Disk-Covering, a Fast-Converging Method for Phylogenetic Tree Reconstruction

DANIEL H. HUSON,[1] SCOTT M. NETTLES,[2] and TANDY J. WARNOW[3]

ABSTRACT

The evolutionary history of a set of species is represented by a *phylogenetic tree*, which is a rooted, leaf-labeled tree, where internal nodes represent ancestral species and the leaves represent modern day species. Accurate (or even boundedly inaccurate) topology reconstructions of large and divergent trees from realistic length sequences have long been considered one of the major challenges in systematic biology. In this paper, we present a simple method, the *Disk-Covering Method* (DCM), which *boosts* the performance of base phylogenetic methods under various Markov models of evolution. We analyze the performance of DCM-boosted distance methods under the Jukes–Cantor Markov model of biomolecular sequence evolution, and prove that for almost all trees, *polylogarithmic* length sequences suffice for complete accuracy with high probability, while polynomial length sequences always suffice. We also provide an experimental study based upon simulating sequence evolution on model trees. This study confirms substantial reductions in error rates at realistic sequence lengths.

Key words: algorithms, evolution, phylogenetic trees, clustering, Jukes Cantor, biomolecular data, chordal graphs.

1. INTRODUCTION

THE EVOLUTION OF BIOMOLECULAR SEQUENCES is usually modeled as a Markov process operating on a rooted binary tree. A biomolecular sequence at the root of the tree "evolves down" the tree, each edge of the tree introducing point mutations, thereby generating sequences at the leaves of the tree. The *phylogenetic tree reconstruction problem* is to take the sequences that occur at the leaves of the tree, and infer, as accurately as possible, the tree that generated the sequences. Under most Markov models of sequence evolution, locating the root is impossible. Consequently, the primary objective in phylogenetic analysis is to recover the branching process, as represented by the unrooted leaf-labeled topology of the evolutionary tree, and the secondary objective is to estimate the parameters of the evolutionary process. While inferring the parameters of the evolutionary process is of interest to both biologists and statisticians, this is relatively easy once an accurate estimate of the topology of the true tree is obtained; consequently, biologists have mostly focused upon the "simpler" question of inferring the tree topology. Methods for inferring or estimating the evolutionary history of biomolecular sequences are evaluated according to the accuracy of this topology

---

[1]Celera Genomics, Rockville, Maryland.

[2]Department of Electrical and Computer Engineering, University of Texas, Austin, Texas.

[3]Department of Computer Science, University of Texas, Austin, Texas.

estimation. Indeed, this is one of the most important problems for computational biology as a whole, because of the centrality of evolutionary studies to biology (Dobzhansky, 1993). [Evolutionary trees are often the basis of multiple sequence alignment algorithms (Gusfield, 1991; Gusfield and Wang, 1996; Hein, 1989), protein structure prediction routines (Rost and Sander, 1993), and other problems in biology.]

Experimentally investigating the performance of phylogenetic methods by simulating sequence evolution on different model trees in order to determine how the sequence length affects the accuracy of the topology prediction is central to systematic biology studies (see, for example, Hillis, 1996; Hillis *et al.*, 1994; Huelsenbeck, 1995; Huelsenbeck and Hillis, 1993; Kuhner and Felsenstein, 1994; Saitou and Imanishi, 1989; Schöniger and von Haeseler, 1995; Sourdis and Nei, 1996; Strimmer and von Haeseler, 1996). The focus on sequence length in these studies is primarily *practical*: biomolecular sequences are not particularly long (those used for phylogenetic tree reconstruction purposes are typically bounded by 1000 nucleotides, often by much smaller numbers), and sequence lengths of 5000 nucleotides are generally considered to be *unusually long* (Puruis and Quicke, 1997). Furthermore, experimental and anecdotal evidence suggests that methods are very sensitive to sequence lengths, and that different methods can have great differences in their accuracy on realistic sequence lengths.

In an earlier paper (Huson *et al.*, 1998b), we showed that a critical factor affecting the accuracy of different methods is the maximum evolutionary distance in the tree, which we call the *divergence* of the tree. Theoretical studies (Berry and Gascuel, 1997; Erd´os *et al.*, 1997a,b, 1999) have bounded the convergence rates of various polynomial time distance methods [Neighbor-Joining (Saitou and Nei, 1987), a simple and very popular clustering technique used by systematic biologists, and more sophisticated methods developed by the theoretical computer science community (Agarwala *et al.*, 1996; Ambainis *et al.*, 1997; Farach *et al.*, 1995), and shown that these bounds grow exponentially in the divergence. This exponential dependency upon the divergence translates to superpolynomial convergence rates for almost all trees (Erdős *et al.*, 1997b), suggesting that accuracy *may* be poor if these methods are used to infer tree topologies under conditions of high divergence. However, these are upper bounds, and may be pessimistic.

Our paper has several contributions. First, we introduce the concept of *fast convergence* under a model of evolution, such as the Jukes–Cantor model. By this we mean that for all model trees, a method will with high probability recover the tree from sequences that grow only polynomially in the number of leaves, once we bound the mutation probabilities on the edges of the tree. We present a general technique for designing methods that are fast converging, and we present a new fast-converging method. This particular method is actually a *boosted* version of a classical method, called the Buneman Method; thus, we also introduce a general technique for boosting the performance of phylogenetic methods. This is the *Disk-Covering Method* (DCM). In its exact form DCM is not polynomial time, but has provable performance guarantees when used with certain base methods. For example, we can prove that under the Jukes–Cantor Markov model (Jukes and Cantor, 1969), the DCM–Buneman method is *fast converging*. In its heuristic form, DCM is polynomial time, and our experimental studies indicate substantial reductions in error rates for DCM-boosted polynomial time methods.

The rest of this paper is organized as follows. In Section 2 we present definitions and some basic results about convergence rates of standard methods. We also present a general technique for establishing fast convergence. In Section 3, we describe the fast-converging method, DCM-Buneman, and provide the proof of fast convergence. In Section 4, we describe DCM more generally, and describe how it can be used with other distance-based methods. In Section 5, we present the results of a preliminary experimental study. Finally, in Section 6, we present open problems and discuss related research.

2. BASICS

2.1. *Markov models of evolution*

Most Markov models of evolution assume that the sites in the sequence evolve identically and independently (i.i.d.), and for the most part, the results that can be proven about any Markov model of i.i.d. site evolution can be proven about any other such model. Because of this, we will describe our results in terms of the Jukes–Cantor model.

T. Jukes and C. Cantor (1969) introduced a very simple model of DNA sequence evolution. [Please see the excellent review articles by Felsenstein (1988) and Swofford *et al.* (1996) for more information about statistical aspects of phylogenetic inference.]

Definition 1.   *Let T be a fixed rooted tree with leaves labeled* $1 \cdots n$. *The* **Jukes–Cantor** *Markov model of evolution makes the following assumptions:*

1. *The sites (positions within the sequence) evolve identically and independently (i.i.d.) down the tree from the root. The probability of an event on edge e is independent of the events outside the subtree below e.*
2. *The possible states for each site are A, C, T, G (denoting the four possible nucleotides), and for each site, the state at the root is drawn from a distribution (typically uniform).*
3. *For each edge $e = (u, v) \in E(T)$, with u the parent of v, if the state of a site is different at u than at v, then the probability that v has any particular state of the three remaining states is equal.*
4. *Associated to each edge e in the tree T is a Poisson random variable X_e for the number of mutations of a randomly selected site on that edge. We let λ_e denote the expectation of X_e.*

We will sometimes refer to the pair (T, λ) as a Jukes–Cantor *tree.*

We are interested in estimating the topology of the Jukes–Cantor tree, and will study the accuracy of different methods for this problem. Since the number of changes on each edge influences the sequence length that is needed to estimate the tree, we will be interested in the performance of different phylogenetic estimators when $f = \min_e \lambda_e$ and $g = \max_e \lambda_e$ are fixed, but arbitrary.

2.2. Topological accuracy

While complete accuracy is the objective, partial accuracy is the rule, especially when trees are large and/or highly divergent (that is, contain evolutionarily distant pairs of taxa). For this reason, quantification of degrees of accuracy have been proposed. The most favored such quantification is as follows.

