A few logs suffice to build (almost) all trees (I)

Péter L. Erdős
Mathematical Institute of the Hungarian Academy of Sciences
Budapest P.O.Box 127, Hungary-1364
E-mail: elp@math-inst.hu

Michael A. Steel
Biomathematics Research Centre
University of Canterbury
Christchurch, New Zealand
E-mail: m.steel@math.canterbury.ac.nz

László A. Székely
Department of Mathematics
University of South Carolina
Columbia, South Carolina
E-mail: laszlo@math.sc.edu

Tandy J. Warnow
Department of Computer and Information Science
University of Pennsylvania, Philadelphia PA
E-mail: tandy@central.cis.upenn.edu

Abstract

A phylogenetic tree (also called an "evolutionary tree") is a leaf-labelled tree which represents the evolutionary history for a set of species, and the construction of such trees is a fundamental problem in biology. Here we address the issue of how many sequence sites are required in order to recover the tree with high probability when the sites evolve under standard Markov-style *i.i.d.* mutation models. We provide analytic upper and lower bounds for the required sequence length, by developing a new (and polynomial time) algorithm. In particular we show that when the mutation probabilities are bounded the required sequence length can grow surprisingly slowly (a power of $\log n$) in the number n of sequences, for almost all trees.

1 Introduction

Rooted leaf-labelled trees are a convenient way to represent historical relationships between extant objects, particularly in evolutionary biology (where such trees are called *phylogenies*). Molecular techniques have recently provided large amounts of sequence data which are being used to reconstruct such trees. These methods exploit the variation in the sequences due to random mutations that have occurred at the sites, and statistically-based approaches typically assume that sites mutate independently and identically according to a Markov model. Under mild assumptions, for sequences generated by such a model, one can recover, with high probability, the underlying *unrooted* tree provided the sequences are sufficiently long (in terms of the number k of sites). How large this value of k needs to be depends on the reconstruction method, the details of the model, and the number n of species. Determining bounds on k and its growth with n has become more pressing since biologists have begun to reconstruct trees on increasingly larger numbers of species (often up to several hundred) from such sequences.

With this motivation, we provide upper and lower bounds for the value of k required to reconstruct an underlying (unrooted) tree with high probability, and address, in particular, the question of how fast k must grow with n. We first show that under any model, and any reconstruction method, k must grow *at least* as fast as $\log n$, and that for a particular, simple reconstruction method, it must grow at least as fast as $n \log n$, for any i.i.d. model. We then construct a new tree reconstruction method (the Dyadic Closure Method) which, for a simple Markov model, provides an upper bound on k which depends only on n, the range of the mutation probabilities across the edges of the tree, and a quantity called the "depth" of the tree. We show that the depth grows very slowly ($O(\log\log n)$) for almost all phylogenetic trees (under two distributions on trees). As a consequence, we show that the value of k required for accurate tree reconstruction by the Dyadic Closure Method need only grow as a power of $\log n$ for almost all trees when the mutation probabilities lie in a fixed interval, thereby improving results by Farach and Kannan in [23].

The structure of the paper is as follows. In Section 2 we provide definitions, and in Section 3 we provide lower bounds for k. In Section 4 we describe a technique for reconstructing a tree from a partial collection of subtrees, each on four leaves. We use this technique in Section 5, as the basis for our "dyadic closure" method. Section 6 is the central part of the paper - here we analyse, using various probabilistic arguments, an upper bound on the value of k required for this method to correctly recover the underlying tree with high probability, when the sites evolve under a simple, symmetric 2-state model. As this upper bound depends critically upon the depth (a feature of the shape of the tree) we show that the depth grows very slowly ($O(\log\log n)$) for a random tree selected under either of two distributions. This gives us the result that k need grow only sublinearly in n for nearly all trees.

2 Definitions

Notation: $\mathbb{P}[A]$ denotes the probability of event A; $\mathbb{E}[X]$ denotes the expectation of random variable X. We denote the natural logarithm by log. The set $[n]$ denotes $\{1, 2, \ldots, n\}$ and for any set S, $\binom{S}{k}$ denotes the collection of subsets of S of size k. $\mathbb{R}$ denotes the real numbers.

Definitions: (I) Trees. We will represent a phylogenetic tree T by a tree whose *leaves* (vertices of degree one) are labelled (by extant species, numbered by $1, 2, \ldots, n$) and whose remaining internal vertices (representing ancestral species) are unlabelled. We will adopt the biological convention that phylogenetic trees are *binary*, so that all internal nodes have degree three, and we will also assume that T is *unrooted*, for reasons described later in this section. There are $(2n-5)!! = (2n-5)(2n-7)\ldots 3 \cdot 1$ different binary trees on n distinctly labelled leaves.

The edge set of the tree is denoted by $E(T)$. Any edge adjacent to a leaf is called a *leaf edge*, any other edge is called an *internal edge*. The path between vertices u and v in the tree is called the uv path, and is denoted $P(u, v)$. For a phylogenetic tree T and $S \subseteq [n]$, there is a unique minimal subtree of T, containing all elements of S. We call this tree the *subtree* of T induced by S, and denote it by $T_{|S}$. We obtain the *contracted subtree* induced by S, denoted by $T^*_{|S}$, if we substitute edges for all maximal paths of $T_{|S}$ in which every internal vertex has degree two. Since all trees are assumed to be binary, all contracted subtrees, including, in particular, the subtrees on four leaves, are also binary. We use the notation $ij|kl$ for the contracted subtree on four leaves i, j, k, l in which the pair i, j is separated from the pair k, l by an internal edge, and we also call $ij|kl$ a *valid quartet split* of T. Clearly any four leaves i, j, k, l in a binary tree have exactly one valid quartet split out of $ij|kl$, $ik|jl$, $il|kj$.

The *topological distance* $d(u,v)$ between vertices u and v in a tree T is the number of edges in $P(u,v)$. A *cherry* in a binary tree is a pair of leaves at topological distance two. The *diameter* of the tree T, $diam(T)$, is the maximum topological distance in the tree. For an edge e of T, let T_1 and T_2 be the two rooted subtrees of T obtained by deleting edge e from T, and for $i = 1, 2$, let $d_i(e)$ be the topological distance from the root of T_i to its nearest leaf in T_i. The *depth* of T is $\max_e \max\{d_1(e), d_2(e)\}$, where e ranges over all internal edges in T. We say that a path P in the tree T is *short* if its topological length is at most $depth(T)+1$, and say that a quartet i,j,k,l is a *short quartet* if it induces a subtree which contains a single edge connected to four disjoint short paths. The set of all short quartets of the tree T is denoted by $Q_{short}(T)$. We will denote the set of valid quartet splits for the short quartets by $Q^*_{short}(T)$.

(II) Sites. Let us be given a set C of character states (such as $C = \{A, C, G, T\}$ for DNA sequences; $C = \{\text{the 20 amino acids}\}$ for protein sequences; $C = \{R, Y\}$ or $\{0, 1\}$ for purine-pyrimidine sequences). A *sequence of length k* is an ordered k-tuple from C—that is, an element of C^k. A collection of n such sequences—one for each species labelled from $[n]$—is called a *collection of aligned sequences.*

Aligned sequences have a convenient alternative description as follows. Place the aligned sequences as rows of an $n \times k$ matrix, and call *site i* the i^{th} column of this matrix. A *pattern* is one of the $|C|^n$ possible columns.

(III) Site substitution models. Many models have been proposed to describe, stochastically, the evolution of sites. Usually these models assume that the sites evolve identically and independently under a distribution that depends on the model tree. Most models are more specific and also assume that each site evolves on a rooted tree from a nondegenerate distribution π (of the r possible states) at the root, according to a Markov assumption (namely, that the state at each vertex is dependent only on its immediate parent). Each edge e oriented out from the root has an associated $r \times r$ stochastic transition matrix $M(e)$. Although these models are usually defined on a rooted binary tree T where the orientation is provided by a time scale and the root has degree 2, these models can equally well be described on an unrooted binary tree (without a degree 2 vertex) by (i) supressing the degree 2 vertex in T, (ii) selecting an arbitrary vertex (leaves not excluded), assigning to it an appropriate distribution of states π' (possibly different from π) and (iii) assigning an appropriate transition matrix $M'(e)$ (possibly different from $M(e)$) for each edge e. If we regard the tree as now rooted at the selected vertex, and the "appropriate" choices in (ii) and (iii) are made, then the resulting models gives *exactly* the same distribution on patterns as the original model (see [47]) and as the re-rooting is arbitrary we see why it is impossible to hope for the reconstruction of more than the *unrooted* underlying tree that generated the sequences (under some time-induced, edge-bisection rooting). Furthermore, while it is possible to obtain evidence supporting the existence of a particular edge in the model tree T, it is impossible to distinguish between a very short edge (that is, an edge e such that the probability of change on that edge is exceedingly small) and the tree that results from contracting that edge (or, equivalently, asserts that the probability of change on that edge is 0). Using sufficiently long sequences we expect to discover the short edges as well. For this reason, the general stochastic model assumes that the underlying trees are *binary.* (This assumption is also in keeping with the assumptions in systematic biology, that speciation events are almost always binary.)

(IV) The Neyman model. The simplest stochastic model is a symmetric model for binary characters due to Neyman [38], and also developed independently by Cavender [12] and Farris [25]. Let $\{0, 1\}$ denote the two states. The root is a fixed leaf, the distribution π at the root is uniform. For each edge e of T we have an associated *mutation probability*, which lies strictly between 0 and 0.5. Let $p : E(T) \rightarrow (0, 0.5)$ denote the associated map. We have an instance of the general Markov model with $M(e)_{01} = M(e)_{10} = p(e)$. We will call this the *Neyman 2-state model*, but note that it has also been called the Cavender-Farris model.

The Neyman 2-state model is hereditary on subsets of the leaves—that is, if we select a subset S of $[n]$, and form the subtree $T_{|S}$, then eliminate vertices of degree two, we can define mutation probabilities on the edges of $T^*_{|S}$ so that the probability distribution on the patterns on S is the same as the marginal of the distribution on patterns provided by the original tree T. Furthermore, the mutation probabilities that we assign to an edge of $T^*_{|S}$ is just the probability p that the endpoints of the associated path in the original tree T are in different states. The probability that the endpoints of a path p are in different states is nicely related to the mutation probabilities $p_1, p_2, ..., p_k$ of edges of the k-path:

$$p = \frac{1}{2}\left(1 - \prod_{i=1}^{k}(1-2p_i)\right) . \tag{1}$$

Formula (1) is well-known, and is easy to prove by induction.

(V) Distances. Any symmetric matrix, which is zero-diagonal and positive off-diagonal, will be called a *distance matrix.* An $n \times n$ distance matrix D_{ij} is called *additive*, if there exists an n-leaf tree (not necessarily binary) with positive edge weights on the internal edges and non-negative edge weights on the leaf edges, so that D_{ij} equals the sum of edge weights in the tree along the $P(i,j)$ path connecting i and j. In [10], Buneman showed that the following Four-Point Condition characterizes additive matrices (see also [43] and [54]):

Theorem 1 (Four Point Condition)
A matrix D is additive if and only if for all i, j, k, l (not necessarily distinct), the maximum of $D_{ij}+D_{kl}, D_{ik}+D_{jl}, D_{il}+D_{jk}$ is not unique. The edge weighted tree (with positive weights on internal edges and non-negative weights on leaf edges) representing the additive distance matrix is unique among the trees without vertices of degree two. □

Given a pair of parameters (T, p) for the Neyman 2-state model, and sequences of length k generated by the model, let $H(i,j)$ denote the *Hamming distance* of sequences i and j and

$$h^{ij} = H(i,j)/k \tag{2}$$

denote the *dissimilarity score* of sequences i and j. The *empirical corrected distance* between i and j is denoted by

$$d_{ij} = -\frac{1}{2}\log(1-2h^{ij}). \tag{3}$$

The probability of a change in the state of any fixed character between the sequences i and j is denoted by $E^{ij} = \mathbb{E}(h^{ij})$, and we let

$$D_{ij} = -\frac{1}{2}\log(1-2E^{ij}) \tag{4}$$

denote the *corrected model distance* between i and j. We assign to any edge e a positive weight

$$w(e) = -\frac{1}{2}\log(1-2p(e)). \tag{5}$$

By Equation (1), D_{ij} is the sum of the weights (see previous equation) along the path $P(i,j)$ between i and j. Therefore, d_{ij} converges in probability to D_{ij} as $k \to \infty$. Corrected distances were introduced to handle the problem that Hamming distances underestimate the "true evolutionary distances". In certain continuous time Markov models the edge weight means the expected number of back-and-forth state changes along the edge, and defines an additive distance matrix.

