLE 1
Empirical Statistical (average and maximum p-distances, i.e., normalized Hamming distances, max = 1.0) of the Simulated Gene Tree Datasets for Experiments 0-3

Datasets	1000L1	101	1000(SB)	10K
Avg. p-distance	0.70	0.13	0.21	0.19
Max p-distance	0.77	0.33	0.32	0.30

101-Taxon Datasets. The model tree has very short internal branches, reflecting a rapid radiation, and evolve under the GTR model without indels (20 replicates). These are not ultrametric.

1000L1 Datasets. We use the true alignments from 20 replicates from the 1000L1 datasets studied in [22]. These datasets have a high rate of evolution with both substitutions and long indels, with average gap length 13.6, and do not have short branches. The model tree is not ultrametric. The average percentage of the true alignment that is gapped (i.e., gappiness) is 73.2 percent.

1000-Taxon (SB) Datasets. The model gene trees for these datasets have very short branches (SB), representing a rapid radiation. The model trees, which are not ultrametric, were studied in [12]. 20 replicate datasets were simulated with INDELible [41] with the same model parameters as in [12].

10K. This model tree has 10,000 leaves and is ultrametric (i.e., the sequence evolution is under a strict molecular clock) with many short branches. We use 10 replicate sequence datasets that evolved down the tree using INDELible.

2.5.3 Multi-Locus Species Tree Datasets for Experiment 4

We use multi-locus datasets from [12], which have 1,000 species and 1,000 exon-like genes where true gene trees differ from the true species tree due to ILS. Each of these datasets was analyzed using the same basic divide-and-conquer strategy we describe here, but with NJMerge used to combine the constraint trees rather than INC.

The datasets we explore have two levels of ILS, which we describe in terms of the average normalized Robinson-Foulds distance (AD) between true gene trees and true species trees: moderate ILS with average AD of 10 percent and very high ILS with average AD of 69 percent. In both cases, the speciation is close to the root. Each of the two model conditions has 20 replicates. The moderate ILS datasets have average gene tree estimation error (GTEE), measured using Robinson-Foulds, of roughly 42 percent, and the very high ILS datasets have GTEE of roughly 64 percent. These GTEE rates are likely representative of phylogenomic studies, where loci from across the genomes of the organisms are used so that many of the loci have low phylogenetic signal, as reflected in the bootstrap support of the estimated gene trees. For example, the Avian Phylogenomics project that estimated species trees on 48 avian species reported that the average bootstrap support of the gene trees was only about 25 percent [42]. Similar trends are shown in general for empirical phylogenomic datasets (see Table 1 in [43]).

As mentioned above, [12] explored the same basic divide-and-conquer strategy, but used NJMerge to combine the constraint trees. To enable a direct comparison between INC and NJmerge, we use the same constraint trees produced in [12], which were computed using either ASTRAL or RAxML, as follows (see [12] for full details). The constraint subsets were produced by the recursive centroid edge decomposition strategy applied to a specified starting tree, recursing until each subset had at most 120 taxa. To compute the RAxML constraint trees on a given species subset, the gene sequence alignments for each gene (restricted to that specified set of species) were concatenated into one large alignment, and then a tree was computed on the concatenated alignment in an unpartitioned GTRGAMMA analysis using RAxML. To compute the ASTRAL constraint trees on a given subset of species, [12] first used FastTree2 to compute the gene trees using all the species; then, for the given subset of species, the gene trees were restricted to the subset and ASTRAL was then used to compute a species tree from the set of gene trees restricted to that specific subset.

3 Results

3.1 Results for Experiment 0: Evaluating INC and INC-NJ for Gene Tree Estimation

INC and INC-NJ are both AFC under the GTR model. Maximum likelihood, if solved exactly, is also absolute fast converging [9], but it is unlikely that heuristics for ML are AFC. We compare INC and INC-NJ to two maximum likelihood heuristics (FastTree2 version 2.1.10 [28] and RAxML version 8.2.12 [13]) and two distance-based methods (BME, balanced minimum evolution, within FastME version 2.1.5 [44] and Neighbor Joining within PAUP* version 4.0a163 [45]). We explore their relative performance on 20 datasets with 1,000 sequences that evolve under a high rate of evolution (the 1000L1 model from [22]).

FastTree2 and RAxML are run with default settings under the GTR model of sequence evolution [5]. The distance-based method BME is run within FastME2 [44] using NNI and SPR moves and NJ is run in default settings within PAUP*; both distance-based methods are given logdet distance matrices computed by PAUP*.

Due to the upper limit of 48 hrs for all methods and all datasets, some RAxML analyses did not complete within that time; we use checkpointing and report the best ML solution found within the 48 hour time limit in such cases. Even so, some RAxmL analyses failed to return any trees at all within that time period.

INC and INC-NJ both have very high error rates of 91 and 70.7 percent, respectively, and the other methods have much lower error rates (Table 2). The best accuracy is obtained using the two maximum likelihood heuristics (error rates under 12 percent), and the two distance-based methods have moderate error rates (43.4 percent for NJ and 30.7 percent for FastME). Thus, INC is much less accurate than INC-NJ and INC-NJ is much less accurate than NJ. Since NJ is *not* AFC [46] and both INC and INC-NJ are AFC, this result is disappointing (to say the least). However, this experiment does not address whether using ML heuristics to compute constraint trees would result in improved accuracy compared to other methods. Exploring this question is the purpose of the next section.

TABLE 2
Results for Experiment 0: False Negative (FN) Tree Error Rates for INC, INC-NJ, and Standard Methods on 20 Replicates of the 1000L1 Model Condition

Method	FN Tree Error Rate
INC	0.910
INC-NJ	0.707
FastTree2	0.109
NJ	0.434
FastME	0.307
RAxML	0.117

3.2 Results for Experiment 1: Exploring the Design Space for Constrained-INC for Gene Tree Estimation

This experiment explores the design space of constrained-INC on 20 replicates of the 1000L1 datasets, evaluating the impact of the following algorithmic parameters: (a) initial tree and decomposition size, (b) how constraint trees are computed, (c) how quartet trees (for merging constraint trees) are computed, and (d) the weight of the votes (i.e., identical weights for all quartets, or weights that depend on the specific quartet tree).

In our first experiment, we compare divide-and-conquer strategies that use INC. We use the centroid edge decomposition (but changing the target subset size) to define the subsets, construct trees on the subsets using different techniques, and then combine the subset (constraint) trees using different ways of running constrained-INC, including specific changes to its algorithm design. We explore these variants on the 20 replicates of the 1000L1 datasets.

Unless specified otherwise, we use the following settings for all methods: INC-ML uses FastTree2 as a starting tree, uses a centroid edge decomposition, constructs the constraint trees using FastTree2 under the GTR model, and employs unweighted voting restricted to the valid quartets. Neighbor joining is run using PAUP*, BME is run using FastME2 [44] with NNI and SPR searches, and FastTree2 and RAxML are run in default mode under the GTR model.

3.2.1 Impact of Subset Size

We began by evaluating the impact of size of the subsets produced by the centroid edge decomposition strategy, using FastTree2 to compute the constraint trees on the disjoint subsets. As shown in Table 3, the largest subsets produce the best accuracy for both FN and quartet tree error rates. Since the largest subsets also have the highest running time, we explore results using maximum subset size of 200 for the remaining studies in Experiment 1.

3.2.2 Impact of the Choice of the Initial Tree

Next we study the impact of the choice of the initial tree. The result of this experiment (Table 4) shows that using FastTree2 to compute the initial tree gives the overall best

TABLE 3
Results for Experiment 1: The Impact of Maximum Subset Size on the False Negative (FN) and Quartet Tree Error Rates (max. 1.0) for INC-ML

Maximum subset size	FN error	Quartet tree error
20	0.70	0.61
50	0.50	0.56
100	0.37	0.49
200	0.26	0.36
500	0.17	0.24

Results shown are mean error rates over 20 replicates of the 1000L1 model condition.

accuracy; hence, we use FastTree2 for the initial tree in our remaining experiments.