Definition 2.   *Let T be the (unrooted) true tree, and let T' be the (unrooted) inferred tree, both leaf-labeled by a set S of taxa. Let e be an edge in T, and let π_e denote the bipartition induced on S by the removal of e from T. Let $C(T) = \{\pi_e : e \in E(T)\}$ and let $C(T')$ be equivalently defined. Then $T = T'$ if and only if $C(T) = C(T')$. Any bipartition $\pi \in C(T) - C(T')$ is called a* **false negative** *(FN) and any bipartition in $C(T') - C(T)$ is said to be a* **false positive** *(FP). Thus, a false negative is an edge in the true tree that is missing from the inferred tree, while a false positive is an edge in the inferred tree that is not in the true tree. We will say that an edge $e \in E(T)$ is* **recovered** *in T' (or by the method M which produces T') if the bipartition in $C(T)$ associated to e appears in $C(T')$. The* **FN rate** *is the ratio of FN and the number of edges in T, and similarly the FP rate is the ratio of FP and the number of edges in T'.*

Usually the model (true) tree is binary, and hence has $n - 3$ edges, where n is the number of leaves. Also, many distance methods, such as Neighbor-Joining (Saitou and Nei, 1987) and the method of Agarwala *et al.* (1996) (which obtains a 3-approximation to the L_∞-nearest tree problem), always produce binary trees. However, the Buneman Tree method (Buneman, 1971) often produces unresolved (i.e., not binary) trees.

2.3. Performance criteria

The assumption of Markov models of evolution allows the performance of phylogenetic tree reconstruction methods to be assessed, either through analytical means or by experiments. There are several criteria that have been considered to be of fundamental importance by statisticians working in evolutionary tree reconstruction (Felsenstein, 1988):

- *statistical consistency* (with respect to a given model of evolution), which is said to be given, if the probability of recovering the leaf-labeled tree converges to 1 as the sequence length increases;
- *convergence rate* (with respect to a given model of evolution), which is the rate at which the probability (of recovering the leaf-labeled tree) goes to 1, as the sequence length increases; and
- *accuracy* (with respect to a given model of evolution), the expected number of topological errors at a given sequence length.

The reference to the model of evolution is important, as some methods are statistically consistent under some models but not under others.

Under the Jukes–Cantor model, many distance methods (and all the methods we consider in this paper) are statistically consistent. However, if the assumption of *i.i.d.* site evolution (or that the sites evolve under different

rates but from a known distribution) is relaxed, then positive results become less likely. For example, under more general models of evolution, even maximum likelihood (under the correct model!) can be statistically inconsistent (Steel *et al.*, 1994). The theorems we will develop in this paper are stated in terms of Jukes–Cantor site evolution, but apply more generally to other models for which statistical consistency of distance-based methods has been established.

2.4. Distance-based reconstruction

We begin with some definitions.

Definition 3. *A matrix D is called* **additive** *(or said to be a* tree-metric*) if there exists a tree T with positive edge weighting w such that $D_{ij} = d_{ij}^T := \sum_{e \in P_{ij}} w(e)$, where P_{ij} is the path in T between leaves i and j.*

Additive matrices have the nice property that they correspond uniquely to positively edge-weighted trees (Buneman, 1971), and that given D, the tree T and the weighting w can be recovered in polynomial time (Waterman *et al.*, 1977).

Let $E(T)$ denote the set of edges of T. We represent the evolutionary process by a set $\{X_e : e \in E(T)\}$ of Poisson processes, where X_e is the Poisson random variable for the number of mutations of a random site on the edge e. Let $X_{ij} = \sum_{e \in P_{ij}} X_e$. Then X_{ij} is a Poisson random variable. Let $\lambda_{ij} = \mathbb{E}[X_{ij}]$. Then, λ is an additive matrix. We will call λ_{ij} the *expected evolutionary distance*, or *true distance* between i and j. Clearly, $\lambda_{ij} = \sum_{e \in P_{ij}} \lambda_e$, where $\lambda_e = \mathbb{E}[X_e]$. Because the matrix λ is additive, given λ the tree T can be constructed in polynomial time using a number of different distance-based methods. Furthermore, a *distance correction transformation* exists for the Jukes–Cantor Markov model of site evolution, so that arbitrarily good approximations to the matrix λ can be obtained if the sequence length is unboundedly large (Felsenstein, 1988; Warnow, 1996).

Definition 4. *The* **Jukes–Cantor distance correction** *is given as follows:*

$$d_{ij} := -\frac{3}{4} \log \left(1 - \frac{4}{3} h_{ij} \right)$$

where h_{ij} is the normalized Hamming distance, or $H(i, j)/k$, where $H(i, j)$ is the Hamming distance between sequences i and j, and k is the sequence length.

It is not hard to show that for all Jukes–Cantor trees (T, λ) and for all i, j, as $k \to \infty$, that $d_{ij} \to \lambda_{ij}$. Consequently, the following two-step process is a statistically consistent *distance-based* approach to Jukes–Cantor tree reconstruction:

First, compute the Jukes–Cantor distance matrix d, *then*, map d (using some distance method M such as Neighbor-Joining) to a (nearby) additive matrix $M(d) = D$.

Since accuracy in phylogenetic reconstruction is based upon comparisons between unrooted tree topologies, we will say that the method is accurate on input d if D and λ define the same unrooted leaf-labeled tree, even if they assign different weights to the edges of the tree.

2.5. Some distance methods

In this paper we demonstrate the value of DCM boosting on three different distance-based methods, all of which are statistically consistent for the Jukes–Cantor model when used with properly corrected distances. We now describe these three methods.

2.5.1. Buneman. The Buneman Tree method was originally suggested by Peter Buneman (1971), and polynomial time algorithms for this method were obtained in Bandelt and Dress (1992) and Berry and Gascuel (1997). This method takes as input a dissimilarity matrix d, and computes a tree as follows. First, the topology on every quartet of taxa is inferred using the Four-Point method, as follows:

The **Four-Point Method** (FPM) computes trees on four-leaf subsets only. Given the 4×4 dissimilarity matrix on i, j, k, l, the topology $ij \mid kl$ is returned (meaning i, j are separated from k, l by an edge) if $d_{ij} + d_{kl} < min\{d_{ik} + d_{jl}, d_{il} + d_{jk}\}$. If the minimum of the three pairwise sums is not unique, then the FPM returns the star tree (that is, the tree with one interior node, and all leaves adjacent to that interior node).

Given a set Q of trees, one on each quartet of leaves, the Buneman Tree is defined to be the *maximally resolved* tree satisfying the following condition:

- for all quartets i, j, k, l if T restricted to i, j, k, l induces a binary tree (instead of a star), then the tree in Q on i, j, k, l is the same binary tree.

Such a tree always exists, since the star tree satisfies this constraint. What is nice is that the maximally resolved tree with this property is unique, as the following lemma shows.

Lemma 1. *Let d be an input dissimilarity matrix, and let Q be the set of four-leaf trees defined by the FPM, and let T be the Buneman Tree defined by d. Then C(T) is the set of bipartitions (A, B) defined by*

- *For all $\{a, a'\} \subseteq A$ and $\{b, b'\} \subseteq B$, the tree $a, a' \mid b, b' \in Q$.*

2.5.2. The Agarwala et al. Method. The L_∞-nearest tree problem is to find, for input dissimilarity matrix d, an additive matrix D minimizing $L_\infty(d, D)$. This is an NP-hard optimization problem (Farach *et al.*, 1995), but can be 3-approximated in polynomial time. Thus, a 3-approximation algorithm for the L_∞-nearest tree problem takes as input a dissimilarity matrix d, and returns an additive metric D' such that $L_\infty(d, D') \leq 3L_\infty(d, D)$ for all additive metrics D. There are several 3-approximation algorithms for this problem, of which the one by Agarwala *et al.* (1996) is the first (to our knowledge) with a provable performance guarantee.

2.5.3. Neighbor-Joining. This is a polynomial time method very much favored in the systematic biology community for use with large data sets. See Saitou and Nei (1987) for the first paper on this method and Atteson (1997) for a proof of statistical consistency and a bound on its convergence rate.

2.6. Convergence rates of some distance methods

We begin with the only established upper bounds on the convergence rates of these three methods. This analysis relates the convergence rate to the "divergence" of a tree, which we now define.

Definition 5. *Let T be an arbitrary Jukes–Cantor tree and let λ_e be the expected number of mutations of a random site on an edge e. Then $\lambda_{ij} = \sum_{e \in P_{ij}} \lambda_e$, where P_{ij} is the path in the tree T between leaves i and j. We define the* **divergence** *of the tree T to be $\lambda_{\max} = \max_{ij}\{\lambda_{ij}\}$.*

Note that $\lambda_{\max}$ is unbounded even when n (the number of leaves) is bounded, due to changes that are not observed.