(VI) Tree reconstruction. A *phylogenetic tree reconstruction method* is a function Φ that associates either a tree or the statement `fail` to every collection of aligned sequences, the latter indicating that the method is unable to make such a selection for the data given. Some methods are based upon sequences, while others are based upon distances.

According to the practice in systematic biology (see, for example, [29, 30, 50]), a method is considered to be *accurate* if it recovers the unrooted binary tree T, even if it does not provide any estimate of the mutation probabilities. A necessary condition for accuracy, under the models discussed above, is that two distinct trees, T, T', do not produce the same distribution of patterns no matter how the trees are rooted, and no matter what their underlying Markov parameters are. This "identifiability" condition is violated under an extension of the *i.i.d.* Markov model when there is an unknown distribution of rates across sites as described by Steel *et al.* [47]. However, it is shown in Steel [45] (see also Chang [13]) that the identifiability condition holds for the *i.i.d* model under the weak conditions that the components of π are not zero and the determinant $\det(M(e)) \neq 0, 1, -1$, and in fact we can recover the underlying tree from the expected frequencies of patterns on just *pairs* of species.

Theorem 1 and the discussion that follows it suggest that appropriate methods applied to corrected distances will recover the correct tree topology from sufficiently long sequences. Consequently, one approach to reconstructing trees from distances is to seek an additive distance matrix of minimum distance (with respect to some metric on distance matrices) from the input distance matrix. Many metrics have been considered, but all resultant optimization problems have been shown or are assumed to be NP-hard (see [1, 15, 24] for results on such problems).

We will use a particular simple distance method, which we call the *(Extended) Four Point Method* (FPM), to reconstruct trees on four leaves from a matrix of interleaf distances.

> **Four-Point Method (FPM)** Given a 4×4 distance matrix d, return the set of splits $ij|kl$ which satisfy $d_{ij} + d_{kl} \leq \min\{d_{ik} + d_{jl}, d_{il} + d_{jk}\}$.

Note that the Four-Point Method can return one, two, or three splits for a given quartet. One split is returned if the minimum is unique, two are returned if the two smallest values are identical but smaller than the largest, and three are returned if all three values are equal.

In [26], Felsenstein showed that two popular methods—*maximum parsimony* and *maximum compatibility* — can be statistically inconsistent, namely, for some parameters of the model, the probability of recovering the correct tree topology tends to 0 as the sequence length grows. This region of the parameter space has been subsequently nicknamed the "Felsenstein zone". This result, and other more recent embellishments (see Hendy [28], Zharkikh and Li [55], Takezaki and Nei [51], Steel *et al.* [47]) are asymptotic results — that is they are concerned with outcomes as the sequence length, k, tends to infinity.

We consider the question of how many sites k must be generated independently and identically, according to a substitution model M, in order to reconstruct the underlying binary tree on n species with pre-specified probability at least ϵ by a particular method Φ. Clearly, the answer will depend on Φ, ϵ, and n, and also on the fine details of M — in particular the unknown values of its parameters. It is clear that for all models that have been proposed, if no restrictions are placed on the parameters associated with edges of the tree then the sequence length might need to be astronomically large, even for four sequences, since the "edge length" of the internal edge(s) of the tree can be made arbitrarily short (as was pointed out by Philippe and Douzery [39]). A similar problem arises for four sequences when one or more of the four non-internal edges is "long"—that is, when site saturation has occurred on the line of descent represented by the edge(s). Unfortunately, it is difficult to analyse how well methods perform for sequences of a given length, k. There has been some empirical work done on this subject, in which simulations of sequences are made on different trees and different methods compared according to the sequence length needed (see [31] for an example of a particularly interesting study of sequence length needed to infer trees of size four!), but little analytical work (see, however, [39]).

In this paper we consider only the Neyman 2-state model as our choice for M. However, our results extend to the general *i.i.d.* Markov model, and the interested reader is referred to the companion paper, [21], for details.

3 Lower bounds

Since the number of binary trees on n leaves is $(2n-5)!!$, encoding deterministically all such trees by binary sequences at the leaves requires that the sequence length, k, satisfy $(2n-5)!! \leq 2^{nk}$, i.e. $k = \Omega(\log n)$. We now show that this information-theoretic argument can be extended for *arbitrary* models of site evolution and *arbitrary* deterministic or even randomized algorithms for tree reconstruction. For each tree, T, and for each algorithm A, whether deterministic or randomized, we will assume that T is equipped with a mechanism for generating sequences, which allows the algorithm A to reconstruct the topology of the underlying tree T from the sequences with probability bounded from below.

Theorem 2 *Let A be an arbitrary algorithm, deterministic or randomized, which is used to reconstruct binary trees from 0-1 sequences of length k associated with the leaves, under an arbitrary model of substitutions. If A reconstructs the topology of any binary tree T from the sequences at the leaves with probability greater than ϵ (respectively greater than $1/2$), then $(2n-5)!!\epsilon < 2^{nk}$ (respectively, $(2n-5)!! \leq 2^{nk}$, under the assumption of independence of the substitution model and the reconstruction), and so $k = \Omega(\log n)$.*

We prove this theorem in a more abstract setting:

Theorem 3 *We have finite sets X and S and random functions $f : S \to X$ and $g : X \to S$.*
(i) If $\mathbb{P}[fg(x) = x] > \epsilon$ for all $x \in X$ then $|S| > \epsilon|X|$.
(ii) If f, g are independent and $\mathbb{P}[fg(x) = x] > 1/2$ for all $x \in X$ then $|S| \geq |X|$.

Proof. Proof of (i): By hypothesis $\epsilon|X| < \sum_x \mathbb{P}[fg(x) = x] = \sum_x \sum_s \mathbb{P}[g(x) = s \ \& \ f(s) = x] \leq \sum_s (\sum_x \mathbb{P}[f(s) = x]) = \sum_s 1 = |S|$.

Proof of (ii): First note that $\mathbb{P}[fg(x) = y] = \sum_s \mathbb{P}[f(s) = y]\mathbb{P}[g(x) = s]$ by independence. Arrange the numbers $\mathbb{P}[f(s) = x]$ into an $|X| \times |S|$ matrix B and the numbers $\mathbb{P}[g(x) = s]$ into an $|S| \times |X|$ matrix A. Now we have ${}_y[BA]_x = \mathbb{P}[fg(x) = y]$ and the column sums in BA equal to 1.

Let $C = (c_{ij})$ denote a complex square matrix. A theorem of Lévy and Desplanques asserts that if for all i, $|c_{ii}| > \sum_{i:\ i \neq j} |c_{ij}|$, then $\det(C) \neq 0$ ([37], p. 146). By our assumptions $C = BA$ satisfies the conditions of the Lévy-Desplanques theorem, and hence $|X| = rank(BA) \leq rank(B) \leq |S|$.

There is another proof which avoids linear algebra. Observe that for each x, there exists an $s = s_x$ for which $\mathbb{P}[f(s_x) = x] > 1/2$, since otherwise we have $\mathbb{P}[fg(x) = x] \leq 1/2$. Now, the map sending x to s_x is one-to-one from X into S (and so $|X| \leq |S|$ as required) since otherwise, if two elements get mapped to s, then $1 = \sum_x \mathbb{P}[f(s) = x] > 1/2 + 1/2$. □

The following example shows that our theorem is tight for $\epsilon < 1/2$: Let $X = \{x_{11}, x_{12}, x_{21}, x_{22}, ..., x_{n1}, x_{n2}\}$ and $S = \{1, 2, ..., n\}$, and let $g(x_{ij}) = i$ (with probability 1); and let $f(i) = x_{i1}$

with probability 1/2; x_{i2} with probability 1/2. Then, $\mathbb{P}[fg(x) = x] = 1/2$, so $\mathbb{P}[fg(x) = x] > \epsilon$, for *any* epsilon less than 1/2. But notice that $(1/2)|X| = |S|$.

Curiously, once ϵ exceeds 1/2 we must have $|X| \leq |S|$, under the assumption of independence. Examples [53] show that the assumption of independence is necessary. Independence is a reasonable assumption if we try to apply this result for evolutionary tree reconstruction and holds automatically if the tree reconstruction method is deterministic.

This lower bound applies to an *arbitrary* algorithm, but *particular* algorithms may admit much larger lower bounds. Consider, for example, the *Maximum Compatibility method* (MC), which we now define. Given a set of binary sequences, each site defines a partition of the sequences into two sets, those containing a 0 in that position, and those containing a 1 in that position. The site is said to be *compatible* on a tree T if the tree T contains an edge whose removal would define the same partition. The objective of the maximum compatibility method is a tree T which has the largest number of sites compatible with it. Maximum compatibility is an NP-hard optimization problem [16], although the MC method can clearly be implemented as a non-polynomial time algorithm. We now show that the sequence length needed by MC to obtain the correct topology with constant probability must grow *as least* as fast as $n \log n$.

Theorem 4 *Assume that 2-state sites on n species evolve on a binary tree T according to any stochastic model in which the sites evolve identically and independently. Let $k(n)$ denote the smallest number of sites for which the Maximum Compatibility Method is guaranteed to reconstruct the topology of T with probability greater than 1/2. Then, for n large enough,*

$$k(n) > (n-3)\log(n-3) - (n-3). \tag{6}$$

Proof. We will say that a site is *trivial* if it defines a partition of the sequences into one class or into two classes so that one of the classes is a singleton. Now, fix x and assume that we are given $k(n)$ sites, out of which the number of non-trivial ones is equal to $k^* = \lceil (n-3)\log(n-3) + x(n-3) \rceil$. We will show that the probability of obtaining the correct tree under MC is at most $e^{-e^{-x}}$, which proves the theorem by setting $x = -1$, since $k(n) \geq k^*$.

Let $\sigma(T)$ denote the set of internal splits of T. Since T is binary, $|\sigma(T)| = n - 3$ [10]. For $\sigma \in \sigma(T)$, let the random variable X_σ be the number of non-trivial sites which induce split σ. Define $X = \sum_{\sigma \in \sigma(T)} X_\sigma$. A necessary (though not sufficient) condition for maximum compatibility to select T is that all the internal splits of T are present amongst the k^* non-trivial sites. Thus, we have the inequality:

$$\begin{aligned}
\mathbb{P}[MC(\mathbf{S}) = T] &\leq \mathbb{P}[\cap_{\sigma \in \sigma(T)} \{X_\sigma > 0\}] \\
&= \sum_{i=1}^{k^*} \mathbb{P}[\cap_{\sigma \in \sigma(T)} \{X_\sigma > 0\} \mid X = i] \times \mathbb{P}[X = i] \\
&\leq \max_{1 \leq i \leq k^*} \mathbb{P}[\cap_{\sigma \in \sigma(T)} \{X_\sigma > 0\} \mid X = i] \\
&= \mathbb{P}[\cap_{\sigma \in \sigma(T)} \{X_\sigma > 0\} \mid X = k^*] \,.
\end{aligned} \tag{7}$$

Let $p(\sigma)$ denote the probability of generating split σ at a particular site. Due to the model, $p(\sigma)$ does not depend on the site. It is not difficult to show that (7) is maximized when the $p(\sigma)$'s are all equal ($\sigma \in \sigma(T)$) and sum to 1.