3.2.3 Impact of Voting Scheme

The original voting scheme, presented in [10], allows only valid quartets to vote, and each vote has the same weight. Given the new taxon q to add and growing tree t_g, each internal node i defines a quartet $Q_i = \{u_1, u_2, u_3, q\}$ and all valid quartets (as defined in [10]) have unit weight. Furthermore, given a valid quartet tree $u_1u_2|u_3q$, this identifies a subtree of the growing tree into which q can be added. The quartet tree adds one vote to each edge in that subtree. The implementation of this voting scheme is performed with a straightforward breadth-first-search [10].

We modify this strategy by changing which quartets are allowed to vote and redefining the weight of their votes; we also consider schemes that have two phases. Since increases in the diameter of a quartet (which is defined by the distance matrix d) are known to increase the error rate in the estimated quartet tree [18], we consider weighting schemes that depend on the diameter of the quartet. Overall, we evaluate the following five protocols.

Voting Protocol 1 (VP1). The valid quartets are used without weights to identify a set of edges that have the maximum total support. If there is more than one edge with maximum total support, the first edge encountered with that maximum support is selected.

Voting Protocol 2 (VP2). This protocol uses two phases. The first phase is as with the first protocol in that the valid quartets without weights vote to identify a set E^* of edges that have the maximum total support. Then, the valid quartets are used with weights to select from among the set E^* of identified edges. If there is still a tie, the first edge encountered with that maximum support is selected. For a quartet Q, with diameter (maximum distance between any two leaves) d_Q, we define its weight as either $\frac{1}{d_Q}$ (VP2.1) or $\frac{1}{d_Q^2}$ (VP2.2).

Voting Protocol 3 (VP3). This protocol uses one phase. All valid quartets vote with weights (see below), and the set of edges that have maximum total support is identified. If there is a tie, the first edge encountered with that maximum support is selected. For a quartet Q with diameter d_Q, we define its weight as $\frac{1}{d_Q^2}$.

Voting Protocol 4 (VP4). This protocol uses two phases. In the first phase, all valid quartets vote with weights (see below), and the set E^* of edges that have maximum total support is identified. If there is a tie, then all quartets are allowed to vote (but only on the set E^*) with weights; this produces a subset of E^* that has the maximum total support. If there is still a tie, then the first edge encountered with that maximum support is selected. For a quartet Q with diameter d_Q, we define its weight as $\frac{1}{d_Q^2}$ in both rounds.

Voting Protocol 5 (VP5). This protocol uses one phase. All quartets vote with weights (see below) and the set of edges that have maximum total support is identified. If there is a tie, the first edge encountered with that maximum support is selected. For a quartet Q with diameter d_Q, we define its weight as $\frac{1}{d_Q^2}$.

As shown in Table 5, VP3-VP5 produced slightly better accuracy than VP1, VP2.1, and VP2.2. Of the three better voting schemes, VP3 and VP5 have the advantage of using only one phase, and VP3 has a (slight) running time advantage over VP5 in that it only allows valid quartets to vote. Therefore, in subsequent analyses we used VP3.

TABLE 4
Results for Experiment 1: The Impact of the Starting Tree on False Negative (FN) Error and Quartet Tree Error Rates (max. 1.0) for INC-ML (max 1.0); Results Shown are Averages Over 20 Replicates of the 1000L1 Model Condition

Starting Tree	FN error	Quartet tree error
FastTree2	0.261	0.361
FastME (BME)	0.272	0.428
NJ	0.277	0.447

TABLE 5
Results for Experiment 1: The Impact of Voting Schemes on Tree Error Rates (max. 1.0, means over 20 replicates) for INC-ML on the 1000L1 Datasets

Voting scheme	FN error	Quartet tree error
VP1	0.266	0.361
VP2.1	0.269	0.364
VP2.2	0.269	0.364
VP3	0.249	0.351
VP4	0.249	0.353
VP5	0.249	0.351

3.2.4 Impact of How Constraint Trees are Computed

We examine three ways of computing constraint trees: RAxML used to estimate the constraint tree for each subset, FastTree2 used to estimate the constraint tree for each subset, and the induced tree on the specified subset of the FastTree2 tree on the full set of taxa. Table 6 shows that using the induced trees on each subset using FPM produces poor results for FN error compared to the two ML methods (FastTree2-induced and RAxML), and using either of the two ML methods produces very similar FN error rates. In addition, although using RAxML for quartet trees produces lower quartet tree error, the differences in terms of quartet tree error between the two ML methods are small. Since RAxML as used here (once for every four leaves) is more computationally

TABLE 6
Results for Experiment 1: The Impact on Tree Error Rates (max. 1.0) of How Constraint Trees are Computed: RAxML, FastTree2, or the Induced Subtree of the FastTree2 Tree on the Full Set of Taxa (i.e., "FastTree2-induced")

Constraint tree method	FN error	Quartet error
FastTree2	0.249	0.351
RAxML	0.255	0.361
FastTree2-induced	0.247	0.350

Results shown are averages over 20 replicate datasets from the 1000L1 model condition.

expensive than using FastTree2-induced, we use FastTree2-induced approach in our subsequent studies.

3.2.5 Impact of How Quartet Trees are Computed

We then explored how the quartet trees are computed. We explored three techniques: the Four Point Method (FPM, the default in [10]), RAxML on each quartet, and using the induced quartet tree from the FastTree2 tree on the full dataset. The best result was obtained using the induced tree from FastTree2 (Table 7). Hence, for subsequent experiments, we compute quartet trees by restricting the FastTree2 tree to each quartet.

3.2.6 Summary of Experiment 1

Experiment 1 showed that changes to the Constrained-INC design could result in improved accuracy, with some algorithmic parameters having large impacts. In particular: the size of the constraint trees was important (with larger subsets better), how quartet trees are computed was important (with the Four Point Method much less accurate than FastTree2-induced quartet trees), and other parameters providing a small improvement.

3.3 Results for Experiment 2: Evaluating Two Variants of INC-ML

In Experiment 2, we maintained the settings selected in Experiment 1. Since we use ML heuristics to compute the constraint trees, we refer to these variants as INC-ML, and we explore two variants of INC-ML: one (INC-ML (fast)) that is designed for speed and the other (INC-ML (slow)) that is slower and designed for improved accuracy. Both variants use the same divide-and-conquer strategy, differing only in the ML heuristic they use to construct trees on subsets (RAxML for the slow variant and FastTree2 for the fast variant). Each method uses FastTree2 to compute an initial tree, divides the dataset into subsets with at most $n/5$ taxa (where n is the number of taxa in the input set) using the centroid edge decomposition, and constructs ML trees on each subset. Each method uses induced quartet trees from the FastTree2 starting tree for the quartet trees (only on the valid quartets) and voting scheme VP3.