Theorem 1. *Let T be a Jukes–Cantor tree with n leaves, Let λ_e be the expected number of changes of a random site on edge e, and let $f = \min_e \lambda_e$. Let d_{ij} be an estimation of λ_{ij}, the expected number of changes of a random site on the path between i and j. Let $\varepsilon = L_\infty(d, \lambda)$.*

1. *The Neighbor-Joining method is guaranteed to be accurate if $\varepsilon < f/2$.*
2. *The Buneman Tree method is guaranteed to be accurate if $\varepsilon < f/2$. Furthermore, every edge $e \in E(T)$ such that $\lambda_e > 2\varepsilon$ is recovered by the Buneman Tree Method.*
3. *Any 3-approximation algorithm for the L_∞-nearest tree problem is guaranteed to be accurate if $\varepsilon < f/8$. Furthermore, if $\lambda_e > 8\varepsilon$, then the edge e is recovered by the algorithm.*

For each method above, there is a constant C that depends upon f and δ such that if the sequence length k exceeds

$$C \log n e^{O(\lambda_{\max})}$$

then with probability at least $1 - \delta$, the method is accurate given sequences of length k generated on T.

Proof. (1) was proven in Atteson (1997), and (2) and (3) were proven in Erdős *et al.* (1999) and Huson *et al.* (1998b). The bound on the convergence rate is based upon the sufficient condition for accuracy, established in items (1)–(3) above, and is given in Erdős *et al.* (1999). ∎

Here is an intuition about why this result should be true. Recall that $\lambda_{ij} = \mathbb{E}[X_{ij}]$. As X_{ij} is a Poisson random variable, its expectation is the same as its variance. Consequently, $\lambda_{ij} = \mathrm{Var}[X_{ij}]$, and when the variance is high, errors in estimating λ_{ij} are also high.

These results prove statistical consistency, and also provide *upper bounds* on the sequence length that suffices for accuracy with high probability for these three methods. They also provide bounds on the false-negative rate for the Buneman Tree Method, and for any 3-approximation algorithm for the L_∞-nearest tree. The term $\lambda_{\max}$ is the important term in this upper bound, and it is bounded by $g \cdot diam(T)$, where $g = \max_e \lambda_e$, and where $diam(T)$ denotes the length of the longest path (measured in terms of the number of edges) in the tree T. The diameter of a tree on n leaves can be as much as $n - 1$, or as small as $O(\log n)$, but under the uniform distribution it is typically $\Omega(\sqrt{n})$, as was shown in Erdős *et al.* (1999). Therefore, this theorem shows that the convergence rate is bounded by a function of n that is at worst exponential, and is typically superpolynomial.

2.7. Fast convergence

Theorem 1 gives an upper bound on the sequence length that suffices for accuracy for three different distance methods, but the upper bound is high. This is unsatisfactory, because we would like methods to converge more quickly to the true tree than those upper bounds would suggest. We therefore define fast convergence as follows:

Definition 6. *A method is said to be* **fast converging** *for the Jukes–Cantor model if for all fixed f, g with $0 < f \le g$, and all Jukes–Cantor trees T with $f \le \lambda_e \le g$ for all edges e, and all $\delta > 0$, there is a constant C that depends upon f, g, and δ, and a polynomial $p(n)$ so that if k exceeds $Cp(n)$ then the method recovers the true tree topology from sequences of length k with probability at least $1 - \delta$.*

Until recently, there were no proofs of fast convergence, and in fact, no bounds on the convergence rates of any method. However, in recent years, there have been several papers providing bounds on the convergence rates for various distance-based methods (Ambainis *et al.*, 1997; Erdős *et al.*, 1997a, 1999; Farach and Kannan, 1996), and the introduction of the first provably fast-converging methods (Csuros and Kao, 1999; Erdős *et al.*, 1997a,b, 1999; Cryan *et al.*, 1998).

2.8. Proving fast convergence

The fast-converging methods have a similar structure, in that they use only "close" relationships to determine the tree. The proofs of fast convergence differ somewhat, but there is in general a common idea behind the proofs, which suggest a common proof technique. We now introduce this proof technique.

Definition 7. *Let $\varepsilon_{ij} = |d_{ij} - \lambda_{ij}|$ and let $\varepsilon(q) = \max\{\varepsilon_{ij} : \min(d_{ij}, \lambda_{ij}) \le q\}$.*

If q is large, then $\varepsilon(q)$ will converge to 0 more slowly than if q is small, as we now show:

Theorem 2. *Let T be a Jukes–Cantor tree. For all $w > 0$, $\delta > 0$, and $y > 0$, there exists a constant C that depends upon δ and y, such that the sequence length that suffices for $\varepsilon(w) < y$ with probability at least $1 - \delta$ is $C \log n e^{O(w)}$.*

Proof. The proof follows from results obtained in the proofs of Theorems 8 and 9 from Erdős *et al.* (1999). The proof of Theorem 8 shows the following:

For all Jukes–Cantor model trees (T, λ), and for all $\delta > 0$, $q > 0$, $y > 0$, there exists a constant C that depends upon δ and y, such that if the sequence length k exceeds $C \log n e^{O(q)}$, then $Prob[|d_{ij} - \lambda_{ij}| < y \ \forall i, j$ with $\lambda_{ij} \le q] > 1 - \delta$.

Similarly the proof of Theorem 9 shows the following:

For all Jukes–Cantor model trees (T, λ), and for all $\delta > 0$, $q > 0$, $y > 0$, there exists a constant C that depends upon δ and y, such that if the sequence length k exceeds $C \log n e^{O(q)}$, then $Prob[|d_{ij} - \lambda_{ij}| < y \ \forall i, j$ with $d_{ij} \le q] > 1 - \delta$.

The proof of the theorem follows from these two observations.   ∎

Corollary 1. *Let M be a fixed phylogenetic method and assume that there exist functions $y(f)$, $A(g)$, and $F(g, n)$, so that for each Jukes–Cantor tree (T, λ) on n leaves with $0 < f \le \lambda_e \le g$ for all edges e,*

- $y(f) = \Theta(f)$,
- $F(g, n) = O(A(g) \log n)$, *and*
- *for all input dissimilarity matrices d, whenever* $\varepsilon[F(g, n)] < y(f)$ *then M is correct on input d.*

Then M is fast converging for the Jukes–Cantor model.

Proof. By Theorem 2, the sequence length that suffices for $\varepsilon(F(g, n)) < y(f)$ is $O(\log n \, e^{O(F(g,n))})$. But this is $O(\log n \, n^{O(A(g))})$, which is bounded by a polynomial in n since we have bounded g. ∎

Note that the smaller $A(g)$ is, the smaller the degree of the polynomial. Thus, a proof technique for establishing fast convergence under the Jukes–Cantor model is to show that functions such as y, A, and F exist.

The first methods that were proven fast converging were the Short Quartet Methods (Erdős *et al.*, 1997a,b, 1999), followed by the *Harmonic Greedy Triplets* method (Csuros and Kao, 1999), a method proposed by Cryan *et al.* (1998), and the fast-converging method in this paper. The techniques used to prove fast convergence in these papers are not alike, but the ones in Erdős *et al.* (1997b, 1999) can be restated as using the technique above, and we believe the other proofs can also be so restated.

The new fast-converging method we present is obtained by using a general technique for boosting the performance of phylogenetic methods, which we call the *Disk-Covering Method*, or DCM.

3. DCM–BUNEMAN, A FAST CONVERGING METHOD

The basic structure of DCM–Buneman is quite similar to the basic structure of the Short Quartet Methods, from which it is derived.

These methods have two phases, and take as input a dissimilarity matrix d. During the first phase a collection of trees is constructed, one for each $q \in \{d_{ij}\}$. The input to the reconstruction of T_q is the submatrix of d consisting only of those entries for which $d_{ij} \leq q$. Note that for $q < q'$, the input to the reconstruction of T_q is more *reliable* than the input to the reconstruction of $T_{q'}$, because $\varepsilon(q) \leq \varepsilon(q')$. On the other hand, the *amount* of data given as input to the reconstruction of $T_{q'}$ is more than the amount of data given as input to the reconstruction of T_q. Thus, the trees T_q will differ as q ranges over d_{ij}, because they will be based upon differing quantities and qualities of data, and the task of the second phase is to select, from the returned trees, one that made best use of the input.

The particular instantiations of the two phases differ between these different fast-converging methods. In the Short Quartet Methods, when reconstructing T_q, we infer a tree on each quartet of leaves if its maximum interleaf distance is bounded by q. A unique tree on the entire set of leaves is then sought that agrees with every quartet tree in the input; if no such tree could be found, then T_q was left undefined. The analysis of the Short Quartet Method showed that when the sequences were long enough, then for all q such that T_q is defined, T_q would be the true tree with high probability. The Short Quartet Methods are polynomial time and fast converging, but by design either reconstruct the true tree or fail (with high probability) to reconstruct anything.

The fast-converging method that we have designed is very similar to the Short Quartet Method, but it has two distinct advantages: first, it always reconstructs a tree, and second, it has much better performance in experimental performance studies. Thus, while it does not provide any theoretical advantage over the Short Quartet Methods, it provides empirical advantages. Furthermore, the fast converging method we present is the result of applying the Disk-Covering Method (or DCM) to a simple polynomial time method. DCM is a very general phylogenetic method booster, which can be used with any phylogenetic reconstruction method. Our experimental performance study shows that DCM boosting improves accuracy at realistic sequence lengths of many distance-based methods.