Indeed, by compactness arguments, there exists a probability distribution maximizing (7). We show that it cannot be non-uniform, and therefore the uniform distribution maximizes (7). Assume that the maximizing distribution p is non-uniform, say, $p(\sigma) \neq p(\rho)$. We introduce a new distribution p' with $p'(\sigma) = p'(\rho) = \frac{1}{2}(p(\sigma) + p(\rho))$, and $p'(\alpha) = p(\alpha)$ for $\alpha \neq \sigma, \rho$. The probability of having exactly i sites supporting σ *or* ρ is the same for p and p'. Conditioning on the number of sites supporting σ or ρ, it is easy to see that any distribution of sites supporting all non-trivial splits has strictly higher probability in p' than in p.

Knowing that the $p(\sigma)$'s are all equal ($\sigma \in \sigma(T)$) and sum to 1, determining (7) is just the classical occupancy problem where k^* balls are randomly assigned to $n-3$ boxes with uniform distribution, and one asks for the probability that each box has at least one ball in it. Equation (6) now follows from a result on the asymptotics of this problem (Erdős and Rényi [18]): for $x \in \mathbb{R}$, k^* balls (k^* as defined above), and $n-3$ boxes, the limit of probability of filling each boxes is $e^{-e^{-x}}$. □

In Section 5, we will present a new method (the *Dyadic Closure Method* (DCM)) for reconstructing trees. DCM has the property that for almost all trees, with a wide range allowed for the mutation probabilities, the sequence length that *suffices* for correct topology reconstruction grows no more than polynomially in the lower bound of $\log n$ (see Theorem 2) required for any method. In fact the same holds for *all trees* with a *narrow range* allowed for the mutation probabilities. First, however, we set up a combinatorial technique for reconstructing trees from selected subtrees of size four.

4 Dyadic Inference of Trees

Certain classical tree reconstruction methods are based upon reconstructing trees on quartets of leaves, then combining these trees into one tree on the entire set of leaves. Here we describe a method which reconstructs only certain quartet splits (the "representative quartet splits"), and then infers the remaining quartet splits using "inference rules". Once we have splits for all the possible quartets of leaves, we can then reconstruct the tree (if one exists) that is uniquely consistent with all the quartet splits.

In this section, we will prove a stronger result than was provided in [19], that the *representative quartet splits* suffice to define the tree. We will also present a tree reconstruction algorithm, *DCTC* (for *Dyadic Closure Tree Construction*) based upon dyadic closure. The input to *DCTC* is a set Q of quartet splits and we show that *DCTC* is guaranteed to reconstruct the tree properly if the set Q contains only valid quartet splits and contains all the representative quartet splits of T. We also show that if Q contains all representative quartet splits but also contains invalid quartet splits, then *DCTC* will discover incompatibility. In the remaining case, where Q does not contain all the representative quartet splits of any T, *DCTC* will return *Inconsistent* (and then the input was inconsistent indeed), or a tree (which is then the only tree consistent with the input), or *Insufficient*.

4.1 Inference rules

Recall that, for a binary tree T on n leaves, and a quartet of leaves, $q = \{a, b, c, d\} \in \binom{[n]}{4}$, $t_q = ab|cd$ is a *valid quartet split* of T if $T^*_{|q} = ab|cd$ (i.e. there is at least one edge in T whose removal separates the pair a, b from the pair c, d). It is easy to see that

$$\text{if } ab|cd \text{ is a valid quartet split of } T\text{, then so are } ba|cd \text{ and } cd|ab, \tag{8}$$

and we identify these three splits; and if (8) holds, then $ac|bd$ and $ad|bc$ are not valid quartet splits of T, and we say that any of them *contradicts* (8). Let $Q(T) = \left\{t_q : q \in \binom{[n]}{4}\right\}$ denote the set of valid quartet splits of T. It is a classical result that $Q(T)$ determines T (Colonius and Schulze [14], Bandelt and Dress [6]); indeed for each $i \in [n]$, $\{t_q : i \in q\}$ determines T, and T can be computed in polynomial time.

It would be nice to determine for a set of quartet splits whether there is a tree for which they are valid quartet splits. Unfortunately, this problem is NP-complete (Steel [44]). It also would be useful to know which subsets of $Q(T)$ determine T, and for which such subsets a polynomial time procedure would exist to reconstruct T. A natural step in this direction is to define *inference*: we can infer from a set of quartet splits A a quartet split t, if whenever $A \subseteq Q(T)$ for a binary tree T, then $t \in Q(T)$ as well.

Instead, Dekker [17] introduced a restricted concept, *dyadic* and higher order inference. Following Dekker, we say that a set of quartet splits A *dyadically implies* a quartet split t, if t can be derived from A by repeated applications of rules (8), (9) and (10):

$$\begin{aligned}&\text{if } ab|cd \text{ and } ac|de \text{ are valid quartet splits of } T,\\ &\qquad\text{then so are } ab|ce,\ ab|de, \text{ and } bc|de,\end{aligned} \tag{9}$$

and,

$$\text{if } ab|cd \text{ and } ab|ce \text{ are valid quartet splits of } T, \text{ then so is } ab|de. \tag{10}$$

It is easy to check that these rules infer valid quartet splits from valid quartet splits, and the set of quartet splits dyadically inferred from an input set of quartet splits can be computed in polynomial time. Setting a complete list of inference rules seems hopeless (Bryant and Steel [9]): for any r, there are r-ary inference rules, which infer a valid quartet split from some r valid quartet splits, such that their action cannot be expressed through lower order inference rules.

4.2 Tree inference using dyadic rules

In this section we define the *dyadic closure* of a set of quartet splits, and describe conditions on the set of quartet splits under which the dyadic closure defines all valid quartet splits of a binary tree. This section extends and strengthens results from earlier work [19, 46].

Definition 1: Given a finite set of quartet splits Q, we define the *dyadic closure* $cl(Q)$ of Q as the set of quartet splits that can be inferred from Q by the repeated use of the rules (8, 9, 10). We say that Q is *inconsistent*, if Q is not contained by the set of valid quartet splits of any tree, otherwise Q is *consistent*. For each of the $n-3$ internal edges of the n-leaf binary tree T we assign a *representative quartet* $\{s_1, s_2, s_3, s_4\}$ as follows. The deletion of the internal edge *and* its endpoints defines four rooted subtrees t_1, t_2, t_3, t_4. Within each subtree t_i, select from among the leaves which are closest topologically to the root the one, s_i, which is the smallest natural number (recall that the leaves of our trees are natural numbers). This procedure associates to each edge a set of four leaves, i, j, k, l. (By construction, it is clear that the quartet i, j, k, l induces a short quartet in T - see Section 2 for the definition of "short quartet.") We call the quartet split of a representative quartet a *representative quartet split* of T, and we denote the set of representative quartet splits of T by R_T.

The aim of the section is to show that the dyadic closure suffices to compute the tree T from any set of valid quartet splits of T which contain R_T. We begin with:

Lemma 1 *Suppose S is a set of $n-3$ quartet splits which is consistent with a unique binary tree T on n leaves. Furthermore, suppose that S can be ordered $q_1, ..., q_{n-3}$ in such a way that q_i contains at least one label which does not appear in $\{q_1, ..., q_{i-1}\}$ for $i = 2, ..., n-3$. Then, the dyadic closure of S is $Q(T)$.*

Proof. First, observe that the lemma is sufficient to show for the case when q_i contains *exactly* one label which does not appear in $\{q_1, ..., q_{i-1}\}$ for $i = 2, ..., n-3$, since $n-4$ quartets have to add $n-4$ new vertices. Let $S_i = \{q_1, ..., q_i\}$, and let L_i be the union of the leaves of the quartet splits in S_i, and let $T_i = T^*_{|L_i}$ be the binary subtree of T induced by L_i. We first make

Claim 1: The only tree on L_i consistent with S_i is T_i, for $i = 1, ..., n-3$.

Proof of Claim 1: The claim is true by the hypothesis of Lemma 1 for $i = n-3$; suppose for some $i < n-3$ it is false. Then there exist (at least) two trees that realize S_i, one of which is T_i, the other we will call $T^\#$. Now each quartet $q_{i+1}, ..., q_{n-3}$ adds a new leaf to the tree so far constructed from T_i and $T^\#$. Now for each quartet *we can always* attach that new leaf in at least one position in the tree so far constructed so as to satisfy the corresponding quartet split (and all earlier ones, since they don't involve that leaf). Thus we end up with two trees consistent with S, and these are different trees since when we restrict them to L_i, they differ. But this contradicts our hypothesis. □

Next we make

Claim 2: If x is the new leaf introduced by $q_{n-3} = xa|bc$ then x and a form a cherry of T.

Proof of Claim 2: First assume that x belongs to the cherry xy but $a \neq y$. Since this quartet is the only occurence of x we do not have any information about this cherry, therefore the reconstruction of the tree T cannot be correct, a contradiction.

Now assume that x is not in a cherry at all. Then the neighbor of x has two other neighbors, and those are not leaves. In turn they have two other neighbors each. Hence, we can describe x's place in T in the following representation: take a binary tree with five leaves, label the middle leaf x, and replace the other four leaves by corresponding subtrees of T.

Now suppose $q_{n-3} = ax|bc$. Regardless of where a, b, c come from (among the four subtrees in the representation), we can always move x onto at least two of the other four edges in T, and so obtain a different tree consistent with S (recall that q_{n-3} is the only quartet containing x, and thereby the only obstruction to us moving x!). Since the theorem assumes that the quartets are consistent with a unique tree, this contradicts our assumptions. □

Finally, it is easy to show the following:

Claim 3: Suppose xy is a cherry of T. Select leaves a, b from each of the two subtrees adjacent to the cherry. Let T' be the binary tree obtained by deleting leaf x. Then $cl(Q(T') \cup \{xy|ab\}) = Q(T)$.

Now, we can apply induction on n to establish the lemma. It is clearly (vacuously) true for $n = 4$, so suppose $n > 4$. Let x be the new leaf introduced by q_{n-3}, and let the binary tree T' be T with x deleted.

In view of Claim 1, S_{n-4} is a set of $n-4$ quartets that define $T_{n-4} = T'$, a tree on $n-1$ leaves and which satisfy the hypothesis that q_i introduces exactly one new leaf. Thus, applying the induction hypothesis, the dyadic closure of S_{n-4} is $Q(T')$. Since $S = S_{n-3}$ contains S_{n-4}, the dyadic closure of S also contains $Q(T')$, which is the set of all quartet splits of T that do not include x.

Now, by Claim 2, x is in a cherry; let its sibling in the cherry be y, so $q_{n-3} = ab|xy$, say, where a and b must lie in each of the two subtrees adjacent to the cherry. (It is easy to see that if a, b both lay in just one of these subtrees, then S would not define T.)

Now, as we just said, the dyadic closure of S contains $Q(T')$ and it also contains $ab|xy$ (where a, b are as specified in the preceding paragraph) and so by the idempotent nature of dyadic closure (i.e.