TABLE 7
Results for Experiment 1: Impact of Quartet Tree Methods on Average Tree Error Rates (max. 1.0, across 20 replicates) INC-ML for the 1000L1 Datasets

Quartet Tree Method	FN error	Quartet tree error
FPM	0.247	0.350
FastTree2-induced	0.109	0.359
RAxML	0.137	0.335

TABLE 8
Results for Experiment 2: FN Error Rates of Two Variants of INC-ML, Two ML Heuristics, and NJ Under Different Model Conditions (over 10 replicates for the 10K-taxon condition and 20 replicates for each other model conditions)

Datasets	101	1000L1	1000(SB)	10K
INC-ML (Fast)	0.453	0.266	0.170	0.130
INC-ML (Slow)	0.453	0.253	0.212	0.155
FastTree2	0.453	0.109	0.180	0.131
NJ	0.500	0.434	0.421	0.271
RAxML	0.375	0.117	0.123	0.096

FN tree error rates under the basic model conditions are shown in Table 8, and results as we scale the branch lengths in each model tree for the 10K-taxon datasets (by multiplying each branch length by the selected scaling factor) are shown in Fig. 1. While results on individual model conditions can vary, the overall most accurate results are obtained by RAxML, with FastTree2 in second place, then the two versions of INC-ML, and finally NJ. We also note that error rates increase for all methods as the scaling factor increases. Interestingly, Constrained-INC with FastTree to compute constraint trees is about as accurate as FastTree2 in most cases (and is less accurate for one model condition), while Constrained-INC with RAxML to compute constraint trees is always less accurate than RAxML (Table 8). Thus, the impact on accuracy of using Constrained-INC is generally neutral for FastTree2 and detrimental for RAxML, and so depends on which ML heuristic is used to compute constraint trees.

Runtimes are shown in Table 9. Unsurprisingly, NJ is the fastest of the methods, completing in less time than the other methods, and FastTree2 is the next fastest method. The slowest method by far is RAxML, which does not complete on any of the 10K datasets within 48 hours. Finally, as expected, INC-ML(slow) is slower than INC-ML(fast).

3.4 Results for Experiment 3: Comparison Between Constrained-INC and NJMerge for Gene Tree Estimation

In this experiment, we compare Constrained-INC and NJMerge in the context of gene tree estimation, given the

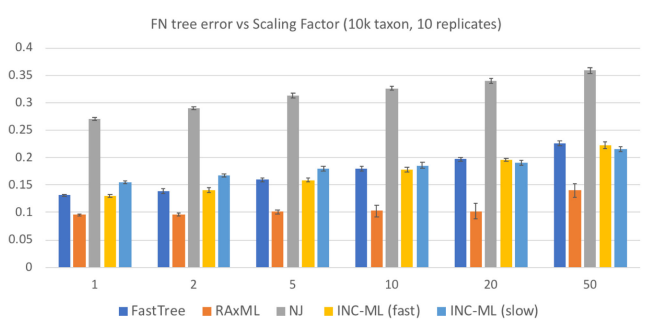

Fig. 1. Results for Experiment 2: FN error rates on the 10K datasets for two variants of INC-ML, two ML heuristics, and NJ, as the model tree branch lengths are scaled from a low rate of evolution (scaling factor 1) to a high rate of evolution (scaling factor 50).

TABLE 9
Results for Experiment 2: Average Runtime (seconds) of INC-ML (both variants) and Standard Methods on Different Datasets

Datasets	101	1000L1	1000(SB)	10K
INC-ML(fast)	26	376	182	4182
INC-ML(slow)	29	1121	750	48385
FastTree2	7	233	75	4071
NJ	0	2	3	2212
RAxML	32	4187	2827	(*)172800

Results shown here are for 10 replicates of 10K model condition and 20 replicates on the other model conditions. The asterisk (*) for RAxML on the 10K model condition reflects that we used checkpointing to return a tree for RAxML (at the end of 48 hours), as it could not complete on any of the 10 replicates within 48 hrs.

same set of constraint trees (each computed using RAxML). As shown in Table 10, although error rates increase for both methods with the scaling factor, the two methods are not distinguishable across any of these scaling factors.

3.5 Results for Experiment 4: Evaluating Constrained-INC for Species Tree Estimation

We now examine the impact of Constrained-INC for use in species tree estimation from multi-locus datasets, where gene trees can differ from the species tree due to ILS. For this experiment, we use multi-locus datasets with 1,000 species and 1,000 exon-like genes from [12] (see Section 2.5.3).

We examine variants of Constrained-INC on these datasets, varying some of the algorithmic parameters, but always using the ASTRID distance matrix (i.e., the average internode distance matrix used in both NJst and ASTRID to compute the species tree), as this converges to an additive distance matrix for the species tree as the number of genes increases [47]. For voting protocol, we use VP3 (which we selected from the earlier experiments), setting d to be the quartet tree diameter computed using the ASTRID/NJst distance matrix (i.e., the average internode distance matrix).

We vary the remaining two algorithmic parameters: how we compute constraint trees and how we compute quartet trees. For constraint trees, we use ASTRAL and RAxML, the two methods that had the best accuracy for constraint trees in [12], each applied to the full set of loci for the specified subset of species. For quartet trees, we use either the Four Point Method or trees induced on guide trees computed using either NJst or ASTRID (i.e., using BME or NJ within FastME on the internode distance matrix).

We compare these variants of Constrained-INC to other species tree estimation methods (ASTRAL, ASTRID, NJst, and RAxML) with respect to topological accuracy on two different model conditions: one with very high ILS and one with low/moderate ILS, each with 1,000 species and 1,000 genes.

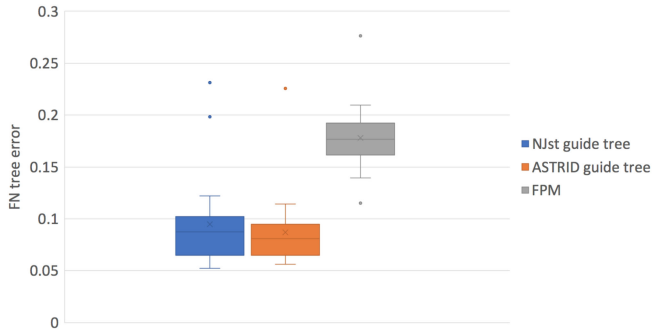

Fig. 2. Results for Experiment 4: Impact of quartet tree method for Constrained-INC for species tree estimation on 1,000-species 1,000-gene datasets with very high ILS (20 replicates). Results are shown for three variants of Constrained-INC, differing only in the method used to compute quartet trees.

TABLE 11
Results for Experiment 4: The Impact of the Quartet Tree Method on the Quartet Tree Error Rate (max 1.0) for Constrained-INC; Results Shown are Averages Over 20 Replicates of the 1000-Taxon 1000-Gene High ILS Datasets

Quartet Tree Method	Quartet tree error
Induced from full NJst tree	0.156
Induced from full ASTRID tree	0.106
FPM	0.429

3.5.1 *Impact of Quartet Tree Method*

Using the ASTRAL constraint trees, we now vary the way quartet trees are computed. To compute quartet trees, we compared the Four Point Method to the induced subtree from either the NJst or ASTRID trees. Results (Fig. 2 and Table 11) under the high ILS condition indicate that using quartet trees induced by the ASTRID guide trees gives the best accuracy, and that the FPM has the highest error. We choose quartet trees induced in the ASTRID guide tree for both low and high ILS conditions, noting however that other techniques (such as a fast maximum likelihood heuristic) might provide better results, but would be more computationally intensive.

3.5.2 *Comparison Between Constrained-INC, NJMerge, and Base Methods for Species Tree Estimation*

We compare Constrained-INC to NJMerge on the same set of constraint trees under both low/moderate ILS and high ILS conditions, each with 1,000 species and 1,000 exons. For both conditions, we use the ASTRID distance matrix and guide tree (to compute quartet trees) as input to Constrained-INC.