3.1. Phase I of DCM–Buneman

We now describe how we compute each T_q, as q ranges over the entries of d_{ij}. Much of the algorithm for computing T_q is based upon graph-theoretic concepts and results, and so we will begin with some graph theory.

3.1.1. Graph-theoretic material.

Definition 8. *A graph is* **triangulated** *(or "chordal") if no subset of nodes induces a cycle of size four or more.*

There are many theoretical results established about triangulated graphs, and many NP-hard problems become solvable in polynomial time when restricted to triangulated graphs. The following can be found in Buneman (1974) and Golumbic (1980).

Lemma 2. *Every triangulated graph is the intersection graph of subtrees of a tree, and vice versa. Every triangulated graph G has a* **simplicial elimination ordering***, $v_1, v_2, \ldots, v_n$; this is an ordering of the nodes so that the set $X_i = \{v_j : j > i$ and $(v_i, v_j) \in E\}$ forms a clique [i.e., for all $\{v_k, v_l\} \subseteq X_i, (v_k, v_l) \in E$]. The* **maximal cliques** *(cliques that cannot be enlarged by the addition of any further vertices) in G are of the form $\{v_i\} \cup X_i$; hence there are at most n maximal cliques and these can be found in $O(n^2)$ time. Given a triangulated graph, a simplicial elimination ordering for the graph can be found in $O(n^2)$ time, and from it the maximal cliques can also be found in that time.*

We now define threshold graphs.

Definition 9. *Let d be an $n \times n$ dissimilarity matrix (i.e., a symmetric matrix that is 0 on the diagonal) and let q be any real number. The* **threshold graph** *$Thresh(d, q)$ is defined as follows. The vertex set is $1, 2, \ldots, n$ and (i, j) is an edge if and only if $d_{ij} \leq q$.*

Lemma 3. *If d is an additive matrix, then $Thresh(d, q)$ is triangulated.*

Proof. (We are not the first to observe this fact, but we provide a proof because it is extremely simple.) By Lemma 2, to prove a graph is triangulated, it suffices to prove that it is isomorphic to such an intersection graph. Let d be an arbitrary additive matrix, and let (T, w) be the edge-weighted tree associated uniquely to d. Let $q > 0$ be given. Add intermediate vertices to the edges of T and reweight the edges so that the path distances between leaf pairs are unchanged, but so that for every pair of leaves u, v in T if $d_{u,v} > q/2$ then there is a node x in the enlarged tree T' so that $d^{T'}(u, x) = q/2$ and $d^{T'}(x, v) = d^{T'}(u, v) - q/2$. Now let X_u denote the subtree of T' of distance at most $q/2$ of u. Note that $X_u \cap X_v \neq \emptyset$ if and only if $d_{u,v} \leq q$, and that the threshold graph $Thresh(d, q)$ is identical to the intersection graph of the X_u, as u ranges over the leaves of T. Consequently $Thresh(d, q)$ is triangulated. ∎

3.1.2. Constructing T_q. We now describe how we compute a particular T_q.

- **Step 1: Compute** $Thresh(d, q)$.
- **Step 2: Triangulate** $Thresh(d, q)$: Add edges to $Thresh(d, q)$ to make it triangulated, while minimizing the weight of the largest edge added. (The weight of edge i, j is given by $d_{i,j}$.) We call the resultant triangulated graph $Thresh^*(d, q)$.
- **Step 3: Compute Buneman Trees for all maximal cliques in** $Thresh^*(d, q)$. Each maximal clique defines a subset of the taxa (for example, as represented by the DNA sequences at the associated leaves of the tree). We compute the Buneman Tree for each such subset of the taxa.
- **Step 4: Merge the subtrees into a supertree.**

We now discuss the specific techniques we use to implement the various steps, and their computational complexity. Computing the threshold graph is polynomial time, but minimally triangulating the threshold graph is NP-hard (McMorris *et al.*, 1994). [In practice, the triangulation can be obtained using greedy heuristics, and these will generally perform well when $Thresh(d, q)$ is close to triangulated. Because we compute d using the Jukes–Cantor distance calculation, for long sequences d is close to additive, and so $Thresh(d, q)$ will be close to triangulated. Thus, polynomial time techniques can be used to triangulate the threshold graph without too much loss in performance, as our experiments suggest.] Calculating maximal cliques in triangulated graphs is polynomial time (Golumbic, 1980). The merger of the subtrees into a supertree is the only remaining task, but we show that we can accomplish this merger in polynomial time and ensure accuracy in the supertree, when the subtrees are correct and based upon a large enough threshold graph.

3.1.3. Supertree Construction Algorithm (SCA). The construction of the supertree from the subtrees is an interesting problem, because we would like to ensure that if all the subtrees are correct (in that the true tree induces these subtrees when restricted to the subsets of leaves) then a supertree consistent with all the subtrees should be returned. However, this generalizes to the Subtree Compatibility Problem, which is NP-complete (Steel, 1992). Thus we will need a special case of the Subtree Compatibility Problem if we are to solve this problem exactly and in polynomial time. (Note that while we were willing to accept a suboptimal triangulation, we are not willing to suboptimally construct the supertree, because we need to obtain the true tree, if possible, and not an incorrect tree.) Also, even if the subtrees are correct, they may not uniquely define the supertree (i.e., many different trees may be consistent with the set of subtrees). For this reason, the set of subsets must be defined with care, with respect both to computational consequences as well as to uniqueness of the supertree compatible with the subtrees.

We now describe how we compute a supertree from a set of subtrees. Our algorithm has the nice property that when applied to properly defined inputs, it is guaranteed to reconstruct a *unique* supertree consistent with the inputs.

We assume that the input to the supertree construction algorithm is a triangulated graph G and a collection of subtrees, one for each maximal clique in G. We let T_C denote the subtree for clique C. If G is not connected, then the algorithm produces a forest (i.e., a tree on each component of G); thus we will assume that G is connected.

Stage I: Preprocessing: First obtain a simplicial elimination ordering $v_1, v_2, \ldots, v_n$ for G. Compute $C_i = \{v_i\} \cup X_i$, where $X_i = \Gamma(v_i) \cap \{v_{i+1}, v_{i+2}, \ldots, v_n\}$ is the set of neighbors of v_i that follow it in the simplicial elimination ordering. The set of maximal cliques is a subset of the set $C_1, C_2, \ldots, C_n$. For each C_i, find a maximal clique C containing C_i and compute a tree for C_i by deleting the leaves in $C - C_i$ from T_C. In this way, we associate a tree t_i with every C_i.

Stage II: Construct the tree: For $i = n - 4, n - 3, \ldots, 1$, compute the tree T_i formed by merging t_i and T_{i+1}, using the *Strict Consensus Subtree Merger* method.

Strict Consensus Subtree Merger. The Strict Consensus Subtree Merger method *contracts a minimum set of edges in each tree* in order to make them identical on the subtrees they induce on X. The strict consensus (Day, 1995) of the induced subtrees is defined to be the maximally resolved tree that is a common contraction of the two subtrees. We will call this subtree on X the **backbone**. Merging the two trees together is then achieved by *attaching* the pieces of each tree appropriately to the different edges of the backbone.

It is worth noting that the strict consensus subtree merger of two trees, while it always exists, may not be unique. In other words, it may be that some piece of each tree attaches onto the same edge of the backbone. We call this a **collision**.

For example, in Fig. 1, the common intersection of the two leaf-sets is $X = \{1, 2, 3, 4\}$, and the strict consensus of the two subtrees induced by X is the 4-star. This is the "backbone," it has four edges, and there is a collision on the edge of the backbone incident to leaf 4, but no collision on any other edge.

Collisions are problematic, as the Strict Consensus Subtree Merger will potentially introduce false edges or lose true edges when they occur. However, as we will show, when the subtrees are correct and the threshold is selected to be large enough, then there are no collisions. In this case, the true tree is reconstructed.
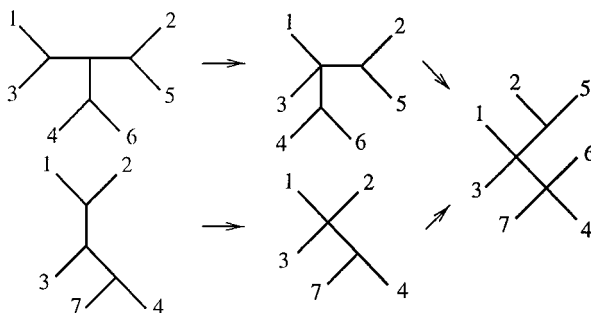

FIG. 1. Merging two trees together, by first transforming them (through edge contractions) so that they induce the same subtrees on their shared leaves.