$cl(B) = cl(cl(B)))$ it follows from Claim 3 that the dyadic closure of S equals $Q(T)$. □□□

Lemma 2 *The set of representative quartet splits R_T of a binary tree T satisfies the conditions of Lemma 1. Hence, the dyadic closure of R_T is $Q(T)$.*

Proof. In order to make an induction proof possible, we make a more general statement. Given a binary tree T with a positive edge weighting w, we define the *representative quartet* of an edge e to be the quartet tree defined by taking the lowest indiced closest leaf in each of the four subtrees, where we define "closest" in terms of the weight of the path (rather than the topological distance) to the root of the subtree. We also define the *representative quartet splits of the weighted tree*, $R_{T,w}$ as in the Definition with the only change that the $s_i \in t_i$ is selected by the sum-of-weights criterion instead of the topological distance criterion. Observe that if all weights are equal to one, then we get back the original definitions from the Definition. When turning to binary subtrees of a given weighted tree, we assign the sum of weights of the original edges to any newly created edge which is composed of them, and denote the new weighting by w^*. Now we easily can prove by induction the following generalization of the statement of Lemma 2:

Claim 4: Take the set of representative quartet splits $R_{T,w}$ of a weighted n-leaf binary tree T. Then for every other n-leaf binary tree F, we have that $R_{T,w} \subseteq Q(F)$ implies $T = F$ as unweighted trees. Furthermore, $R_{T,w}$ can be ordered $q_1, ..., q_{n-3}$ in such a way that q_i contains exactly one label that does not appear in $\{q_1, ..., q_{i-1}\}$ for $i = 2, ..., n-3$.

Proof of Claim 4: First we show that the only tree consistent with the set of representative splits $R_{T,w}$ of a binary tree T is T itself. Look for the smallest (in n) counterexample T, such that $R_{T,w} \subseteq Q(F)$ for a tree $F \neq T$. Clearly n has to be at least 5. Therefore T has at least two different cherries, say xy and uv, such that $d(u,x) \geq 4$. Let us denote by $w(l)$ the weight of the leaf edge corresponding to the leaf l. If $w(x) < w(y)$ or ($w(x) = w(y)$ and $x < y$), then due to the construction of $R_{T,w}$, vertex y occurs in exactly one element of $R_{T,w}$, say p, which is the representative of the edge that separate xy from the rest of the tree. A similar argument would show that one of u, v, say v, occurs in exactly one element of $R_{T,w}$, say q. It also follows that $p \neq q$. It is not difficult to check that

$$R_{T^*_{|[n]\setminus\{y\}},w^*} = R_T \setminus \{p\} \text{ and } R_{T^*_{|[n]\setminus\{v\}},w^*} = R_T \setminus \{q\} \tag{11}$$

according to the definition of weight after contracting edges, where $T^*_{|K}$ is the binary tree obtained by contracting paths into edges in the subtree of T spanned by the vertex set K. Hence, by the minimality of the counterexample, $T^*_{|[n]\setminus\{y\}} = F^*_{|[n]\setminus\{y\}}$ and $T^*_{|[n]\setminus\{v\}} = F^*_{|[n]\setminus\{v\}}$. We know that any edge of F defines a bipartition of $[n]$, and traces of these bipartitions on $[n] \setminus \{y\}$ and $[n] \setminus \{v\}$ are exactly the bipartitions produced by the edges of $F^*_{|[n]\setminus\{y\}}$ on $[n] \setminus \{y\}$ and the bipartitions produced by the edges of $F^*_{|[n]\setminus\{v\}}$ on $[n] \setminus \{v\}$. Therefore also in F both xy and uv make cherries, and hence $T = F$, a contradiction.

For the other part of the claim, it immediately follows by induction from formula (11) that $R_{T,w}$ can be ordered so that every quartet in the order contains *at least one* (and therefore *exactly* one) new leaf. (Eliminate quartet splits recursively using (11), and put $R_{T,w}$ in the reverse order.) □

Note that the generalization for weighted trees was necessary, since without weights formula (11) would fail. □□□

We note here that representative quartets cannot be defined by selecting *any* nearest leaf in the four subtrees associated with an internal edge. For example, consider the tree T on six leaves labelled 1 through 6, with a central vertex and cherries $(1,2)$, $(3,4)$, and $(5,6)$, hanging from the central vertex. If we selected the quartet splits by arbitrarily picking "closest" leaves in each of the four subtrees around each internal edge, we could possibly select splits $12|36$, $34|15$ and $56|24$; however, these splits do not uniquely identify the tree T, since the tree with cherries 15, 24 and 36 is also consistent with these quartets. Therefore, in order to be able to keep Lemma 2 and Theorem 6, we will require that the representative quartet splits be selected as we have defined them.

4.3 Algorithm $DCTC$

We now present the Dyadic Closure Tree Construction method (*DCTC*) for computing the dyadic closure of a set Q of quartet splits, and which returns the tree T when $cl(Q) = Q(T)$.

Before we present the algorithm, we note the following interesting lemma:

Lemma 3 *If $cl(Q)$ contains exactly one split for each possible quartet then $cl(Q) = Q(T)$ for a unique binary tree T.*

Proof. By Proposition (2) of [6], a set Q^* of noncontradictory quartet splits equals $Q(T)$ for some tree T precisely if it satisfies the substitution property: If $ab|cd \in Q^*$, then for all $e \notin \{a, b, c, d\}$, $ab|ce \in Q^*$, or $ae|cd \in Q^*$. Furthermore, in that case, T is unique.

Applying this characterization to $Q^* = cl(Q)$, suppose $ab|cd \in cl(Q)$ but $ab|ce \notin cl(Q)$. Thus, either $ae|bc \in cl(Q)$ or $ac|be \in cl(Q)$. In the either case, the dyadic inference rule applied to the pair $\{ab|cd, ae|bc\}$ or to $\{ab|cd, ac|be\}$ implies $ae|cd \in cl(Q)$, and so $cl(Q)$ satisfies the substitution property. Thus $cl(Q) = Q(T)$ for a unique tree T. Finally, since $cl(Q)$ contains a split for each possible quartet, it follows that T must be binary. □

We now continue with the description of the DCTC algorithm.

Algorithm DCTC:

Step 1: We compute the dyadic closure, $cl(Q)$, of Q.

Step 2:
- **Case 1:** $cl(Q)$ contains a pair of contradictory splits for some quartet: return *Inconsistent*.
- **Case 2:** $cl(Q)$ has no contradictory splits, but fails to have a split for every quartet: Return *Insufficient*
- **Case 3:** $cl(Q)$ has exactly one split for each quartet: apply standard algorithms [6, 52] to $cl(Q)$ to reconstruct the tree T such that $Q(T) = cl(Q)$. Return T.

(Case 3 depends upon Lemma 3 above.)

To completely describe the $DCTC$ method we need to specify how we compute the dyadic closure of a set Q of quartet splits.

Efficient computation of dyadic closure The description we now give of an efficient method for computing the dyadic closure will only actually completely compute the dyadic closure of Q if $cl(Q) = Q(T)$ for some tree T. Otherwise, $cl(Q)$ will either contain a contradictory pair of splits for some quartet, or $cl(Q)$ will not contain a split for every quartet. In the first of these two cases, the method will return *Inconsistent*, and in the second of these two cases, the method will return *Insufficient*. However, the method can be easily modified to compute $cl(Q)$ for all sets Q.

We will maintain a four-dimensional array `Splits` and constrain $\texttt{Splits}_{\texttt{i,j,k,l}}$ to either be empty, or to contain exactly one split that has been inferred so far for the quartet i, j, k, l. In the event that two conflicting splits are inferred for the same quartet, the algorithm will immediately return *Inconsistent*, and halt. We will also maintain a queue Q_{new} of new splits that must be processed. We initialize `Splits` to contain the splits in the input Q, and we initialize Q_{new} to be Q, ordered arbitrarily.

The dyadic inference rules in equations (8), (9), and (10) show that we infer new splits by combining two splits at a time, where the underlying quartets for the two splits share three leaves. Consequently, each split $ij|kl$ can only be combined with splits on quartets $\{a,i,j,k\}$, $\{a,i,j,l\}$, $\{a,i,k,l\}$, and $\{a,j,k,l\}$, where $a \notin \{i,j,k,l\}$. Consequently, there are only $4(n-4)$ other splits with which any split can be combined using these dyadic rules to generate new splits.

Pop a split $ij|kl$ off the queue Q_{new}, and examine each of the appropriate $4(n-4)$ entries in `Splits`. For each non-empty entry in *Splits* that is examined in this process, compute the $O(1)$ splits that arise from the combination of the two splits. Suppose the combination generates a split $ab|cd$. If $\texttt{Splits}_{a,b,c,d}$ contains a different split from $ab|cd$, then Return *Inconsistent*. If $\texttt{Splits}_{a,b,c,d}$ is empty, then set $\texttt{Splits}_{a,b,c,d} = ab|cd$, and add $ab|cd$ to the queue Q_{new}. Otherwise $\texttt{Splits}_{a,b,c,d}$ already contains the split $ab|cd$, and we do not modify the data structures.

Continue until the queue Q_{new} is empty, or Inconsistency has been observed. If the Q_{new} empties before Inconsistency is observed, then check if every entry of `Splits` is non-empty. If so, then $cl(Q) = Q(T)$ for some tree; Return `Splits`. If some entry in `Splits` is empty, then return *Insufficient*.

Theorem 5 *The efficient computation of the dyadic closure uses $O(n^5)$ time, and at the termination of the algorithm the* `Splits` *matrix is either identically equal to $cl(Q)$, or the algorithm has returned* Inconsistent*. Furthermore, if the algorithm returns* Inconsistent*, then $cl(Q)$ contains a pair of contradictory splits.*

Proof. It is clear that if the algorithm only computes splits using dyadic closure, so that at any point in the application of the algorithm, $\texttt{Splits} \subseteq cl(Q)$. Consequently, if the algorithm returns *Inconsistent*, then $cl(Q)$ does contain a pair of contradictory splits. If the algorithm does not return *Inconsistent*, then it is clear from the design that every split which could be inferred using these dyadic rules would be in the Splits matrix when the algorithm terminates. Briefly, we show why this is true. If $ab|cd$ is inferred via dyadic rules and $ab|cd \notin Q$, then there is some sequence of pairwise combinations of splits (from Q, or from newly generated splits) that would give rise to $ab|cd$. Let S_1 and S_2 be the two splits that are combined to give rise to $ab|cd$, and assume inductively that each is either in Q or is generated by the algorithm; consequently, both are placed in Q_{new}. Suppose, without loss of generality, that the position of S_1 in Q_{new} precedes the position of S_2 in Q_{new}. Then since the algorithm does not return *Inconsistent*, it will process S_2 at some point during its operation, and before it processes S_2, `Splits` will contain the split S_1. Since S_1 and S_2 *can* be combined, the processing of S_2 will include combining S_1 and S_2, and hence will generate $ab|cd$, which will then be placed in $\texttt{Splits}_{a,b,c,d}$.

The running time analysis is easy. Every combination of quartet splits takes $O(1)$ time to process. Processing a quartet split involves examining $4(n-4)$ entries in the `Splits` matrix, and hence costs $O(n)$. If a split $ij|kl$ is generated by the combination of two splits, then it is only added to the queue if $\texttt{Splits}_{i,j,k,l}$ is empty when $ij|kl$ is generated. Consequently, at most $O(n^4)$ splits ever enter the queue. □

We now prove our main theorem of this section:

Theorem 6 *Let Q be a set of quartet splits.*

1. *If $DCTC(Q) = T$, $DCTC(Q') = T'$, and $Q \subseteq Q'$, then $T = T'$.*
2. *If $DCTC(Q) =$ Inconsistent and $Q \subseteq Q'$, then $DCTC(Q') =$ Inconsistent.*
3. *If $DCTC(Q) =$ Insufficient and $Q' \subseteq Q$, then $DCTC(Q') =$ Insufficient.*
4. *If $R_T \subseteq Q \subseteq Q(T)$, then $DCTC(Q) = T$.*

Proof. Assertion (1) follows from the fact that if $DCTC(Q) = T$, then the dyadic closure phase of the $DCTC$ algorithm computes exactly one split for every quartet, so that $cl(Q) = Q(T)$ by Lemma 3. Therefore, if $Q \subseteq Q'$, then $cl(Q) \subseteq cl(Q')$, so that $Q(T) \subseteq cl(Q') = Q(T')$. Since T and T' are binary trees, it follows that $Q(T) = Q(T')$ and $T = T'$.