TABLE 10
Results for Experiment 3: FN Tree Error Rates on the 1000-Taxon Short Branch Datasets for Constrained-INC and NJMerge on RAxML Constraint Trees, as a Function of the Scaling Factor Applied to the Branch Lengths

Scaling factor	0.2	0.5	1	2	5	10	20	50
Tree error rate for Constrained-INC	0.16	0.18	0.21	0.19	0.20	0.22	0.23	0.26
Tree error rate for NJMerge	0.16	0.18	0.19	0.20	0.20	0.22	0.23	0.26

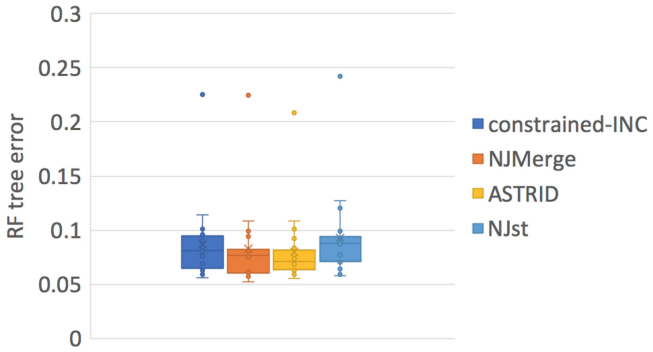

Fig. 3. Results for Experiment 4: Average species tree FN error rates for multi-locus species tree estimation on 1000-species 1000-gene datasets with very high ILS (over 20 replicate datasets). NJMerge and Constrained-INC are given the same set of ASTRAL constraint trees, and Constrained-INC is given the ASTRID distance matrix and induced quartet trees. Results for ASTRAL are not shown, because it only completed on one replicate.

We also explore other species tree estimation methods, which we run on the full datasets.

For the high ILS datasets, we use ASTRAL to compute the constraint trees based on our earlier experiment. ASTRAL only completed within the 48 hour allowed time on one replicate of these high ILS datasets. Fig. 3 shows FN error rates for all the methods other than ASTRAL and Fig. 4 shows result for the single replicate that ASTRAL completes. Results on the single replicate where ASTRAL completes show that ASTRAL has slightly higher error than Constrained-INC, NJMerge, and ASTRID, but is more accurate than NJst. Results on the full set of replicates (Fig. 3) shows that Constrained-INC is slightly less accurate than NJMerge and ASTRID on these high ILS datasets, and matches NJst. Differences between the methods are very small in both cases. Similar results are shown for quartet distances (see Table 12).

For the low/moderate ILS datasets, we use RAxML to compute the constraint trees, since this is the constraint tree method found in [12] to produce the best results for NJMerge under low/moderate ILS conditions. RAxML failed to return any trees on 3 of the 20 replicates within the allowed 48 hours. Fig. 5 shows FN error rates for all replicates (without RAxML) and Fig. 6 shows FN error rates for the 17 replicates for which RAxML returns a tree. The

TABLE 12
Results for Experiment 4: Average Quartet Tree Error Rate (max. 1.0) for Multi-Locus Species Tree Estimation on 1,000-Species 1,000-Gene Datasets With Very High ILS (over 20 replicate datasets)

Species Tree Method	ASTRAL complete	Remaining replicates
Constrained-INC	0.079	0.107
NJMerge	0.078	0.115
ASTRAL	0.066	n/a
ASTRID	0.079	0.107
NJst	0.079	0.125

The single replicate that ASTRAL completed on is shown separately from the remaining 19 replicates.

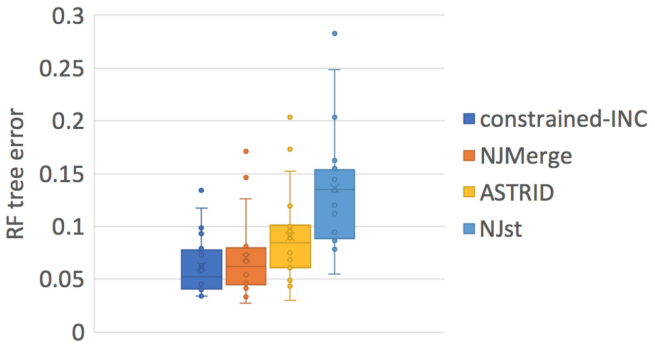

Fig. 5. Results for Experiment 4: Average FN error rates on 1,000-species 1,000-gene datasets with low/moderate ILS, without RAxML (results averaged across all 20 replicates). Constrained-INC and NJMerge both use the same set of RAxML constraint trees and constrained-INC uses the ASTRID distance matrix and the ASTRID guide tree to derive quartet trees.

comparison between Constrained-INC and NJMerge across all 20 replicates show nearly identical accuracy (with a slight advantage to Constrained-INC), and both are more accurate than ASTRID (in third place) and NJst (in last place). A comparison on the 17 replicates where we report results for RAxML show slightly different trends: ASTRID and NJst are still in third and fourth places, respectively, but now NJMerge is slightly better than Constrained-INC. Also, RAxML is tied with NJMerge. Overall, NJMerge and Constrained-INC have very similar accuracy, and both

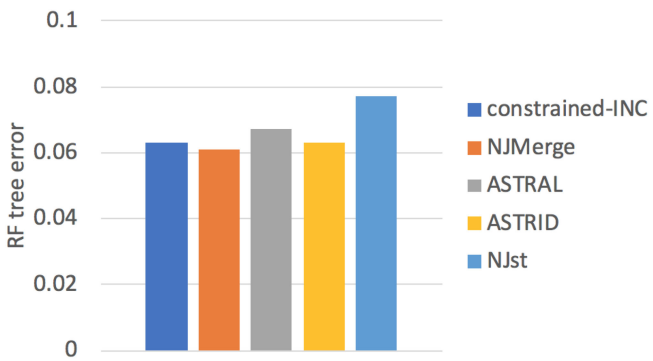

Fig. 4. Results for Experiment 4: Average species tree FN error rates on multi-locus species tree estimation on 1,000-species 1,000-gene datasets with very high ILS on the single replicate where ASTRAL completes within 48 hours. NJMerge and Constrained-INC are given the same set of ASTRAL constraint trees, and Constrained-INC is given the ASTRID distance matrix and induced quartet trees.

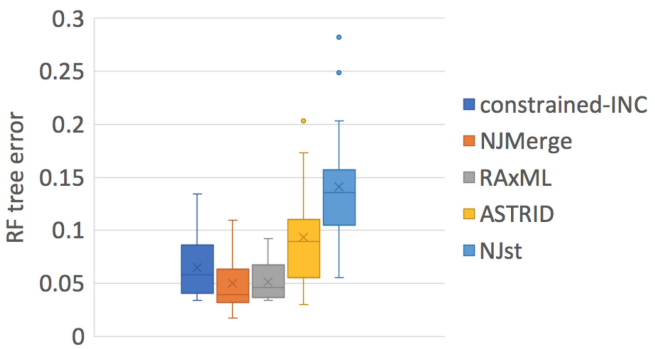

Fig. 6. Results for Experiment 4: Average FN error rates on 1,000-species 1,000-gene datasets with low/moderate ILS, restricted to those 17 replicates for which we are able to report a tree for RAxML (the best tree found in 48 hours). Constrained-INC and NJMerge are given the same set of RAxML constraint trees, and Constrained-INC was given the ASTRID distance matrix and the ASTRID guide tree to derive quartet trees.

TABLE 13
Results for Experiment 4: Average Quartet Tree Error Rate (max. 1.0) for Multi-Locus Species Tree Estimation on 1,000-Species 1,000-Gene Datasets With Low/Moderate ILS (over 20 replicate datasets)

Species Tree Method	RAxML complete	Remaining replicates
Constrained-INC	0.077	0.043
NJMerge	0.072	0.024
RAxML	0.023	n/a
ASTRID	0.075	0.028
NJst	0.083	0.035

Results for the 17 replicates that RAxML gave an output tree are shown separately from the remaining 3 replicates.

come close to the accuracy of RAxML for these low/moderate ILS conditions. A comparison of these methods with respect to quartet tree error rates shows very similar trends, but RAxML has lower error rates than the other methods (Table 13). This is an interesting difference, but the quartet error rates are low for all methods, so that the differences may not be meaningful.