Theorem 3. *Let G be a triangulated graph with n vertices, $\mathcal{T}$ be the associated set of trees on each maximal clique, and assume that G and $\mathcal{T}$ are given as input. Then SCA takes $O(n^2)$ time.*

Proof. The proof of this is straightforward. Computing the perfect elimination ordering takes $O(n^2)$ time, and the rest follows from the observation that merging two trees takes $O(n)$ time since computing the strict consensus of two trees takes $O(n)$ time (Day, 1995). ∎

3.1.4. Conditions under which T_q is the true tree. We now describe the conditions under which the reconstructed tree T_q is the true tree. We begin with some definitions.

Definition 10. *Let (T, w) be a binary tree edge weighted by $w : E(T) \to \mathbb{R}^+$, and leaf labeled by the set $S = \{1, 2, \ldots, n\}$ of taxa. Let λ be the additive distance matrix associated to T. Let e be an edge in T that is not incident to a leaf of T. Around e, there are four subtrees, A, B, C, and D. Let a, b, c, and d be four leaves in each of the four subtrees A, B, C and D, respectively, closest to e [where the distance between nodes p and q is measured as $\sum_{e \in P_{pq}} w(e)$]. We call $\{a, b, c, d\}$ a* **short quartet** *around e, and the collection of all short quartets around internal edges of T is denoted by $Q_{\text{short}}(T)$. The maximum $\lambda_{i,j}$ such that i and j are in a short quartet together is called the λ-width(T). The graph G_{sq} on vertex set $S = \{1, 2, \ldots, n\}$ is defined by $(i, j) \in E(G_{sq})$ if i and j are in some short quartet together.*

Theorem 4. *Let T be a fixed leaf-labeled tree, let G be a triangulated graph such that $G_{sq} \subseteq G$, and assume that the Buneman Tree Method applied to each maximal clique in G reconstructs the correct subtree (i.e., it reconstructs the subtree of T induced by the maximal clique). Let $\mathcal{T}$ be the collection of Buneman Trees on maximal cliques of G, and let T^* be the tree obtained by applying SCA to $(G, \mathcal{T})$. Then $T^* = T$.*

Proof. Let T be a tree whose leaves are labeled by $S = \{v_1, v_2, \ldots, v_n\}$. Let G be a triangulated graph on S, and let $\mathcal{T} = \{T_A\}$, where T_A is a tree on leaf set A for every maximal clique A in G. Let $\sigma = \{v_1, v_2, \ldots, v_n\}$ be a simplicial elimination ordering for G. Recall the definitions of t_i, T_i, X_i from the description of SCA. The proof proceeds by showing that $T \mid \{v_i, v_{i+1}, \ldots, v_n\} = T_i$ for all i.

The base case requires that we show that $T_{n-3} = T \mid \{v_{n-3}, v_{n-2}, v_{n-1}, v_n\}$, but this follows trivially since we assume T_{n-3} is true. Now assume that $T_i = T \mid \{v_i, v_{i+1}, \ldots, v_n\}$ for some $i \in \{1, 3, \ldots, n - 4\}$. Consider X_{i-1}. Note that by definition $X_{i-1} = \Gamma(v_{i-1}) \cap \{v_i, v_{i+1}, \ldots, v_n\}$, and that X_{i-1} forms the leaf set of the backbone of the strict consensus merger of t_{i-1} and T_i. Also X_{i-1} is a clique, and so by assumption $T_i \mid X_{i-1} = t_{i-1} \mid X_{i-1}$. Consequently there is no edge contraction when we compute the backbone. To complete the proof that $T_{i-1} = T \mid \{v_{i-1}, v_i, \ldots, v_n\}$ we need only show that there is no collision formed by the merger of the two trees.

There can be a collision only if the backbone contains an edge onto which both v_{i-1} and some other $v_j \notin X_i$ attach. Let e be the edge onto which v_{i-1} attaches, and suppose there is a collision on this edge e. Thus, some subtree t' of T_i attaches onto e. (Note that in this case, these are true attachments in the sense that v_{i-1} and t' also attach to the path associated to e in the true tree.) Let the leaf set of T' be Y, and note $Y \subseteq \{v_i, v_{i+1}, \ldots, v_n\} - X_{i-1}$. Let P be the path in T corresponding to the edge e and let its endpoints be a and b. Consider the subtree T_0 of T obtained by deleting all the nodes in T that are separated from a by the deletion of b, or vice versa, and let $A_{a,b}$ be the leaves of T_0. In other words, T_0 consists of the path P and all subtrees of T that attach to interior nodes of P. The following conditions are then true:

1. $v_{i-1} \in A_{a,b}$ and all leaves in t' are also in $A_{a,b}$.
2. G_{sq} restricted to $A_{a,b}$ is path connected.
3. $X_{i-1} \cap A_{a,b} = \emptyset$.

The proofs of (1) and (3) follow from the fact that T_i and t_{i-1} are correct. Fact (2) can be proven by induction, and uses the fact that every short quartet in the true tree T induces a four-clique in G.

Now, let P' be a path lying in $G_{sq} \cap A_{a,b}$ from v_{i-1} to some node in Y. Let y be the first node from Y on P'; by definition, $y \notin X_{i-1}$. By (3), the path from v_{i-1} to y lies entirely in $v_1, v_2, \ldots, v_{i-1}$, so that $(v_{i-1}, y) \in E(G)$ [this follows from facts about simplicial elimination orderings, see (Golumbic, 1980)]. Consequently $y \in \Gamma(v_{i-1}) \cap \{v_i, v_{i+1}, \ldots, v_n\} = X_{i-1}$. However, this contradicts our earlier conclusion that $y \notin X_{i-1}$. ∎

We now describe another condition under which T_q is guaranteed to be the true tree.

Theorem 5. *Let (T, λ) be a Jukes–Cantor tree, d the input dissimilarity matrix, and G a triangulated graph with $G_{sq} \subseteq G$; thus every short quartet in T induces a four-clique in G. Furthermore assume that for every short quartet $\{i, j, k, l\}$ in T that the Buneman Tree on $\{i, j, k, l\}$ is $T \mid \{i, j, k, l\}$ (i.e., the correct tree). If the supertree construction algorithm applied to Buneman Trees on the maximal cliques of G produces a binary tree T', then $T' = T$.*

Proof.   We begin by citing the result proved by Erdős *et al.* (1999).

Lemma 4. *Let (T, w) be an edge-weighted tree leaf labeled by S and let T' be a tree also leaf labeled by S. If for every short quartet $\{i, j, k, l\}$ in T, the tree T' induces the same tree on $\{i, j, k, l\}$ as T, then $T = T'$.*

Now suppose that the supertree construction algorithm produces a binary tree T'. In this case, every Buneman Tree on every maximal clique is binary, and there are no collisions during the merger. Therefore, T' agrees with T for every short quartet of T. Then by the above stated lemma, $T = T'$.   ■

3.2. Phase II of DCM–Buneman

In the previous sections we showed how to compute each T_q, and also established two conditions under which T_q would be the true tree. We now show how we select a particular tree T_q to return as the output of the DCM–Buneman Method. We select T_q using the following rule:
Return the most resolved T_q (i.e., the one with the most internal edges), and if there are more than one such tree, then return the one associated to the largest q.

3.3. Performance guarantees of DCM–Buneman

Theorem 6. *Let T be a Jukes–Cantor model tree and let $0 < f \leq \lambda_e \leq g$ for all edges e. Recall that λ-width(T) is the largest λ-distance between two leaves in a short quartet. Then DCM–Buneman is accurate on input d if $\varepsilon(\lambda\text{-width}(T) + 3f/2) < f/2$.*

Proof.   Let $q = \lambda\text{-width}(T) + 3f/2$ and assume that $\varepsilon(q) < f/2$. Now consider the threshold graph $Thresh[d, \lambda\text{-width}(T) + f/2]$. Since $\varepsilon(q) < f/2$, the following are true:

- $G_{sq} \subseteq Thresh[d, \lambda\text{-width}(T) + f/2]$.
- $Thresh[d, \lambda\text{-width}(T) + f/2] \subseteq Thresh[\lambda, \lambda\text{-width}(T) + f]$.
- $Thresh[\lambda, \lambda\text{-width}(T) + f] \subseteq Thresh[d, \lambda\text{-width}(T) + 3f/2]$.

Since λ is additive, $Thresh[\lambda, \lambda\text{-width}(T) + f]$ is triangulated by Lemma 3. Hence, the minimal triangulation of $Thresh[d, \lambda\text{-width}(T) + f/2]$ is a subgraph of $Thresh[d, \lambda\text{-width}(T) + 3f/2]$.