Assertion (2) follows from the fact that if $DCTC(Q) =$ *Inconsistent*, then $cl(Q)$ contains two contradictory splits for the same quartet. If $Q \subseteq Q'$, then $cl(Q')$ also contains the same two contradictory splits, and so $DCTC(Q') =$ *Inconsistent*.

Assertion (3) follows from the fact that if $DCTC(Q) =$ *Insufficient*, then $cl(Q)$ does not contain contradictory pairs of splits, and also lacks a split for at least one quartet. If $Q' \subseteq Q$, then $cl(Q')$ also does not contain contradictory pairs of splits and also lacks a split for some quartet. Consequently, $DCTC(Q') =$ *Insufficient*.

Assertion (4) follows from Lemma 2. □

Note that $DCTC(Q) =$ *Insufficient* does not actually imply that $Q \subset Q(T)$ for any tree; that is, it may be that $Q \not\subseteq Q(T)$ for any tree, but $cl(Q)$ may not contain any contradictory splits!

5 Dyadic Closure Method (DCM)

We now describe a new method for tree reconstruction, which we call the *Dyadic Closure Method*, or *DCM*.

Suppose T is a fixed binary tree. From the previous section, we know that if we can find a set Q of quartet splits such that $R_T \subseteq Q \subseteq Q(T)$, then $DCTC(Q)$ will reconstruct T.

One approach to find such a set Q would be to let Q be the set of splits (computed using the Four Point Method) on all possible quartets. However, it is possible that the sequence length needed to ensure that *every* quartet is accurately analyzed might be too large to obtain accurate reconstructions of large trees, or of trees containing short edges.

The approach we take in the Dyadic Closure Method is to use sets of quartet splits based upon *few* quartets, rather than upon all possible quartets (by contrast, other quartet based methods, such as Quartet Puzzling [48, 49], the Buneman tree construction [7], etc. infer quartet splits for all the possible quartets in the tree). Basing the tree reconstruction upon properly selected sets of quartets makes it possible to expect, even from short sequences, that all the quartet splits inferred for the selected subset of quartets will be valid.

Since what we need is a set Q such that $R_T \subseteq Q \subseteq Q(T)$, we need to ensure that we pick a *large enough* set of quartets so that it contains all of R_T, and yet not too large that it contains any invalid quartet splits. Surprisingly, obtaining such a set Q is quite easy (once the sequences are long enough), and we will describe a greedy approach which accomplishes this task. We will also show that the greedy approach can be implemented very efficiently, so that not too many calls to the $DCTC$ algorithm need to be made in order to reconstruct the tree, and analyze the sequence length needed for the greedy approach to succeed with $1 - o(1)$ probability.

We now describe how this is accomplished.

Definition 2: [Q_w, and the *width* of a quartet:]
The *width* of a quartet i, j, k, l is defined to be the maximum of of $h^{ij}, h^{ik}, h^{il}, h^{jk}, h^{jl}, h^{kl}$, where h^{ij} denotes the dissimilarity score between sequences i and j (see Section 2). For each quartet whose width is at most w, compute all feasible splits on that quartet using the Four-Point Method. Q_w is defined to be the set of all such reconstructed splits.

(We note that we could also compute the split for a given quartet of sequences in any number of ways, including maximum likelihood estimation, parsimony, etc., but we will not explore these options in this paper.)

For large enough values of w, Q_w will with high probability contain invalid quartet splits (unless the sequences are very long), while for very small values of w, Q_w will with high probability only contain valid quartet splits (unless the sequences are very short). Since our objective is a set of quartet splits Q such that $R_T \subseteq Q \subset Q(T)$, what we need is a set Q_w such that Q_w contains only valid quartet splits, and yet w is large enough so that all representative quartets are contained in Q_w as well.

We define sets

$$\mathcal{A} = \{w \in \{h^{ij} : 1 \leq i, j \leq n\} : R_T \subseteq Q_w\} \tag{12}$$

and

$$\mathcal{B} = \{w \in \{h^{ij} : 1 \leq i, j \leq n\} : Q_w \subseteq Q(T)\}. \tag{13}$$

In other words, $\mathcal{A}$ is the set of widths w (drawn from the set of dissimilarity scores) which equal or exceed the largest width of any representative quartet, and $\mathcal{B}$ is the set of widths (drawn from the same set) such that all quartet splits of that dissimilarity score are correctly analyzed by the Four Point Method.

It is clear that $\mathcal{B}$ is an initial segment in the list of widths, and that $\mathcal{A}$ is a final segment (these segments can be empty). It is easy to see that if $w \in \mathcal{A} \cap \mathcal{B}$, then $DCTC(Q_w) = T$. Thus, if the sequences are long enough, we can apply $DCTC$ to each of the $O(n^2)$ sets Q_w of splits, and hence reconstruct the tree properly. However, the sequences may not be long enough to ensure that such a w exists; i.e. $\mathcal{A} \cap \mathcal{B} = \emptyset$ is possible! Consequently, we will require that $\mathcal{A} \cap \mathcal{B} \neq \emptyset$, and state this requirement as an hypothesis (later, we will show in Theorem 9 that this hypothesis holds with high probability for sufficiently long sequences):

$$\mathcal{A} \cap \mathcal{B} \neq \emptyset. \tag{14}$$

When this hypothesis holds, we clearly have a polynomial time algorithm, but we can also show that the $DCTC$ algorithm enables a *binary search* approach over the realized widths values, so that instead of $O(n^2)$ calls to the $DCTC$ algorithm, we will have only $O(\log n)$ such calls.

Recall that $DCTC(Q_w)$ is either a tree T, *Inconsistent*, or *Insufficient*.

- **Insufficient:** This indicates that w is too small, because not all representative quartet splits are present, and we should increase w.
- **Tree output:** If this happens, the quartets are consistent with a unique tree, and that tree is returned.

- **Inconsistent:** This indicates that the quartet splits are incompatible, so that no tree exists which is consistent with each of the constraints. In this case, we have computed the split of at least one quartet incorrectly. This indicates that w is too large, and we should decrease w.

If not all representative quartets are inferred correctly, then every set Q_w will be either insufficient or inconsistent with T, perhaps consistent with a different tree. In this case the sequences are too short for the DCM to reconstruct a tree accurately.

We summarize our discussion as follows:

Dyadic Closure Method:

Step 1: Compute the distance matrices d and h (recall that d is the matrix of corrected empirical distances, and h is the matrix of normalized Hamming distances, i.e. the *dissimilarity* score).

Step 2: Do a *binary search* as follows: for $w \in \{h^{ij}\}$, determine Q_w. If $DCTC(Q_w) = T$, for some tree T, then Return T. If $DCTC$ returns *Inconsistent*, then w is too large; decrease w. If $DCTC$ returns *Insufficient*, then w is too small; increase w.

Step 3: If for all w, $DCTC$ applied to Q_w returns *Insufficient* or *Inconsistent*, then Return *Fail*.

We now show that this method accurately reconstructs the tree T if $\mathcal{A} \cap \mathcal{B} \neq \emptyset$ (i.e. if Hypothesis 14 holds).

Theorem 7 *Let T be a fixed binary tree. The Dyadic Closure Method returns T if Hypothesis (14) holds, and runs in $O(n^5 \log n)$ time on any input.*

Proof. If $w \in \mathcal{A} \cap \mathcal{B}$, then $DCTC$ applied to Q_w returns the correct tree T by Theorem 6. Hypothesis (14) implies that $\mathcal{A} \cap \mathcal{B} \neq \emptyset$, hence the Dyadic Closure Method will return a tree if it examines any width in that intersection, hence we need only prove that DCM either examines a width in that intersection, or else reconstructs the correct tree for some other width. This follows directly from Theorem 6.

The running time analysis is easy. Since we do a binary search, the $DCTC$ algorithm is called at most $O(\log n)$ times. The dyadic closure phase of the $DCTC$ algorithm costs $O(n^5)$ time, by Lemma 5, and reconstructing the tree T from $cl(Q)$ uses at most $O(n^5)$ time using standard techniques. □

Note that we have only guaranteed performance for DCM when $\mathcal{A} \cap \mathcal{B} \neq \emptyset$; indeed, when $\mathcal{A} \cap \mathcal{B} = \emptyset$, we have no guarantee that DCM will return the correct tree. In the following section, we will discuss the ramifications of this requirement for accuracy, and show that the sequence length needed to guarantee that $\mathcal{A} \cap \mathcal{B} \neq \emptyset$ with high probability is actually not very large.

6 Performance of DCM for tree reconstruction under the Neyman 2-state model

In this section we analyse the performance of a distance-based application of DCM to reconstruct trees under the Neyman 2-state model under two standard distributions.

6.1 Analysis of the Dyadic Closure Method

Our analysis of the Dyadic Closure Method has two parts. In the first part, we establish the probability that the estimation (using the Four Point Method) of the split induced by a given quartet will be correct. In the second part, we establish the probability that the greedy method we use will contain all short quartets but no incorrectly analysed quartet.

Our analysis of performance of the DCM method depends heavily on the following two lemmas:

Lemma 4 [Azuma–Hoeffding inequality, see [3]]
Suppose $X = (X_1, X_2, \ldots, X_k)$ are independent random variables taking values in any set S, and $L : S^k \to \mathbb{R}$ is any function that satisfies the condition: $|L(\mathbf{u}) - L(\mathbf{v})| \leq t$ whenever $\mathbf{u}$ and $\mathbf{v}$ differ at just one coordinate. Then,

$$\begin{aligned} \mathbb{P}[L(\mathbf{X}) - \mathbb{E}[L(\mathbf{X})] \geq \lambda] &\leq \exp\left(-\frac{\lambda^2}{2t^2k}\right), \\ \mathbb{P}[L(\mathbf{X}) - \mathbb{E}[L(\mathbf{X}] \leq -\lambda] &\leq \exp\left(-\frac{\lambda^2}{2t^2k}\right). \qquad \square \end{aligned}$$

We define the (standard) L_∞ metric on distance matrices, $L_\infty(d, d') = \max_{ij} |d_{ij} - d'_{ij}|$. The following discussion relies upon definitions and notations from Section 2.

Lemma 5 *Let T be an edge weighted binary tree with four leaves i, j, k, l, let D be the additive distance matrix on these four leaves defined by T, and let x be the weight on the single internal edge in T. Let d be an arbitrary distance matrix on the four leaves. Then the Four-Point Method infers the split induced by T from d if $L_\infty(d, D) < x/2$.*

Proof. Suppose that $L_\infty(d, D) < x/2$, and assume that T has the valid split $ij|kl$. Note that the Four Point Method will return a single quartet split $ij|kl$ if and only if $d_{ij} + d_{kl} < \min\{d_{ik} + d_{jl}, d_{il} + d_{jk}\}$. Note that since $ij|kl$ is a valid quartet split in T, $D_{ij} + D_{kl} + 2x = D_{ik} + D_{jl} = D_{il} + D_{jk}$. Since $L_\infty(d, D) < x/2$, it follows that

$$d_{ij} + d_{kl} < D_{ij} + D_{kl} + x,$$

$$d_{ik} + d_{jl} > D_{ik} + D_{jl} - x,$$

and

$$d_{il} + d_{jk} > D_{il} + D_{jk} - x,$$

with the consequence that $d_{ij} + d_{kl}$ is the (unique) smallest of the three pairwise sums. $\square$

Recall that DCM applied to the Neyman 2-state model computes quartet splits using the Four Point Method (FPM).