3.5.3 Running Time

We now compare the running times for Constrained-INC and NJMerge, given a set of constraint trees. NJMerge and Constrained-INC both need to compute the ASTRID distance matrix, but Constrained-INC also needs to compute the ASTRID tree on the full dataset (as it uses this to compute quartet trees); we therefore report the time to compute the ASTRID tree in the time reported for Constrained-INC, but show this part separately in Table 14. We also do not report the time to compute the constraint trees, as this is common to both methods (and a standard part of most phylogenomic pipelines).

Table 14 shows these times on the datasets with 1,000 species and 1,000 genes, under both ILS levels. Note that running times do not change with the ILS level, and that for both conditions Constrained-INC uses less than 10 percent of the time used by NJMerge. Furthermore, the vast majority of the time used for Constrained-INC is calculating the ASTRID tree; the merger step itself is a very small part of its total time. Thus, Constrained-INC is much faster than NJMerge.

3.5.4 Summary of Experiment 4

In general, we find that Constrained-INC and NJMerge have similar accuracy for species tree estimation (with a slight advantage to NJMerge), but that Constrained-INC is faster. Under low/moderate ILS conditions (where we used constraint trees computed using RAxML), both NJMerge and Constrained-INC were similar in accuracy to RAxML (with Constrained-INC slightly less accurate), but both were much faster than RAxML. Under high ILS conditions, ASTRAL and the two DTM methods (Constrained-INC and NJMerge), both using ASTRAL for constraint trees, had nearly identical accuracy, but ASTRAL was much slower than NJMerge and Constrained-INC. Also, ASTRAL and RAxML failed to complete within the allowed time on many replicate datasets, but using these methods within the divide-and-conquer pipelines allowed them to complete on all replicates.

TABLE 14
Results for Experiment 4: Average Runtime (in seconds) $\pm$ Standard Deviation for Constrained-INC and NJMerge on 1,000 Species and 1,000 Genes

Datasets	ASTRID guide + Constrained-INC	NJMerge
Moderate ILS	$(17.7 \pm 0.1) + (1.0 \pm 0.05)$	1939 ± 66
Very high ILS	$(17.7 \pm 0.1) + (1.0 \pm 0.05)$	1950 ± 283

Both methods are given the same set of constraint trees (computed by ASTRAL). The time shown for Constrained-INC is restricted to the merging step, and does not include the time to compute the ASTRID tree (nor the time to compute the distance matrix). Note that calculating the ASTRID tree takes the vast majority (95 percent) of the total merging time.

3.6 Theoretical Results

We examine statistical consistency for pipelines using Constrained-INC under the GTR and MSC+GTR models.

Theorem 1. *Let Φ be a model of evolution. Suppose that (T, Θ) is a model tree under Φ (so that T denotes the tree topology and Θ denotes the numeric parameters) and let S denote the set of species for the leaves of T. Suppose that $S = S_1 \cup S_2 \ldots \cup S_k$ is a partition of S into pairwise disjoint sets. Let M_Q the method used to construct quartet trees and M_C the method used to construct the constraint trees on S_i, $i = 1, 2, \ldots, k$. If M_Q and M_C are statistically consistent for unrooted tree topology estimation under Φ, then the pipeline obtained by any decomposition of S into disjoint sets, followed by using M_Q, M_C and Constrained-INC, is a statistically consistent method for estimating the unrooted version of T.*

Proof. Under the assumption that M_Q and M_C are statistically consistent under Φ, then as the amount of data increases (e.g., we assume the number of sites increases if Φ is the GTR model and that the number of sites and number of genes both increase if Φ is the MSC+GTR model), with probability converging to 1, the output from the pipeline will include a set of unrooted constraint trees and quartet trees that are correct (i.e., identical to the model tree topology). Now consider any ordering of the taxa (including the one defined by Constrained-INC, based on its computed distance matrix). Inductively, it is easy to see that each taxon is added into a growing tree in exactly the correct location, when the quartet trees are correct. Hence, given such an input, Constrained-INC will combine the constraint trees into the true tree T, and so the pipeline is statistically consistent under Φ. □

The proof given here is nearly identical to Theorem 3 from [20], but did not use any properties about the taxon addition ordering; in fact, as we show here, any such pipeline is statistically consistent for any taxon addition ordering. However, the choice of taxon addition ordering does have an impact on the sample complexity, and this is why the algorithms described in [10] carefully constrain the taxon addition ordering.

Note that Theorem 1 immediately implies that many pipelines we have proposed using Constrained-INC to combine constraint trees are statistically consistent estimates of gene trees under the GTR model; for example, using NJ or FastME on logdet distance matrices to compute constraint trees, and many methods (e.g., the FPM method or NJ on

logdet distances, or maximum likelihood or Bayesian estimation under GTR) to compute quartet trees would be statistically consistent under the GTR model.

Corollary 1. *Consider a pipeline for species tree estimation from multi-locus datasets that has the following steps: it (i) computes gene trees using a method that is statistically consistent under GTR, (ii) computes the average internode distance matrix based on the gene trees, (iii) uses quartet trees induced by the ASTRID tree (computed on average internode distance matrix), (iv) computes constraint trees using any method that is statistically consistent under the MSC+GTR model, and then (v) combines the constraint trees using Constrained-INC in conjunction with some taxon addition ordering (e.g., computed using the internode distance matrix). Then the pipeline is statistically consistent for estimating the unrooted species tree topology under the MSC+GTR model.*

Proof. We will show that (a) the method for computing quartet trees is statistically consistent under the MSC+GTR model, and (b) and the method for computing unrooted constraint trees is statistically consistent under the MSC +GTR model. Hence, by Theorem 1, the pipeline will be statistically consistent for estimating unrooted species trees under the MSC+GTR model. Since the gene trees are computed using a method that is statistically consistent under the GTR model, then with probability converging to 1, as the number of sites per gene increases each gene tree will be correct. When the gene trees are correct, then as the number of genes increases, the average internode distance matrix will converge to an additive matrix for the true species tree [47]. The quartet trees are estimated using ASTRID on the average internode distance matrix, which (because the internode distance matrix converges to an additive matrix for the true species tree) is established to be statistically consistent under the MSC+GTR model [34]; hence, (a) is true. By assumption, the constraint trees are computed using methods that are statistically consistent under the MSC+GTR model; hence (b) is true. Hence (a) and (b) are established, and the result follows. □

Note that this corollary implies that using the average internode distance matrix, computing gene trees using maximum likelihood, using ASTRID (on the internode distance matrix) to compute quartet trees, and using ASTRAL to compute constraint trees, is statistically consistent under the MSC+GTR model. Note also that we did not rely on any property about the taxon addition ordering to establish statistical consistency.

4 Discussion

In our study, the divide-and-conquer pipeline using Constrained-INC was highly beneficial for computational and scalability issues, but had mixed impact on topological accuracy. Specifically, we saw that Constrained-INC used with even the best performing base methods (e.g., RAxML) never matched the accuracy of RAxML for gene tree estimation, and in fact generally was less accurate. On the other hand, the impact of using Constrained-INC in species tree estimation was more favorable, in that it came close to the accuracy of its base method that is used to compute constraint trees on subsets. Thus, for low ILS conditions, the best base method was RAxML, and for high ILS conditions the best base method was ASTRAL, and using Constrained-INC with the appropriate base method came close to the accuracy of that base method in our study.