Consequently the Buneman Tree method computes the correct tree for every maximal clique in $Thresh^*[d, \lambda\text{-width}(T) + f/2]$. By Theorem 4, the strict consensus subtree merger reconstructs the true tree.

Hence there is at least one threshold p for which T_p is the true tree, and it is $p = \lambda\text{-width}(T) + f/2$. In Phase II of DCM–Buneman we select the most resolved tree, and if there is more than one equally resolved tree, we select the tree associated to the largest threshold. So suppose there is a $p' \geq p$ such that $T_{p'}$ is also binary. By Theorem 5, they are identical, and both equal T. Thus if $\varepsilon(q) < f/2$ where $q = \lambda\text{-width}(T) + 3f/2$, then DCM–Buneman reconstructs the true tree.   ■

Theorem 7. *DCM–Buneman is fast converging for the Jukes–Cantor model.*

Proof.   By Theorem 6 and Corollary 1, all we need to establish is that $\lambda\text{-width}(T) + 3f/2 = O(g \cdot \log n)$. Clearly $\lambda\text{-width}(T) + 3f/2 = O(\lambda\text{-width}(T))$. Then $\lambda\text{-width}(T) = O(g \cdot \log n)$, as was shown in Erdős *et al.* (1999). Thus, DCM–Buneman is fast converging.   ■

A comparison between the convergence rates of the Buneman Tree Method and DCM–Buneman is interesting. Let (T, λ) be a fixed Jukes–Cantor model tree. The Buneman Tree Method is statistically consistent for the Jukes–Cantor model of evolution, but the only established upper bounds on the convergence rate indicate that it converges from sequence lengths that grow exponentially in $\lambda_{max} = \max_{ij} \lambda_{ij}$, and so the Buneman

Tree Method is not likely to be fast converging (see discussion following Theorem 1). However, for the same tree, the convergence rate of the DCM–Buneman method is much faster, and in fact DCM–Buneman is fast converging. The difference in convergence rates is obtained through restricting the attention to the small distances in the data set, rather than using all the distances. Thus DCM–Buneman is a *boosted* version of the Buneman method.

4. DCM–BOOSTING OTHER METHODS

DCM–Buneman is actually a special instantiation of a very general two-phase technique (called the Disk-Covering Method, or DCM), which can be used in conjunction with any phylogenetic method. In the first phase, we construct a tree T_q for each $q \in d_{ij}$, and in the second phase, we compute a tree on the entire set of taxa, by taking a *consensus* of the trees T_q. When used in conjunction with the phylogenetic method M, we call this **DCM–M**. The method M is used to reconstruct the subtrees on the maximal cliques of the triangulated threshold graph; and second, we design the second phase (taking the consensus of the trees T_q) to optimize the performance of the method.

4.1. Phase I

We now describe how we perform the first phase, in which a tree T_q is computed for each $q \in \{d_{ij}\}$. Let S be the input set of sequences, and let $q \in \{d_{ij}\}$ be the selected threshold. Let M be the base phylogenetic method.

- We construct the threshold graph $Thresh(d, q)$. We *triangulate* each component of $Thresh(d, q)$, minimizing the weight of the largest edge added, thus obtaining a triangulated graph $Thresh^*(d, q)$.
- We compute the maximal cliques in $Thresh^*(d, q)$, and compute a tree on each maximal clique using the method M.
- We apply the Supertree Construction Algorithm to the set of trees defined for the maximal cliques, obtaining a tree T_q if $Thresh^*(d, q)$ is connected, and otherwise obtaining a forest F_q.

This first phase can be modified to allow for the triangulation of the threshold graph to be done suboptimally, and in practice this is what we have done (greedy triangulations affect the performance of the method only very slightly, as our experiments show). The construction of a supertree from subtrees can be implemented in various ways, as well. We have elected to be quite conservative, and hence our Supertree Construction Algorithm employs the Strict Consensus Subtree Merger technique; however, this can also be modified.

4.2. Phase II

In the second phase, we take the trees T_q we have computed in Phase I and infer a consensus of these trees.

We have experimented with using DCM boosting in conjunction with several distance based methods, including Neighbor-Joining (NJ), the Agarwala *et al.* (1996) algorithm that 3-approximates the L_∞-nearest tree, and the Buneman Tree method. Our experimental studies for all these methods indicate that for almost every *small q*, the tree T_q has very low false-positive rates, typically close to 0. Consequently, almost all T_q are either contractions of the true tree or close to being contractions of the true tree. (This is perhaps not at all surprising, since we designed the merger using the strict consensus technique, and this collapses edges that are not supported by every subtree!) This suggests the following implementation of Phase II: *take all the trees T_q, and compute the asymmetric median tree of these trees* (Phillips and Warnow, 1996). We now define this consensus technique.

Definition 11.   *The **asymmetric median tree** of a set of leaf-labeled trees $\mathcal{T} = \{T_1, T_2, \ldots, T_p\}$ computes a tree T such that $C(T) \subseteq \cup_i C(T_i)$, and such that if each $c \in C(T)$ is weighted by the number of trees T_i that contain c, then $w(T) = \sum_{c \in C(T)} w(c)$ is maximum.*

The idea behind the asymmetric median tree is that when the input trees have low false-positive rates, the asymmetric median tree method recovers as many of the true edges as possible. Computing the asymmetric median tree is NP-hard, and so in practice we have implemented this using a greedy strategy (which is not guaranteed to find an optimal solution). This greedy technique nevertheless has good empirical performance,

as our experimental study shows. This implementation of Phase II can be used with DCM–Buneman as well, but under these conditions we do not have provable performance guarantees.

5. EXPERIMENTAL RESULTS

We briefly describe a small portion of our experimental performance analysis. For additional performance results based upon simulating sequence evolution, see Huson *et al.* (1998a).

5.1. Model trees and simulated datasets

The basic model tree that we use has its topology and rates of evolution along the edges based upon reconstructions of the African Eve data set (Maddison *et al.*, 1992) restricted to its human mitochondrial DNA sequences. We then scaled the rates of evolution on this basic model tree up, to produce a number of different trees on which there were high evolutionary rates. We used this larger set of model trees to generate several hundred different sets of DNA sequences using the *ecat* simulator (Rice, 1997), and using the Jukes–Cantor model of evolution. Later, we will report on the performance of DCM boosting on one particular scaled up version (in which the largest probability of change on any edge equal to 0.48) of this basic tree.

5.2. Distance calculations

We computed Jukes–Cantor distance matrices for each data set. On some data sets the rate of evolution was high enough that some pairs of sequences differed in 75% or more of their positions. For such pairs of sequences, the standard Jukes–Cantor distance calculation cannot be used since the log cannot be computed. For these pairs, we defined d_{ij} using the following version of the "large value replacement" technique (Swofford *et al.*, 1996). We computed the maximum Jukes–Cantor distance, multiplied that value by the number n of leaves in the matrix, and replaced all undefined values by this large number. These matrices were then input to six different distance-based methods: Neighbor-Joining (NJ), the Agarwala *et al.* method, the Buneman Tree, and the DCM-boosted versions of these three methods.

5.3. Performance evaluation criteria

We explored performance with respect to accuracy of the topology recovered by each method, by comparing the reconstructed tree to the model tree. Recall that this accuracy is quantified by examining false-negative (FP) rates and false-positive (FN) rates (see Definition 2).

Recall that sequence lengths beyond 5000 nucleotides are considered unusually long for tree reconstruction, and that in general convergence to the true tree or acceptable error rates within 1000 nucleotides is thus the critical test of performance (all these methods will converge to the true tree given long enough sequences, since they are all statistically consistent under this model of evolution! The question is *at what rate*). Also, for systematic biology purposes, error rates below 5% can probably be tolerated, though of course this will depend upon the tree. Hence, we examined these experiments with the following specific questions in mind:

- *At what sequence length do we get an error rate below 5%?*
- *At what sequence length (if any) do we recover the true tree reliably?*
- *How well do the different methods do when restricted to typical length sequences (between 200 and 1200 nucleotides)?*

Since we are interested in how DCM boosting affects performance, we will specifically address how DCM-boosted methods differ from their base methods with respect to these three questions.

5.4. Summary of experimental results

We report on the results of a set of experiments on the African Eve tree with rates of evolution scaled up so that the maximum probability of change was 0.48. This model tree is a good example of how DCM boosting affects performance when the tree is a difficult one to reconstruct, due to the combination of large numbers of taxa and high divergence.

Here are some of the basic observations about the performance of these six methods on this tree.

1. DCM–NJ significantly outperformed all other methods,
2. Even the worst DCM-boosted method (DCM–Buneman) outperformed the best base method (Neighbor-Joining).
3. Typical reductions in FN error rates obtained via DCM boosting for typical length sequences (200–1200 nucleotides) were very high for all three base methods.