Theorem 8 *Assume that z is a lower bound for the transition probability of any edge of a tree T in the Neyman 2-state model, $y \geq \max E^{ij}$ is an upper bound on the compound changing probability over all ij paths in a quartet q of T. The probability that FPM fails to return the correct quartet split on q is at most*

$$18 \exp\left(-(1 - \sqrt{1 - 2z})^2 (1 - 2y)^2 k/8)\right). \tag{15}$$

Proof. First observe from Formula (1) that z is also a lower bound for the compound changing probability for the path connecting *any* two vertices of T.
We know that FPM returns the appropriate subtree given the additive distances D_{ij}; furthermore, if $|d_{ij} - D_{ij}| \leq -\frac{1}{4}\log(1-2z)$ for all i, j, then FPM will also return the appropriate subtree on all $ijkl$, by Lemma 5. Consequently,

$$\mathbb{P}[FPM \text{ errs}] \leq \mathbb{P}\left[\exists i,j : |D_{ij} - d_{ij}| > -\frac{1}{4}\log(1-2z)\right]. \tag{16}$$

Hence by (16), we have

$$\mathbb{P}[FPM \text{ errs}] \leq \sum_{ij} \mathbb{P}\left[|D_{ij} - d_{ij}| > -\frac{1}{4}\log(1-2z)\right]. \tag{17}$$

For convenience we drop the subscripts when we analyse the events in (17) and just write D and d, we write p for the corresponding transition probability E^{ij} and $\hat{p}$ for the relative frequency h^{ij}. By simple algebra,

$$|D - d| = \frac{1}{2}\log\frac{1-2p}{1-2\hat{p}} \qquad \text{if } p < \hat{p} \tag{18}$$

$$|D - d| = \frac{1}{2}\log\frac{1-2\hat{p}}{1-2p} \qquad \text{if } p \geq \hat{p}. \tag{19}$$

Now we consider the probability that the four point method fails; i.e. the event estimated in (17). If $p \geq \hat{p}$, then formula (19) applies, so that $\mathbb{P}[FPM \text{ errs}]$ is algebraically equivalent to

$$p - \hat{p} \geq \frac{1}{2}\left[(1-2z)^{-1/2} - 1\right](1-2p). \tag{20}$$

This can then be analysed using Lemma 4. The other case is where $p < \hat{p}$. In this case, formula (18) applies, and $\mathbb{P}[FPM \text{ errs}]$ is algebraically equivalent to

$$\frac{\hat{p} - p}{1 - 2\hat{p}} \geq \frac{1}{2}\left[(1-2z)^{-1/2} - 1\right]. \tag{21}$$

Select an arbitrary positive number ϵ. Then $\hat{p} - p \geq (1-2p)\epsilon$ with probability

$$\exp(-\epsilon^2(1-2p)^2 k/2) \tag{22}$$

by Lemma 4. If $\hat{p} - p < (1-2p)\epsilon$, then

$$\frac{1}{1-2\hat{p}} < \frac{1}{(1-2p) - 2\epsilon(1-2p)} = \frac{1}{(1-2p)}\frac{1}{(1-2\epsilon)}.$$

Hence

$$\begin{aligned}
& \mathbb{P}\left[\frac{\hat{p}-p}{1-2\hat{p}} \geq \frac{1}{2}\left[(1-2z)^{-1/2} - 1\right]\right] \\
\leq \; & \mathbb{P}\left[\frac{\hat{p}-p}{(1-2p)(1-2\epsilon)} \geq \frac{1}{2}\left[(1-2z)^{-1/2} - 1\right]\right] + \exp(-\epsilon^2(1-2p)^2 k/2) \\
\leq \; & \exp(-\epsilon^2(1-2p)^2 k/2) \qquad (23) \\
+ \; & \exp\big(-(1-2p)^2(1-2\epsilon)^2\left[(1-2z)^{-1/2} - 1\right]^2 k/8\big). \qquad (24)
\end{aligned}$$

Note that $\epsilon = (1/2)[1 - (1-2z)^{1/2}]$ is the optimal choice. Formulae (22, 23, and 24) contribute each the same exponential expression to the error, and (16) or (17) multiplies it by 6, due to the six pairs in the summation. □

This allows us to state our main result. First, recall the definition of *depth* from Section 2.

Theorem 9 *Suppose k sites evolve under the Neyman 2-state model on a binary tree T, so that for all edges e, $p(e) \in [f,g]$, where we allow f,g to be functions of n. Then the Dyadic Closure Method reconstructs T with probability $1-o(1)$, if*

$$k > \frac{c \cdot \log n}{(1-\sqrt{1-2f})^2(1-2g)^{4depth(T)+6}} \tag{25}$$

where c is a fixed constant.

Proof. It suffices to show that Hypothesis (14) holds. For k evolving sites (i.e. sequences of length k), and $\tau > 0$, let us define the following two sets, $S_\tau = \{\{i,j\} : h^{ij} < 0.5 - \tau\}$ and $Z_\tau = \{q \in \binom{[n]}{4} :$ for all $i,j \in q, \{i,j\} \in S_{2\tau}\}$, and the following four events,

$$A = \quad Q_{short}(T) \subseteq Z_\tau, \tag{26}$$

$$B_q = \quad \text{FPM correctly returns the split of the quartet } q \in \binom{[n]}{4}, \tag{27}$$

$$B = \bigcap_{q \in Z_\tau} B_q, \tag{28}$$

$$C \quad = \quad S_{2\tau} \text{ contains all pairs } \{i,j\} \text{ with } E^{ij} < 0.5 - 3\tau \text{ and no pair } \{i,j\} \text{ with } E^{ij} \geq 0.5 - \tau. \tag{29}$$

Thus, $\mathbb{P}[\mathcal{A} \cap \mathcal{B} \neq \emptyset] \geq \mathbb{P}[A \cap B]$. Define

$$\lambda = (1-2g)^{2depth(T)+3}. \tag{30}$$

We claim that:

$$\mathbb{P}[C] \geq 1 - (n^2 - n)e^{-\tau^2 k/2} \tag{31}$$

and

$$\mathbb{P}[A|C] = 1, \ \text{ if } \ \tau \leq \lambda/6\ . \tag{32}$$

To establish (31), first note that h^{ij} satisfies the hypothesis of the Azuma-Hoeffding Inequality (Lemma 4 with X_i the sequence of states for site i and $t = 1/k$). Suppose $E^{ij} \geq 0.5 - \tau$. Then,

$$\begin{aligned} &\mathbb{P}[\{i,j\} \in S_{2\tau}] = \mathbb{P}[h^{ij} < 0.5 - 2\tau] \\ \leq\ &\mathbb{P}\left[h^{ij} - E^{ij} \leq 0.5 - 2\tau - E^{ij}\right] \leq \mathbb{P}\left[h^{ij} - \mathbb{E}[h^{ij}] \leq -\tau\right] \leq e^{-\tau^2 k/2}. \end{aligned}$$

Since there are at most $\binom{n}{2}$ pairs $\{i,j\}$, the probability that at least one pair $\{i,j\}$ with $E^{ij} \geq 0.5 - \tau$ lies in $S_{2\tau}$ is at most $\binom{n}{2} e^{-\tau^2 k/2}$. By a similar argument, the probability that $S_{2\tau}$ fails to contain a pair $\{i,j\}$ with $E^{ij} < 0.5 - 3\tau$ is also at most $\binom{n}{2} e^{-\tau^2 k/2}$. These two bounds establish (31).

We now establish (32). For $q \in R(T)$ and $i,j \in q$, if a path $e_1 e_2 ... e_t$ joins leaves i and j, then $t \leq 2depth(T) + 3$ by the definition of $R(T)$. Using these facts, (1), and the bound $p(e) \leq g$, we obtain $E^{ij} = 0.5\,[1 - (1-2p_1)\cdots(1-2p_t)] \leq 0.5(1-\lambda)$. Consequently, $E^{ij} < 0.5 - 3\tau$ (by assumption that $\tau \leq \lambda/6$) and so $\{i,j\} \in S_{2\tau}$ once we condition on the occurrence of event C. This holds for all $i,j \in q$, so by definition of Z_τ we have $q \in Z_\tau$. This establishes (32).

Define a set

$$X = \left\{ q \in \binom{[n]}{4} \ : \ \max\{E^{ij} : i,j \in q\} < 0.5 - \tau \right\}$$

(note that X is not a random variable, while Z_τ, S_τ are). Now, for $q \in X$, the induced subtree in T has mutation probability at least $f(n)$ on its central edge, and mutation probability of no more than $\max\{E^{ij} : i,j \in q\} < 0.5 - \tau$ on any pendant edge. Then, by Theorem 8 we have:

$$\mathbb{P}[B_q] \geq 1 - 18\exp\left(-(1-\sqrt{1-2f})^2\tau^2k/8\right). \tag{33}$$

whenever $q \in X$. Also, the occurrence of event C implies that

$$Z_\tau \subseteq X \tag{34}$$

since if $q \in Z_\tau$, and $i,j \in q$, then $i,j \in S_{2\tau}$, and then (by event C), $E^{ij} < 0.5 - \tau$, hence $q \in X$. Thus, since $B = \bigcap_{q \in Z_\tau} B_q$, we have:

$$\mathbb{P}[B \cap C] = \mathbb{P}[(\bigcap_{q \in Z_\tau} B_q) \cap C] \geq \mathbb{P}[(\bigcap_{q \in X} B_q) \cap C]$$

where the second inequality follows from (34), as this shows that when C occurs, $\bigcap_{q \in Z_\tau} B_q \supseteq \bigcap_{q \in X} B_q$. Invoking the Bonferonni inequality, we deduce that

$$\mathbb{P}[B \cap C] \geq 1 - \sum_{q \in X} \mathbb{P}[\overline{B_q}] - \mathbb{P}[\overline{C}]. \tag{35}$$

Thus, from above,

$$\mathbb{P}[A \cap B] \geq \mathbb{P}[A \cap B \cap C] = P[B \cap C]$$

(since $\mathbb{P}[A|C] = 1$), and so, by (33) and (35),

$$\mathbb{P}[A \cap B] \geq 1 - 18\binom{n}{4}\exp\left(-(1-\sqrt{1-2f})^2\tau^2k/8\right) - (n^2-n)e^{-\tau^2k/2}.$$

Formula (25) follows by an easy calculation. □

6.2 Distributions on trees

In the previous section we provided an upper bound on the sequence length that suffices for the Dyadic Closure Method to achieve an accurate estimation with high probability, and this upper bound depends critically upon the *depth* of the tree. In this section, we determine the depth of a random tree under two simple models of random binary trees.

These models are the *uniform* model, in which each tree has the same probability, and the *Yule-Harding* model, studied in [2, 8, 27] (the definition of this model is given in the proof). This distribution is based upon a simple model of speciation, and results in "bushier" trees than the uniform model. The following results are needed to analyse the performance of our method on random binary trees.

Theorem 10 (i) *For a random semilabelled binary tree T with n leaves under the uniform model, depth$(T) \leq (2+o(1))\log_2\log_2(2n)$ with probability $1-o(1)$.*
(ii) *For a random semilabelled binary tree T with n leaves under the Yule-Harding distribution, after suppressing the root, depth$(T) = (1+o(1))\log_2\log_2 n$ with probability $1-o(1)$.*

Proof. This proof relies upon the definition of an *edi-subtree*, which we now define. An *edge-deletion induced* subtree (edi-subtree) is a rooted subtree of a tree T induced by deleting an edge but not its endpoints.