To understand the difference between gene tree estimation and species tree estimation, we begin with a review of the prior studies that examined other DTM methods for species tree estimation [12], [20]. For example, Figure 10 in [12] showed that divide-and-conquer using RAxML for constraint trees followed by NJMerge matched or improved on the accuracy of RAxML on the full dataset when ILS was high, and matched its accuracy for low/moderate ILS conditions. Furthermore, these studies showed that using NJMerge and GTM with ASTRAL came close to the accuracy of ASTRAL trees on the full dataset, a trend that is also observed in this study for Constrained-INC.

Hence, divide-and-conquer has the *potential* to improve accuracy in some cases, but not all. Here we conjecture that *improvement is possible when the constraint trees are computed using methods that are not optimized for the specific estimation problem, but otherwise is likely to lead to a reduction in accuracy (or at best maintain accuracy)*.

We begin by considering gene tree estimation, where sequences evolve under the GTR model. For this condition, neither NJMerge nor Constrained-INC was able to match the accuracy of RAxML. The most likely explanation for this trend is that RAxML (which is regarded as possibly the most accurate method for gene tree estimation, and a leading maximum likelihood software) has superb accuracy that is hard to improve on. In particular, although maximum likelihood is NP-hard to solve, it may be that RAxML is able to find very good local optima (or perhaps even global optima) on these simulated datasets (which are all simulated under the same model that RAxML uses for its estimation model). Furthermore, maximum likelihood has been proven to have optimal sample complexity if solved exactly [9]. Together, these points suggest that RAxML may just be hard to improve on, in terms of accuracy. Interestingly, although FastTree2 is not generally considered as reliable as RAxML, it has also been shown to produce very accurate tree topologies under some conditions [48], and this may also explain why using divide-and-conquer with FastTree2 sometimes resulted in worse trees.

Now consider species tree estimation under MSC+GTR, so that genes evolve down a species tree under the MSC model and sequences evolve down each gene under the GTR model. Maximum likelihood under GTR, applied to a concatenation of the alignments for each gene, is not statistically consistent under this model (as noted earlier), and so RAxML is not optimized for this estimation problem. ASTRAL has the advantage over RAxML of being statistically consistent, but we argue it is also not optimized for the MSC+GTR tree estimation problem: its statistical consistency and excellent sample complexity [49] under the MSC +GTR model depends on having true gene trees, which is not generally achievable given finite length sequences per gene [23]. However, its empirical performance depends on other factors, such as how the set of allowed bipartitions is computed from the input, which could be impacted (favorably) through divide-and-conquer. Hence, neither of these

methods is optimized for accuracy under the MSC+GTR model, and there is a potential for divide-and-conquer to provide some benefit–even with respect to accuracy.

We now address the basic question of whether there are inherent limitations in divide-and-conquer strategies. It is well known that dense taxonomic sampling improves accuracy when phylogenies are estimated using good methods, such as maximum likelihood [50], so that the subset trees computed within any divide-and-conquer pipeline are unlikely to attain the same accuracy as a good ML heuristic applied to the entire dataset, and then restricted to the specified subset. Thus, although divide-and-conquer approaches (such as the one we explore here) can reduce running time, they can also reduce accuracy when the base method (used to construct the constraint trees as well as the entire tree on the full dataset) is highly accurate for the statistical inference problem. In such a case, there is a definite potential for divide-and-conquer to result in reduced accuracy.

That said, there is a very clear benefit to using divide-and-conquer strategies: they enable highly accurate methods that are computationally intensive to run on larger datasets with limited resources. Our study showed this clearly. Furthermore, phylogeny estimation is one of the most computationally intensive problems commonly attempted, with many phylogenomic datasets taking tends to hundreds of CPU years (e.g., the Avian phylogenomics project, which analyzed 48 bird genomes [42]). In addition, some of the most interesting methods for phylogeny estimation are too computationally intensive to use on even moderate sized datasets; the Bayesian method, StarBEAST2, is an example of such a method. Thus, divide-and-conquer approaches such as these may also be useful for improving scalability of powerful statistical approaches, and future research should investigate this.

5 Conclusion

Phylogeny estimation is a computationally challenging problem, with many moderate sized datasets taking years of CPU time. Prior divide-and-conquer strategies for phylogeny estimation have followed a natural design, where a dataset of species is divided into overlapping subsets, trees are computed on the subsets, and then merged together into a tree on the full dataset. DACTAL [51] is one such example, and has been shown to be useful for both gene tree estimation from unaligned sequences [51] and for multi-locus species tree estimation [52]. However, the final step requires methods for supertree construction, and all the best current methods (e.g., FastRFS [53]) are based on heuristics for NP-hard optimization problems and do not scale well to large datasets [54]. This was the motivation for the Disjoint Tree Merger approach, which combines a set of disjoint trees, and so can be used in divide-and-conquer pipelines without having to rely on supertree methods.

The method we explored, Constrained-INC, is one such DTM method, and was presented in [10] for gene tree estimation. In this study, we explored different ways of running Constrained-INC within an established divide-and-conquer pipeline, and evaluated the impact of these variants on both gene tree and species tree estimation. We also compared Constrained-INC to NJMerge, a prior method for combining constraint trees.

Constrained-INC has two advantages over NJMerge. The first is that Constrained-INC, unlike NJMerge, is guaranteed algorithmically to return a tree, but the same is not true for NJMerge. Instead, NJMerge uses a heuristic to decide whether to make a pair of leaves into siblings, and this heuristic can make mistakes (see discussion in [12] abut why NJMerge can fail, and see [12], [19], [20] for empirical evidence that it can fail). The second is running time: Constrained-INC is much faster than NJMerge; on the multi-locus datasets with 1,000 species and 1,000 genes, it used less than 10 percent of the time used by NJMerge, and on average finished in under a minute for each dataset.

One major benefit for using DTM methods within divide-and-conquer pipelines is that they can enable computationally intensive methods to complete on datasets given limited resources, whereas otherwise they might not be able to complete. We observed this in our study, where all analyses completed for Constrained-INC, while RAxML and ASTRAL were unable to complete on some inputs given limited computational resources. Hence, Constrained-INC improves scalability and speed for both ASTRAL and RAxML, while largely maintaining accuracy, and provides a running time advantage over NJMerge, the previous most accurate DTM.

We also established theoretical properties about the Constrained-INC method and its use within the divide-and-conquer pipeline. We proved that a divide-and-conquer pipeline using Constrained-INC with ASTRAL constraint trees and the ASTRID distance matrix and quartet trees is statistically consistent under the MSC+GTR model.

This study contributes to a growing body of research exploring divide-and-conquer strategies used with DTM methods in phylogenomic estimation. Our results show that Constrained-INC is similar in performance (i.e., impact on accuracy, running time, and scalability) to other DTM methods, which is noteworthy, since the different DTM methods each use very different specific techniques to merge sets of disjoint trees. Thus, divide-and-conquer using DTM is a promising general algorithmic strategy that should be explored further. At the same time, our study showed conditions where these divide-and-conquer strategies can reduce accuracy, and reveals challenges to (and possibly inherent limitations of) divide-and-conquer strategies for statistical phylogenetics. Future research is needed to explore these techniques under a wider range of model conditions, in order to better assess the value and potential of these approaches.

Acknowledgments

The authors thank the anonymous reviewers and also Sarah Christensen, Pranjal Vachaspati, and Vladimir Smirnov for valuable feedback that led to improvements in the paper. This work was supported in part by NSF grants CCF-1535977 (to TW) and 1535989 (to SR). EKM was supported by the NSF Graduate Research Fellowship (Award DGE-1144245) and the Ira and Debra Cohen Graduate Fellowship in Computer Science. TL was supported by a research assistantship funded in part by NSF CAREER award number 1553284 (to Stefanie Jegelka). Computational experiments were performed on Blue Waters. This research is part of the Blue Waters sustained-petascale computing project, which is supported by the NSF (Awards OCI-0725070 and

ACI-1238993) and the state of Illinois. Blue Waters is a joint effort of the University of Illinois at Urbana-Champaign and its National Center for Supercomputing Applications.