We now discuss the comparisons between each method and its DCM-boosted version. In the graphs in Figs. 2–7, we plot the false-positive (FP) and false-negative (FN) error rates of each method as a function of the sequence length, where the sequence lengths range from 100 to 12,800. Each point plotted indicates the average over five independent experiments.

5.5. Buneman vs. DCM–Buneman

We summarize the results for the Buneman Tree method and DCM–Buneman in Figs. 2 and 3. Note that the Buneman Tree method returns trees with no false positives and either 100% false negatives or very close to 100% false negatives. This means that the Buneman Tree method returns a "star" for almost all data sets generated under these conditions. DCM boosting reduces FN error rates at all sequence lengths, although it very slightly elevates the FP error rates (from 0% to about 3%). Note also the surprising fact that DCM–Buneman achieves FN error rates below 5% (the "acceptable" level of error) at 10,000 nucleotides, but that Buneman still has 98% FN error rates even at 12,800 nucleotides!

5.6. Agarwala vs. DCM–Agarwala

In Figs. 4 and 5 we depict the results for the Agarwala *et al.* algorithm. Here, DCM boosting significantly reduces all error rates at all sequence lengths we examined. Also, DCM–Agarwala attains acceptably low (below 5%) FN error rates at 1000 nucleotides (at which point the Agarwala algorithm has an FN error rate
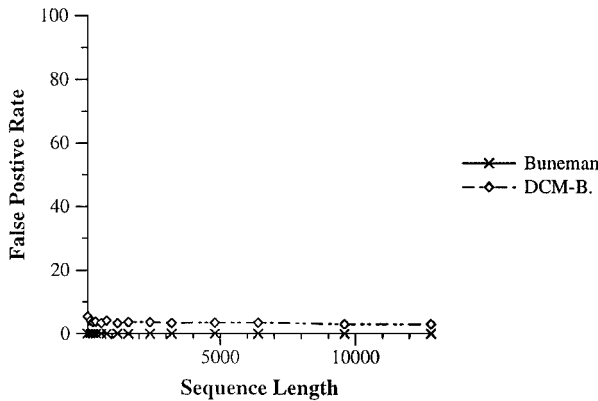

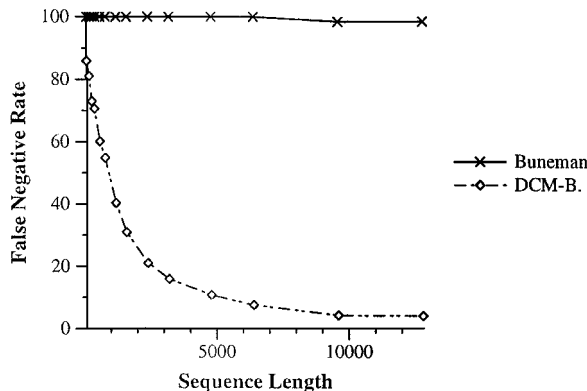**FIG. 2.** Buneman vs. DCM–Buneman, false positives.

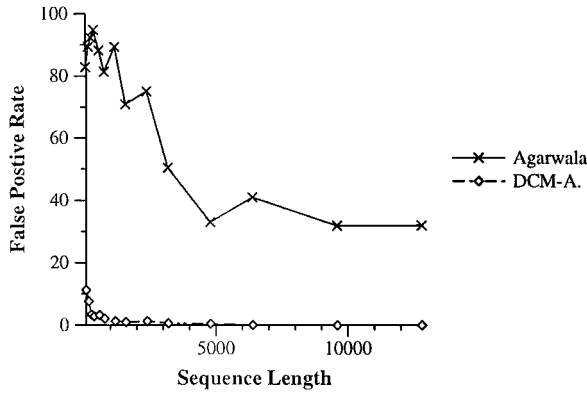**FIG. 3.** Buneman vs. DCM–Buneman, false negatives.

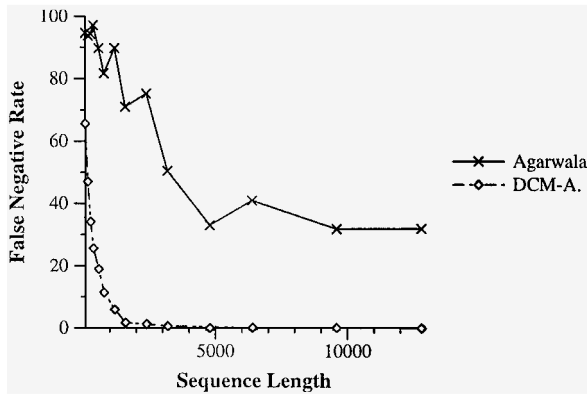**FIG. 4.**   Agarwala vs. DCM–Agarwala, false positives.

FIG. 5.   Agarwala vs. DCM–Agarwala, false negatives.

of 80%!). Also, the Agarwala algorithm does not achieve FN error rates below 30% at *any* sequence length we examined, while DCM–Agarwala obtains the true tree at all sequence lengths beyond 3000.

5.7. NJ vs. DCM–NJ

Finally, we summarize the results for Neighbor-Joining in Figs. 6 and 7. DCM–NJ obtains an acceptable FN error rate (below 5%) at just 250 nucleotides, at which point NJ has more than 60% FN errors! Also, NJ does not achieve acceptably low FN error rates until 8000 nucleotides. Finally, DCM–NJ recovers the tree at all sequence lengths beyond 900, while NJ does not recover the true tree even from sequences of length 12,800.
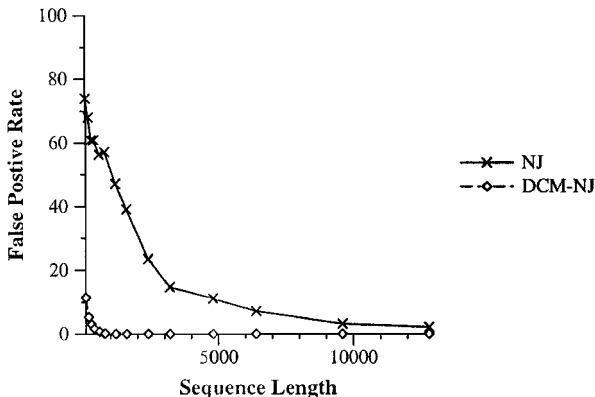
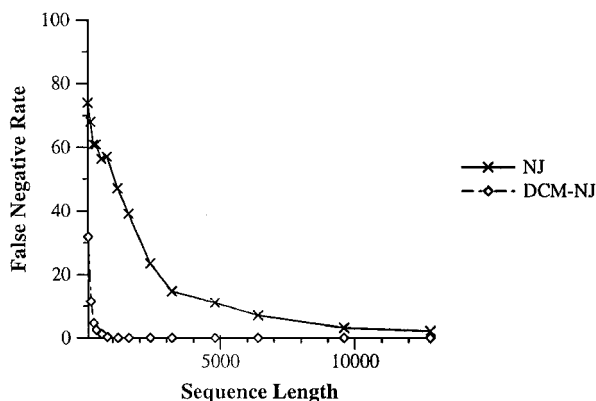**FIG. 6.**   Neighbor-Joining (NJ) vs. DCM–NJ, false positives.

FIG. 7. Neighbor-Joining (NJ) vs. DCM–NJ, false negatives.

5.8. Summary

Thus, the theoretical advantages we have established of DCM–Buneman over Buneman show up in the experimental performance study, but it is worth noting that although we have no theoretical results for the impact of DCM boosting on other methods, the improvement obtained through DCM boosting is very noticeable on NJ and on Agarwala. This improvement over NJ is particularly noteworthy, since any significant improvement over NJ was previously achievable only by using exponential time methods (Huson *et al.*, 1998b; Rice and Warnow, 1997).

6. RELATED WORK

The first analytical bounds on the convergence rates of any methods under standard Markov models of evolution were obtained by Farach and Kannan (1996), Erdős *et al.* (1999), and Atteson (1997). These bounds established the upper bounds on the convergence rates that we gave in this paper for the classical methods we discussed. However these bounds were not polynomial in n.

The polynomial time Short Quartet Methods of Erdős *et al.* (1997b, 1999) were the first provably fast-converging methods, followed by the DCM–Buneman method (originally introduced in Huson *et al.*, 1998a) the *Harmonic Greedy Triplets* method (Csuros and Kao, 1999), and the method developed by Cryan *et al.*, 1998. It is possible that other methods are fast converging, but to our knowledge no proofs of fast convergence exist for other methods under any Markov models.

There are several interesting open problems related to convergence rates. Finding tighter bounds on the convergence rates of classical methods is very desirable, as their performance in simulation studies suggests the upper bounds are probably loose. Second, while we have bounds for the false negatives of some distance methods, expressed as a function of the error in the estimations of the tree distances, we do not have any corresponding bounds for the false negatives of any fast-converging methods.