We now establish (i). Recall that the number of all semilabelled binary trees is $(2n-5)!!$. Now there is a unique (unlabelled) binary tree F on 2^t+1 leaves with the following description: one endpoint of an edge is identified with the degree two root of a complete binary tree with 2^t leaves. The number of semilabelled binary trees whose underlying topology is F is $(2^t+1)!/2^{2^t-1}$. This is fairly easy to check, and this also follows from Burnside's Lemma as applied to the action of the symmetric group on trees, as was first observed by [32] in this context. A rooted semilabelled binary forest is a forest on n labelled leaves, m trees, such that every tree is either a single leaf or a binary tree which is rooted at a vertex of degree 2. It was proved by Carter *et al.* [11] that the number of rooted semilabelled binary forests is $N(n,m)=\binom{2n-m-1}{m-1}(2n-2m-1)!!$. Now we apply the probabilistic method. We want to set a number t large enough, such that the total number of edi-subtrees of depth at least t in the set of all semilabelled binary trees on n vertices is $o((2n-5)!!)$. The theorem then follows for this number t. We will show that some $t=(2+o(1))\log_2\log_2(2n)$ suffices. We count ordered pairs in two ways, as usual: Let E_t denote the number of edi-subtrees of depth at least t (edi-subtrees induced by internal edges and leaf edges combined) counted over of all semilabelled trees. Then E_t is equal to the number of ways to construct a rooted semilabelled binary forest on n leaves of 2^t+1 trees, then use the 2^t+1 trees as leaf set to create all F-shaped semilabelled trees (as described above), with finally attaching the leaves of F to the roots of the elements of the forest. Then $E_t=\left((2^t+1)!/2^{2^t-1}\right)N(n,2^t+1)$. Hence everything boils down to finding a t for which

$$\left((2^t+1)!/2^{2^t-1}\right)\binom{2n-2^t-2}{2^t}(2n-2^{t+1}-3)!!=o((2n-5)!!).$$

Clearly $t=(1+\delta)\log_2\log_2(2n)$ suffices.

We now consider (ii). First we describe the proof for the usual rooted Yule-Harding trees. These trees are defined by the following constructive procedure. Make a random permutation $\pi_1,\pi_2,\dots,\pi_n$ of the n leaves, and join π_1 and π_2 by edges to a root R of degree 2. Add each of the remaining leaves sequentially, by randomly (with the uniform probability) selecting an edge incident to a leaf in the tree already constructed, subdividing the edge, and make π_i adjacent to the newly introduced node. For the depth of a Yule-Harding tree, consider the following recursive labelling of the edges of the tree. Call the edge $\pi_i R$ (for $i=1,2$) "i new". When π_i is added ($i\geq 3$) by insertion into an edge with label "j new", we give label "i new" to the leaf edge added, give label "j new" to the leaf part of the subdivided edge, and turn the label "j new" into "j old" on the other part of the subdivided edge. Clearly, after l insertions, all numbers $1,2,...,l$ occur exactly once with label new, in each occasion labelling leaf edges. The following which may help in understanding the labelling: edges with "old" label are exactly the internal edges and j is the smallest label in the subtree separated by an edge labelled "j old" from the root R, any time during the labelling procedure.

We now derive an upper bound for the probability that an edi-subtree of depth d develops. If it happens, then a leaf edge inserted at some point has to grow a deep edi-subtree on one side. Let us denote by T_i^R the rooted random tree that we already obtained with i leaves. Consider the probability that the most recently inserted edge "i new" ever defines an edi-subtree with depth d. Such an event can happen in two ways: this edi-subtree may emerge on the leaf side of the edge or on the tree side of the edge (these sides are defined when the edge is created). Let us denote these probabilities by $\mathbb{P}[i,\mathrm{OUT}|T_i^R]$ and $\mathbb{P}[i,\mathrm{IN}|T_i^R]$, since these probabilities may depend on the shape of tree already obtained (and, in fact, the second probability does so depend on the shape of T_i^R). We estimate these quantities with tree-independent quantities.

For the moment, take for granted the following inequalities:

$$\mathbb{P}[i,\mathrm{OUT}|T_i^R]\leq\mathbb{P}[i,\mathrm{IN}|T_i^R], \tag{36}$$

$$\mathbb{P}[i,\mathrm{IN}|T_i^R]\leq\epsilon(d,n) \tag{37}$$

for some function $\epsilon(d,n)$ defined below. Clearly

$$\mathbb{P}[\exists \text{ depth } d \text{ edi-subtree }] \leq \sum_{i=1}^{n} \sum_{T_i^R} \mathbb{P}[i,\mathrm{OUT}|T_i^R]\mathbb{P}[T_i^R] + \mathbb{P}[i,\mathrm{IN}|T_i^R]\mathbb{P}[T_i^R], \tag{38}$$

and using (36) and (37), (38) simplifies to

$$\mathbb{P}[\exists \text{ depth } d \text{ edi-subtree }] \leq 2n\epsilon(d,n). \tag{39}$$

We now find an appropriate $\epsilon(d,n)$.

For convenience we assume that $2^s = n-2$, since it simplifies the calculations. Set $k = 2^{d-1}-1$, it is clear that at least k properly placed insertions are needed to make the current edge "i new " have depth d on its tree side. Indeed, π_i was inserted into a leaf edge labelled "j new", and one side of this leaf edge is still a leaf, which has to develop into depth $d-1$, and this development requires at least k new leaf insertions.

Focus now entirely on the k insertions that change "j new" into an edi-subtree of depth $d-1$. Rank these insertions by $1,2,...,k$ in order, and denote by 0 the original "j new" leaf edge. Then any insertion ranked $i \geq 1$ may go into one of those ranked $0,1,...,i-1$. Call the function which tells for $i=1,2,...,k$, which depth i is inserted into, a *core*. Clearly, the number of cores is at most k^k.

We now estimate the probability that a fixed core emerges. For any fixed $i_1 < i_2 < ... < i_k$, the probability that inserting π_{i_j} will make the insertion enumerated under depth j, for all $j=1,2,...,k$, is at most

$$\frac{1}{i_1-1}\cdot\frac{1}{i_2-1}\cdots\frac{1}{i_k-1}$$

by independence. Summarizing our observations,

$$\begin{aligned} \mathbb{P}[i,\mathrm{IN}|T_i^R] &\leq k^k\sigma_{n-i}^k\left(1/i,1/(i+1),...,1/(n-1)\right) \\ &\leq \sigma_{n-2}^k\left(1/2,1/3,...,1/(n-1)\right), \end{aligned} \tag{40}$$

where σ_m^k is the symmetric polynomial of m variables of degree k. We set $\epsilon(n,d) = \sigma_{n-2}^k(1/2,1/3,...,1/(n-1))$. To estimate (40), observe that any term in $\sigma_{n-2}^k(1/2,1/3,...,1/(n-1))$ can be described as having exactly a_i reciprocals of integers substituted from the interval $(2^{-(i+1)},2^{-i}]$. The point is that those reciprocals differ little in each of those intervals, and hence a close estimate is possible. A generic term of σ_{n-2}^k as described above is estimated from above by

$$2^{-(1\cdot a_1+2\cdot a_2+...+(s-1)a_{s-1})}. \tag{41}$$

Hence $\epsilon(n,d)$ is at most

$$\sum_{\substack{a_1+a_2+\cdots+a_{s-1}=k \\ a_i\leq 2^i}} \binom{2}{a_1}\binom{4}{a_2}\binom{8}{a_3}\cdots\binom{2^{s-1}}{a_{s-1}}2^{-(1\cdot a_1+2\cdot a_2+...+(s-1)a_{s-1})}, \tag{42}$$

by (41). Since

$$\binom{2^i}{a_i}2^{-ia_i} \leq \frac{1}{a_i!},$$

(42) is less than or equal

$$\sum_{\substack{a_1+a_2+...+a_{s-1}=k \\ a_i\leq 2^i}} \frac{1}{a_1!a_2!\cdots a_{s-1}!}. \tag{43}$$

Observe that the number of terms in (43) is at most the number of compositions of k into $s-1$ terms, $\binom{k+s-2}{s-2}$. The product of factorials is minimized (irrespective of $a_i \leq 2^i$) if all a_i's are taken equal. Hence, setting $k = s^{1+\delta}$ for any fixed $\delta > 0$, (43) is at most

$$\left(\frac{(k+s-2)^{s-2}}{(s-2)!}\right) \Big/ \left((\frac{k}{s-1})!^k\right),$$

and hence

$$\epsilon(n,d) \leq k^k \left(\frac{(k+s-2)^{s-2}}{(s-2)!}\right) \Big/ \left((\frac{k}{s-1})!^k\right) \leq n^{-c\log n \log\log n},$$

and (39) goes to zero. For the depth d, our calculation yields $(1+\delta+o(1))\log_2\log_2 n$ with probability $1-o(1)$.

We leave the establishment of (36) to the reader. Now, to obtain a similar result for unrooted Yule-Harding trees, just repeat the argument above, but use the unrooted T_i instead of the rooted T_i^R. The probability of any T_i is the sum of probabilities of $2i-3$ rooted T_i^R's, since the root could have been on every edge of T_i. Hence Formula (37) has to be changed for $\mathbb{P}[i, \mathrm{IN}|T_i] \leq (2n-3)\epsilon(d,n)$. With this change the same proof goes through, and the threshold does not change. □

6.3 The performance of DCM and two other distance methods for inferring trees in the Neyman 2-state model

In this section we describe the convergence rate for the DCM method, and compare it briefly to the rates for two other distance-based methods, the Agarwala *et al.* 3-approximation algorithm [1] for the L_∞-nearest tree, and Neighbor-joining [41]. We make the natural assumption that all methods use the same corrected empirical distances from Neyman 2-state model trees.

The neighbor-joining method is perhaps the most popular distance-based method used in phylogenetic reconstruction, and in many simulation studies (see [33, 34, 42] for an entry into this literature) it seems to outperform other popular distance based methods. The Agarwala *et al.* algorithm [1] is a distance-based method which provides a 3-approximation to the L_∞ nearest tree problem, so that it is one of the few methods which provide a provable performance guarantee with respect to any relevant optimization criterion. Thus, these two methods are two of the most promising distance-based methods against which to compare our method. Both these methods use polynomial time.

In [23], Farach and Kannan analyzed the performance of the 3-approximation algorithm with respect to tree reconstruction in the Neyman 2-state model, and proved that the Agarwala *et al.* algorithm converged quickly for the *variational distance* (a related but different concern). Recently, Kannan [35] extended the analysis and obtained the following counterpart to (25): If T is a Neyman 2-state model tree with mutation rates in the range $[f,g]$, and if sequences of length k' are generated on this tree, where

$$k' > \frac{c' \cdot \log n}{f^2(1-2g)^{2diam(T)}} \tag{44}$$

for an appropriate constant c', and where $diam(T)$ denotes the "diameter" of T, then with probability $1-o(1)$ the result of applying Agarwala *et al.* to corrected distances will be a tree with the same topology as the model tree. In [5], Atteson proved an identical statement for neighbor-joining, though with a different constant (the constant for neighbor-joining is smaller than the constant for the Agarwala *et al.* algorithm, suggesting that neighbor-joining should be accurate from shorter sequences than Agarwala *et al.*, on any tree in the Neyman 2-state model).

Comparing this formula to (25), we note that the comparison of depth and diameter is the issue, since $(1-\sqrt{1-2f})^2 = \Theta(f^2)$ for small f. It is easy to see that $diam(T) \geq 2depth(T)$ for binary trees T, but the diameter of a tree can in fact be quite large (up to $n-1$), while the depth is never more than $\log n$. Thus, for every fixed range of mutation probabilities, the sequence length that suffices to guarantee accuracy for the neighbor-joining or Agarwala *et al.* algorithms can be quite large (i.e. it can grow exponentially in the number of leaves), while the sequence length that suffices for the Dyadic Closure Method will never grow more than polynomially. See also [20, 21, 40] for further studies on the sequence length requirements of these methods.