References

[1] W. Maddison, "Gene trees in species trees," *Systematic Biol.*, vol. 46, no. 3, pp. 523–536, 1997.

[2] N. Goldman, "Maximum likelihood inference of phylogenetic trees, with special reference to a Poisson process model of DNA substitution and to parsimony analyses," *Systematic Zool.*, vol. 39, pp. 345–361, 1990.

[3] Z. Yang, *Computational Molecular Evolution*. Oxford, UK: Oxford Univ. Press, 2006.

[4] T. H. Jukes and C. R. Cantor, "Evolution of protein molecules," in *Mammalian Protein Metabolism*, vol. 3, H. Munro, Ed. New York, NY, USA: Academic Press, 1969, pp. 21–132.

[5] S. Tavaré, "Some probabilistic and statistical problems in the analysis of DNA sequences," in *Lectures on Mathematics in the Life Sciences*, vol. 17. Providence, RI, USA: American Mathematical Society, 1986, pp. 57–86.

[6] P. Erdös, M. Steel, L. Székely, and T. Warnow, "A few logs suffice to build (almost) all trees (I)," *Random Structures Algorithms*, vol. 14, pp. 153–184, 1999.

[7] P. Erdös, M. Steel, L. Székely, and T. Warnow, "A few logs suffice to build (almost) all trees (II)," *Theor. Comput. Sci.*, vol. 221, pp. 77–118, 1999.

[8] T. Warnow, B. M. E. Moret, and K. St. John, "Absolute convergence: True trees from short sequences," in *Proc. ACM-SIAM Symp. Discrete Algorithms*, 2001, pp. 186–195.

[9] S. Roch and A. Sly, "Phase transition in the sample complexity of likelihood-based phylogeny inference," *Probability Theory Related Fields*, vol. 169, no. 1, pp. 3–62, 2017.

[10] Q. Zhang, S. Rao, and T. Warnow, "Constrained incremental tree building: New absolute fast converging phylogeny estimation methods with improved scalability and accuracy," *Algorithms Mol. Biol.*, vol. 14, no. 2, 2019, Art. no. 2. [Online]. Available: https://doi.org/10.1186/s13015–019-0136-9

[11] N. Saitou and M. Nei, "The neighbor-joining method: A new method for reconstructing phylogenetic trees," *Mol. Biol. Evol.*, vol. 4, pp. 406–425, 1987.

[12] E. K. Molloy and T. Warnow, "Statistically consistent divide-and-conquer pipelines for phylogeny estimation using NJMerge," *Algorithms Mol. Biol.*, vol. 14, no. 14, 2019, Art. no. 14, doi: 10.1186/s13015–019-0151-x.

[13] A. Stamatakis, "RAxML version 8: A tool for phylogenetic analysis and post-analysis of large phylogenies," *Bioinformatics*, vol. 30, no. 9, pp. 1312–1313, 2014.

[14] T. Le, "GitHub site for the inc and constrained-inc software," 2019. [Online]. Available: https://github.com/steven-le-thien/INC

[15] T. Le, A. Sy, E. Molloy, Q. Zhang, S. Rao, and T. Warnow, "Using INC within divide-and-conquer phylogeny estimation - datasets," 2019. [Online]. Available: https://databank.illinois.edu/datasets/IDB-8518809

[16] E. K. Molloy and T. Warnow, "Datasets for the paper 'NJMerge: A generic technique for scaling phylogeny estimation methods and its application to species trees'," 2018. [Online]. Available: https://doi.org/10.13012/B2IDB-1424746_V1

[17] P. Buneman, "A note on the metric properties of trees," *J. Combinatorial Theory*, vol. 17, pp. 48–50, 1974.

[18] P. Erdös, M. Steel, L. Székely, and T. Warnow, "Local quartet splits of a binary tree infer all quartet splits via one dyadic inference rule," *Comput. Artif. Intell.*, vol. 16, no. 2, pp. 217–227, 1997.

[19] E. K. Molloy and T. Warnow, "TreeMerge: A new method for improving the scalability of species tree estimation methods," *Bioinformatics*, vol. 35, no. 14, pp. i417–i426, Jul. 2019. [Online]. Available: https://doi.org/10.1093/bioinformatics/btz344

[20] V. Smirnov and T. Warnow, "Unblended disjoint tree merging using GTM improves species tree estimation," *BMC Genomics*, vol. 21, 2020, Art. no. 235.

[21] K. Liu *et al.*,"SATé-II: Very fast and accurate simultaneous estimation of multiple sequence alignments and phylogenetic trees," *Syst. Biol.*, vol. 61, no. 1, pp. 90–106, Jan. 2012.

[22] S. Mirarab, N. Nguyen, L.-S. Wang, S. Guo, J. Kim, and T. Warnow, "PASTA: Ultra-large multiple sequence alignment of nucleotide and amino acid sequences," *J. Comput. Biol.*, vol. 22, pp. 377–386, 2015.

[23] S. Roch, M. Nute, and T. Warnow, "Long-branch attraction in species tree estimation: Inconsistency of partitioned likelihood and topology-based summary methods," *Systematic Biol.*, vol. 68, no. 2, pp. 281–297, 2018.

[24] J. T. Chang, "Full reconstruction of Markov models on evolutionary trees: Identifiability and consistency," *Math. Biosciences*, vol. 137, no. 1, pp. 51–73, 1996.

[25] S. Roch, "A short proof that phylogenetic tree reconstruction by maximum likelihood is hard," *IEEE/ACM Trans. Comput. Biol. Bioinf.*, vol. 3, no. 1, pp. 92–94, Firstquarter 2006.

[26] L. Nguyen, H. Schmidt, A. von Haeseler, and B. Minh, "IQ-TREE: A fast and effective stochastic algorithm for estimating maximum likelihood phylogenies," *Mol. Biol. Evol.*, vol. 32, no. 1, pp. 268–274, 2015.

[27] S. Guindon and O. Gascuel, "A simple, fast, and accurate algorithm to estimate large phylogenies by maximum likelihood," *Systematic Biol.*, vol. 52, no. 5, pp. 696–704, 2003.

[28] M. N. Price, P. S. Dehal, and A. P. Arkin, "FastTree 2 - Approximately maximum-likelihood trees for large alignments," *PLoS One*, vol. 5, no. 3, pp. 1–10, 2010.

[29] M. A. Steel, "Recovering a tree from the leaf colourations it generates under a Markov model," *Appl. Math. Lett.*, vol. 7, no. 2, pp. 19–23, 1994.

[30] K. Atteson, "The performance of neighbor-joining methods of phylogenetic reconstruction," *Algorithmica*, vol. 25, no. 2/3, pp. 251–278, 1999.

[31] T. Warnow, *Computational Phylogenetics: An Introduction to Designing Methods for Phylogeny Estimation*. Cambridge, U.K.: Cambridge Univ. Press, 2018.

[32] S. Mirarab, R. Reaz, M. S. Bayzid, T. Zimmermann, M. S. Swenson, and T. Warnow, "ASTRAL: Genome-scale coalescent-based species tree estimation," *Bioinformatics*, vol. 30, no. 17, pp. i541–i548, 2014.

[33] S. Mirarab and T. Warnow, "ASTRAL-II: Coalescent-based species tree estimation with many hundreds of taxa and thousands of genes," *Bioinformatics*, vol. 31, pp. i44–52, 2015.