7. ACKNOWLEDGMENTS

REFERENCES

Agarwala, R., Bafna, V., Farach, M., Narayanan, B., Paterson, M., and Thorup, M. 1996. On the approximability of numerical taxonomy: Fitting distances by tree metrics. *Proc. 7th Annu. ACM-SIAM Symp. Discrete Algorithms* 365–372.

Ambainis, R., Desper, R., Farach, M., and Kannan, S. 1997. Nearly tight bounds on the learnability of evolution. *Proc. 38th Annu. IEEE Symp. Found. Comput. Sci.* 524–533.

Atteson, K. 1997. The performance of neighbor-joining algorithms of phylogeny reconstruction, 101–110. *In* Jiang, T., and Lee, D., eds., *Lecture Notes in Computer Science, 1276*, Springer-Verlag, Berlin. COCOON '97.

Bandelt, H.-J., and Dress, A. 1992. A canonical decomposition theory for metrics on a finite set. *Adv. Math.* 92, 47–105.

Berry, V., and Gascuel, O. 1997. Inferring evolutionary trees with strong combinatorial evidence, 111–123. *In* Jiang, T., and Lee, D., eds., *Lecture Notes in Computer Science, 1276*, Springer-Verlag, Berlin. COCOON '97.

Buneman, P. 1971. The recovery of trees from measures of dissimilarity, 387–395. *In Mathematics in the Archaeological and Historical Sciences*, Edinburgh University Press, Edinburgh.

Buneman, P. 1974. A characterization of rigid circuit graphs. *Discrete Math.* 9, 205–212.

Cryan, M., Goldberg, L.A., and Goldberg, P.W. 1998. Evolutionary trees can be learned in polynomial time in the two-state general markov model. *Proceedings of FOCS*, 436–445.

Csuros, M., and Kao, M.-Y. 1999. Recovering evolutionary trees through harmonic greedy triplets. *Proc.* SODA'99, 261–268.

Day, W. 1995. Optimal algorithms for comparing trees with labelled leaves. *J. Class.* 2, 7–28.

Dobzhansky, T. 1993. Nothing in biology makes sense except in the light of evolution. *Am. Biol. Teacher* March, 125–129.

Erdős, P.L., Steel, M.A., Székely, L.A., and Warnow, T. 1997a. Constructing big trees from short sequences. In Goos, G., Hartmanis, J., and van Leeuwen, J., eds., *Lecture Notes in Computer Science*, volume 1256. ICALP'97, 827–837.

Erdős, P.L., Steel, M.A., Székely, L.A., and Warnow, T. 1997b. A few logs suffice to build (almost) all trees II. *Theor. Comp. Sci.* 221, 77–118.

Erdős, P.L., Steel, M.A., Székely, L.A., and Warnow, T. 1999. A few logs suffice to build (almost) all trees I. *Random Struct. Algorithms* 14(2), 153–184.

Farach, M., and Kannan, S. 1996. Efficient algorithms for inverting evolution. *Proc. 28th Annu. ACM Symp. Theory Comput. J. ACM.* In press; *STOC Proceedings*, 230–236.

Farach, M., Kannan, S., and Warnow, T. 1995. A robust model for finding optimal evolutionary trees. *Algorithmica* 13(1), 155–179. (Special issue on Computational Biology.)

Felsenstein, J. 1988. Phylogenies from molecular sequences: Inference and reliability. *Annu. Rev. Genet.* 22, 521–565.

Golumbic, M. 1980. *Algorithmic Graph Theory and Perfect Graphs.* Academic Press, New York.

Gusfield, D. 1991. Efficient methods for multiple sequence alignment with guaranteed error bounds. Technical Report CSE 91-4, Computer Science Division, UC Davis.

Gusfield, D., and Wang, L. 1996. New uses for uniform lifted alignments. Technical Report CSE 96-4, Computer Science Division, UC Davis.

Hein, J. 1989. A new method that simultaneously aligns and reconstructs ancestral sequences for any number of homologous sequences, when the phylogeny is given. *Mol. Biol. Evol.* 6, 649–668.

Hillis, D. 1996. Inferring complex phylogenies. *Nature* (*London*) 383, 130–131.

Hillis, D., Huelsenbeck, J., and Cunningham, C. 1994. Application and accuracy of molecular phylogenies. *Science* 264, 671–677.

Huelsenbeck, J. 1995. Performance of phylogenetic methods in simulation. *Syst. Biol.* 44, 17–48.

Huelsenbeck, J., and Hillis, D. 1993. Success of phylogenetic methods in the four-taxon case. *Syst. Biol.* 42, 247–264.

Huson, D., Nettles, S., Parida, L., Warnow, T., and Yooseph, S., 1998a. The disk-covering method for tree reconstruction. *Proc. "Algorithms and Experiments" (ALEX98), Trento* 62–75.

Huson, D., Nettles, S., Rice, K., and Warnow, T. 1998b. Hybrid tree reconstruction methods. *Proc. "Workshop Algorithm Engineer," Saarbrücken* 172–192.

Jukes, T., and Cantor, C. 1969. Evolution of protein molecules, 21–132. *In* Munro, H., ed., *Mammalian Protein Metabolism.* Academic Press, New York.

Kuhner, M., and Felsenstein, J. 1994. A simulation comparison of phylogeny algorithms under equal and unequal evolutionary rates. *Mol. Biol. Evol.* 11, 459–468.

Maddison, D.R., Ruvolo, M., and Swofford, D.L. 1992. Geographic origins of human mitochondrial DNA: Phylogenetic evidence from control region sequences. *System. Zool.* 41, 111–124.

McMorris, F., Warnow, T., and Wimer, T. 1994. Triangulating vertex-colored graphs. *SIAM J. Discrete Math.* 7(2), 296–306.

Phillips, C., and Warnow, T. 1996. The asymmetric median tree: A new model for building consensus trees. *Discrete Appl. Math.* 71, 311–335. (Special Issue on Computational Molecular Biology.)

Purvis, A., and Quicke, D. 1997. Building phylogenics: Are the big easy? *Trends Ecol. Evol.* 12(2), 49–50.

Rice, K. 1997. ECAT, an evolution simulator. http://www.cis.upenn.edu/∼krice.

Rice, K., and Warnow, T. 1997. Parsimony is hard to beat!, 124–133. *In* Jiang, T., and Lee, D., eds., *Lecture Notes in Computer Science, 1276*, Springer-Verlag, Berlin. COCOON '97.

Rost, B., and Sander, C. 1993. Prediction of protein secondary structure at better than 70% accuracy. *J. Mol. Biol.* 232, 584–599.

Saitou, N., and Imanishi, T. 1989. Relative efficiencies of the Fitch-Margoliash, maximum parsimony, maximum likelihood, minimum evolution, and neighbor-joining methods of phylogenetic tree construction in obtaining the correct tree. *Mol. Biol. Evol.* 6, 514–525.

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

Schöniger, M., and von Haeseler, A. 1995. Performance of maximum likelihood, neighbor-joining, and maximum parsimony methods when sequence sites are not independent. *Syst. Biol.* 44(4), 533–547.

Sourdis, J., and Nei, M. 1996. Relative efficiencies of the maximum parsimony and distance-matrix methods in obtaining the correct phylogenetic tree. *Mol. Biol. Evol.* 5(3), 293–311.

Steel, M. 1992. The complexity of reconstructing trees from qualitative characters and subtrees. *J. Class.* 9, 91–116.

Steel, M.A., Székely, L.A., and Hendy, M.D. 1994. Reconstructing trees when sequence sites evolve at variable rate. *J. Comput. Biol.* 1(2), 153–163.

Strimmer, K., and von Haeseler, A. 1996. Accuracy of neighbor-joining for n-taxon trees. *Syst. Biol.* 45(4), 516–523.

Swofford, D.L., Olsen, G.J., Waddell, P.J., and Hillis, D.M. 1996. Phylogenetic inference, 407–514. *In* Hillis, D.M., Moritz, C., and Mable, B.K., eds., *Molecular Systematics*, 2nd ed. Sinauer, Sunderland, MA.

Warnow, T. 1996. Some combinatorial problems in phylogenetics. *In* Gyárfás, A., Lovász, L., and Székely, L.A., eds., *Proceedings of Graph Theory and Combinatorial Biology, Balatonlelle, Hungary, July 15–20. Bolyai Soc. Math. Studies*. 7, 363–413.

Waterman, M., Smith, T., and Beyer, W. 1977. Additive evolutionary trees. *J. Theoret. Biol.* 64, 199–213.

Address reprint requests to:
Tandy Warnow
Department of Computer Science
Taylor Hall, Room 2-124
University of Texas at Austin
Austin, TX 78712

E-mail: tandy@cs.utexas.edu