The following table summarizes the worst-case analysis of the the sequence length that suffices for the Dyadic Closure Method to obtain an accurate estimation of the tree, for a fixed and a variable range of mutation probabilities. We express these sequence lengths as functions of the number n of leaves, and use results from (25) and Section 6.2 on the depth of random binary trees. "Best case" (respectively, "worst case") trees refers to best case (respectively worst case) *shape* with respect to the sequence length needed to recover the tree as a function of the number n of leaves. Best case trees for DCM are those whose depth is small with respect to the number of leaves; these are the *caterpillar* trees – i.e. trees which are formed by attaching n leaves to a long path. Worst case trees for DCM are those trees whose depth is large with respect to the number of leaves; these are the *complete binary trees.* All trees are assumed to be binary.

	range of mutation probabilities on edges:	
	$[f,g]$ f,g are constants	$\left[\frac{1}{\log n}, \frac{\log\log n}{\log n}\right]$
worst case trees	polynomial	polylog
best case trees	logarithmic	polylog
random (uniform) trees	polylog	polylog
random (Yule-Harding) trees	polylog	polylog

Table 1. Sequence length needed by DCM to return trees under the Neyman 2*-state model*

7 Summary

We have provided upper and lower bounds on the sequence length k required for accurate tree reconstruction, and shown that in certain cases these two bounds are surprisingly close in their order of growth with n. It is quite possible that even better upper bounds could be obtained by a tighter analysis of our DCM approach, or perhaps by analyzing other methods.

Our results may provide a nice analytical explanation for some of the surprising results of recent simulation studies (see for example, [30]) which found that trees on hundreds of species could be accurately reconstructed from sequences of only a few thousand sites long. For molecular biology the results of this paper may be viewed, optimistically, as suggesting that large trees can be reconstructed accurately from realistic length sequences. Nevertheless, some caution is required, since the evolution of real sequences will only be approximately described by these models, and the presence of very short and/or very long edges will call for longer sequence lengths.

8 Acknowledgements

Thanks are due to Sampath Kannan for extending the analysis of [23] to consider the topology estimation, and to David Bryant and Éva Czabarka for proofreading the manuscript.

Tandy Warnow was supported by an NSF Young Investigator Award CCR-9457800, a David and Lucille Packard Foundation fellowship, and generous research support from the Penn Research Foundation and Paul Angello. Michael Steel was supported by the New Zealand Marsden Fund and the New Zealand Ministry of Research, Science and Technology. Péter L. Erdős was supported in part by the Hungarian National Science Fund contracts T 016 358. László Székely was supported by the National Science Foundation grant DMS 9701211, the Hungarian National Science Fund contracts T 016 358 and T 019 367, and and European Communities (Cooperation in Science and Technology with Central and Eastern European Countries) contract ERBCIPACT 930 113. This research started in 1995 when the authors enjoyed the hospitality of DIMACS during the Special Year for Mathematical Support to Molecular Biology, and was completed in 1997 while enjoying the hospitality of Andreas Dress, at Universität Bielefeld, in Germany.

References

[1] R. Agarwala, V. Bafna, M. Farach, B. Narayanan, M. Paterson, and M. Thorup, On the approximability of numerical taxonomy: fitting distances by tree metrics, *Proceedings of the 7th Annual ACM-SIAM Symposium on Discrete Algorithms*, 1996, 365–372.

[2] D. J. Aldous, Probability distributions on cladograms, in: *Discrete Random Structures*, eds. D. J. Aldous and R. Permantle, Springer-Verlag, IMA Vol. in Mathematics and its Applications. Vol. 76, 1995, 1–18.

[3] N. Alon and J. H. Spencer, *The Probabilistic Method,* John Wiley and Sons, New York, 1992.

[4] A. Ambainis, R. Desper, M. Farach, and S. Kannan. Nearly tight bounds on the learnability of evolution. To appear, *Proc. of the 1998 Foundations of Computer Science.*

[5] K. Atteson. On the performance of neighbor-joining. To appear, *Proc. of COCOON 1997.*

[6] H.-J. Bandelt and A. Dress, Reconstructing the shape of a tree from observed dissimilarity data, *Adv. App. Math.*, **7** (1986), 309–343.

[7] V. Berry and O. Gascuel, Inferring evolutionary trees with strong combinatorial evidence, *Proc. COCOON 1997.*

[8] J. K. M. Brown, Probabilities of evolutionary trees, *Syst. Biol.* **43** (1994), 78–91.

[9] D. J. Bryant and M. A. Steel, Extension operations on sets of leaf-labelled trees, *Adv. App. Math.*, **16** (1995), 425–453.

[10] P. Buneman, The recovery of trees from measures of dissimilarity, in *Mathematics in the Archaeological and Historical Sciences*, F. R. Hodson, D. G. Kendall, P. Tautu, eds.; Edinburgh University Press, Edinburgh, 1971, 387–395.

[11] M. Carter, M. Hendy, D. Penny, L. A. Székely, and N. C. Wormald, On the distribution of lengths of evolutionary trees, *SIAM J. Disc. Math.* **3** (1990), 38–47.

[12] J. A. Cavender, Taxonomy with confidence, *Math. Biosci.*, **40** (1978), 271–280.

[13] J. T. Chang and J. A. Hartigan, Reconstruction of evolutionary trees from pairwise distributions on current species, *Computing science and statistics: Proceedings of the 23rd symposium on the interface* (1991), 254–257.

[14] H. Colonius and H. H. Schultze, Tree structure for proximity data, *Brit. J. Math. Stat. Psychol.*, **34** (1981), 167–180.

[15] W.H.E. Day, Computational complexity of inferring phylogenies from dissimilarities matrices, *Information Processing Letters*, 30:215-220, 1989.

[16] W.H.E. Day and D. Sankoff, Computational complexity of inferring phylogenies by compatibility, *Syst. Zoology*, 35(2): 224-229 (1986).

[17] M. C. H. Dekker, *Reconstruction methods for derivation trees*, Master's Thesis, Vrije Universiteit, Amsterdam, 1986.

[18] P. Erdős and A. Rényi, On a classical problem in probability theory, *Magy. Tud. Akad. Mat. Kutató Int. Közl.*, **6** (1961), 215–220.

[19] P. L. Erdős, M. A. Steel, L. A. Székely and T. Warnow, Local quartet splits of a binary tree infer all quartet splits via one dyadic inference rule. *Computers and Artificial Intelligence*, Vol. 16, No. 2, pp.217-227, 1997.

[20] P. L. Erdős, M. A. Steel, L. A. Székely and T. Warnow, Inferring big trees from short quartets. *Proceedings of ICALP 1997.*

[21] P. L. Erdős, M. A. Steel, L. A. Székely and T. Warnow, A few logs suffice to build (almost) all trees -II, Submitted, Theoretical Computer Science, special issue on selected papers from ICALP 1997.

[22] P.L. Erdős, K. Rice, M. Steel, L. Szekely, and T. Warnow, The Short Quartet Method. to appear, *Mathematical Modelling and Scientific Computing,* Principia Scientia.

[23] M. Farach, and S. Kannan, Efficient algorithms for inverting evolution, *Proceedings of the ACM Symposium on the Foundations of Computer Science*, 1996, 230–236.

[24] M. Farach, S. Kannan, and T. Warnow, A robust model for inferring optimal evolutionary trees, *Algorithmica* **13** (1995) 155–179.

[25] J. S. Farris, A probability model for inferring evolutionary trees, *Syst. Zool.*, **22** (1973), 250–256.

[26] J. Felsenstein, Cases in which parsimony or compatibility methods will be positively misleading, *Syst. Zool.*, **27** (1978), 401–410.

[27] E. F. Harding, The probabilities of rooted tree shapes generated by random bifurcation, *Adv. Appl. Probab.* **3** (1971), 44–77.

[28] M. D. Hendy, The relationship between simple evolutionary tree models and observable sequence data, *Syst. Zool.*, **38(4)** (1989), 310–321.

[29] D. Hillis, Approaches for assessing phylogenetic accuracy, *Syst. Biol.* **44** (1995), 3–16.

[30] D. Hillis, Inferring complex phylogenies, *Nature* Vol **383** 12 September, 1996, 130–131.

[31] D. Hillis, J. Huelsenbeck, and D. Swofford, Hobgoblin of phylogenetics? *Nature*, **369**, (1994), 363–364.

[32] M. Hendy, C. Little, and D. Penny, Comparing trees with pendant vertices labelled, *SIAM J. Appl. Math.*, **44** (1984), 1054–1065.

[33] Huelsenbeck, J.1995. Performance of phylogenetic methods in simulation. *Syst. Biol.* 44:17-48.

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

[35] S. Kannan, personal communication.

[36] M. Kimura, Estimation of evolutionary distances between homologous nucleotide sequences. *Proc. Nat. Acad. Sci. USA*, 78: 454-458, 1981.

[37] M. Marcus and H. Minc, *A Survey of Matrix Theory and Matrix Inequalities*, Dover Publications Inc., New York, 1992.

[38] J. Neyman, Molecular studies of evolution: a source of novel statistical problems. in *Statistical Decision Theory and Related Topics*, Gupta, S. S. and J. Yackel (eds), New York, Academic Press, 1971, 1–27.

[39] H. Philippe and E. Douzery, The pitfalls of molecular phylogeny based on four species, as illustrated by the cetacea/artiodactyla relationships, *J. Mammal. Evol.*, **2** (1994), 133–152.

[40] K. Rice and T. Warnow, Parsimony is hard to beat!, Proceedings, COCOON 1997.

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

[42] Saitou, N. and T. Imanishi. 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.

[43] Y. A. Smolensky: A method for linear recording of graphs, *USSR Comput. Math. Phys., 2* (1969), 396–397.

[44] M. A. Steel, The complexity of reconstructing trees from qualitative characters and subtrees, *J. Classification*, **9** (1992), 91–116.

[45] M. A. Steel, Recovering a tree from the leaf colourations it generates under a Markov model, *Appl. Math. Lett.*, **7** (1994), 19–24.

[46] M. A. Steel, L. A Székely, and P. L. Erdős, *The number of nucleotide sites needed to accurately reconstruct large evolutionary trees*, DIMACS Tech. Rep. No. 96-19.

[47] M. A. Steel, L. A. Székely, and M. D. Hendy, Reconstructing trees when sequence sites evolve at variable rates, *Journal of Comput. Biol.*, **1** (1994), 153–163.

[48] K. Strimmer and A. von Haeseler, Quartet Puzzling: a quartet Maximum Likelihood method for reconstructing tree topologies, *Mol. Biol. Evol.*, **13** (1996), 964–969.

[49] K. Strimmer, N. Goldman, and A. von Haeseler, Bayesian probabilities and Quartet Puzzling, *Mol. Biol. Evol.*, **14** (1997), 210–211.

[50] D. L Swofford, G. J. Olsen, P. J. Waddell, D. M. Hillis, Chapter 11: Phylogenetic inference, in: *Molecular Systematics*, D. M. Hillis, C. Moritz, B. K. Mable, eds., 2nd edition, Sinauer Associates, Inc., Sunderland, 1996, 407–514.

[51] N. Takezaki and M. Nei, Inconsistency of the maximum parsimony method when the rate of nucleotide substitution is constant, *J. Mol. Evol.*, **39** (1994), 210–218.

[52] T. Warnow, *Combinatorial algorithms for constructing phylogenetic trees*, PhD thesis, University of California-Berkeley, 1991.

[53] Peter Winkler, personal communication.

[54] K. A. Zaretsky, Reconstruction of a tree from the distances between its pendant vertices, *Uspekhi Math. Nauk (Russian Mathematical Surveys), 20* (1965), 90–92 (in Russian).

[55] A. Zharkikh and W. H. Li, Inconsistency of the maximum-parsimony method: The case of five taxa with a molecular clock, *Syst. Biol.*, **42** (1993), 113–125.