[34] P. Vachaspati and T. Warnow, "ASTRID: Accurate species TRees from internode distances," *BMC Genomics*, vol. 16, no. 10, 2015, Art. no. S3.

[35] J. Chifman and L. Kubatko, "Quartet inference from SNP data under the coalescent model," *Bioinformatics*, vol. 30, no. 23, pp. 3317–3324, 2014.

[36] P. Vachaspati and T. Warnow, "SVDquest: Improving SVDquartets species tree estimation using exact optimization within a constrained search space," *Mol. Phylogenetics Evol.*, vol. 124, pp. 122–136, 2018.

[37] D. Bryant, R. Bouckaert, J. Felsenstein, N. A. Rosenberg, and A. RoyChoudhury , "Inferring species trees directly from biallelic genetic markers: Bypassing gene trees in a full coalescent analysis," *Mol. Biol. Evol.*, vol. 29, no. 8, pp. 1917–1932, Mar. 2012. [Online]. Available: https://doi.org/10.1093/molbev/mss086

[38] J. Heled and A. J. Drummond, "Bayesian inference of species trees from multilocus data," *Mol. Biol. Evol.*, vol. 27, no. 3, pp. 570–580, Nov. 2009. [Online]. Available: https://doi.org/10.1093/molbev/msp274

[39] S. Roch and M. Steel, "Likelihood-based tree reconstruction on a concatenation of aligned sequence data sets can be statistically inconsistent," *Theor. Population Biol.*, vol. 100, pp. 56–62, 2015.

[40] A. Sand, M. K. Holt, J. Johansen, G. S. Brodal, T. Mailund, and C. N. S. Pedersen, "tqDist: A library for computing the quartet and triplet distances between binary or general trees," *Bioinformatics*, vol. 30, no. 14, pp. 2079–2080, Mar. 2014. [Online]. Available: https://doi.org/10.1093/bioinformatics/btu157

[41] W. Fletcher and Z. Yang, "INDELible: A flexible simulator of biological sequence evolution," *Mol. Biol. Evol.*, vol. 26, no. 8, pp. 1879–1888, 2009.

[42] E. D. Jarvis *et al.*, "Whole-genome analyses resolve early branches in the tree of life of modern birds," *Science*, vol. 346, no. 6215, pp. 1320–1331, 2014.

[43] E. K. Molloy and T. Warnow, "To include or not to include: The impact of gene filtering on species tree estimation methods," *Systematic Biol.*, vol. 67, no. 2, pp. 285–303, Sep. 2017. [Online]. Available: https://doi.org/10.1093/sysbio/syx077

[44] V. Lefort, R. Desper, and O. Gascuel, "FastME 2.0: A comprehensive, accurate, and fast distance-based phylogeny inference program," *Mol. Biol. Evol.*, vol. 32, no. 10, pp. 2798–2800, 2015.

[45] D. L. Swofford, "PAUP* (*Phylogenetic analysis using PAUP), Version 4a161," 2018. [Online]. Available: http://phylosolutions.com/paup-test/

[46] M. R. Lacey and J. T. Chang, "A signal-to-noise analysis of phylogeny estimation by neighbor-joining: Insufficiency of polynomial length sequences," *Math. Biosciences*, vol. 199, no. 2, pp. 188–215, 2006.
[47] E. S. Allman, J. H. Degnan, and J. A. Rhodes, "Species tree inference from gene splits by unrooted star methods," *IEEE/ACM Trans. Comput. Biol. Bioinf.*, vol. 15, no. 1, pp. 337–342, Jan./Feb. 2018.
[48] K. Liu, C. R. Linder, and T. Warnow, "RAxML and FastTree: Comparing two methods for large-scale maximum likelihood phylogeny estimation," *PloS One*, vol. 6, no. 11, 2011, Art. no. e27731.
[49] S. Shekhar, S. Roch, and S. Mirarab, "Species tree estimation using ASTRAL: How many genes are enough?" *IEEE/ACM Trans. Comput. Biol. Bioinf.*, vol. 15, no. 5, pp. 1738–1747, Sep. 2018. [Online]. Available: https://doi.org/10.1109/TCBB.2017.2757930
[50] T. A. Heath, S. M. Hedtke, and D. M. Hillis, "Taxon sampling and the accuracy of phylogenetic analyses," *J. Systematics Evol.*, vol. 46, no. 3, pp. 239–257, 2008.
[51] S. Nelesen, K. Liu, L.-S. Wang, C. R. Linder, and T. Warnow, "DACTAL: Divide-and-conquer trees (almost) without alignments," *Bioinformatics*, vol. 28, no. 12, pp. i274–i282, 2012.
[52] M. S. Bayzid, T. Hunt, and T. Warnow, "Disk covering methods improve phylogenomic analyses," *BMC Genomics*, vol. 15 (Suppl 6), 2014, Art. no. S7.
[53] P. Vachaspati and T. Warnow, "FastRFS: Fast and accurate Robinson-Foulds supertrees using constrained exact optimization," *Bioinformatics*, vol. 33, no. 5, pp. 631–639, 2016.
[54] T. Warnow, "Divide-and-conquer tree estimation: Opportunities and challenges," in *Bioinformatics and Phylogenetics*, T. Warnow, Ed. Berlin, Germany: Springer, 2019.

Thien Le received the undergraduate degree from the University of Illinois at Urbana-Champaign (UIUC), Champaign, Illinois, in May 2019, where he worked with Tandy Warnow on phylogeny research. He is currently working toward the EECS PhD degree at the Massachusetts Institute of Technology, Cambridge, Massachusetts. He received the Most Outstanding Major Award in mathematics and computer science from the UIUC Department of Mathematics, in April 2019.

Aaron Sy received the undergraduate degree in EECS from the University of California, Berkeley, Berkeley, California, in 2018, and the MS degree in EECS from the University of California, Berkeley, Berkeley, California, in 2019, advised by Satish Rao. Currently he works as an engineer at YouTube.

Erin K. Molloy received the BS degree in physics from the University of Chicago, Chicago, Illinois. She is working toward the CS PhD degree at the University of Illinois at Urbana-Champaign, Champaign, Illinois, where she is co-advised by Tandy Warnow and Bill Gropp. Before joining Illinois, she was a neuroimaging researcher with the University of Wisconsin-Madison, Madison, Wisconsin.

Qiuyi Zhang received the PhD degree in applied mathematics and computer science under professor Satish Rao and professor Nikhil Srivastava, in 2019. He is a member of Google Brain, working on hyperparameter optimization on the Vizier team, and his interests are in the intersection of optimization, theoretical computer science, and machine learning.

Satish Rao received the PhD degree from the Massachusetts Institute of Technology, Cambridge, Massachusetts, in 1989, then held a scientist position at NEC Laboratories until 1999. He then joined the faculty at the University of California, Berkeley, Berkeley, California, where he is now a professor of computer science. He works in the areas of combinatorial optimization and approximation algorithms. He received the Fulkerson Prize (with Sanjeev Arora and Umesh Vazirani), in 2012 for his work on approximation algorithms for sparsest cut, and was elected a fellow of the ACM, in 2013.

Tandy Warnow received the PhD degree in mathematics from the University of California, Berkeley, Berkeley, California under the direction of Gene Lawler, and did postdoctoral training with Simon Tavaré and Michael Waterman at the University of Southern California, Los Angeles, California. She is the founder professor of computer science at the University of Illinois at Urbana-Champaign, Champaign, Illinois. Her awards include the David and Lucile Packard foundation Award (1996), a Radcliffe Institute fellowship (2006), and the John Simon Guggenheim Foundation fellowship (2011). She was elected a fellow of ACM, in 2015 and of the ISCB, in 2017.