Inferring Phylogenies from LCA-Distances

Ilan Gronau and Shlomo Moran

March 16, 2006
Abstract

Reconstructing phylogenetic trees efficiently and accurately from distance estimates is an ongoing challenge in computational biology, from both practical and theoretical considerations. We study a representation of trees by distances to LCAs (Least Common Ancestors), which seems to combine the theoretical advantages of the Farris transform of additive distances to ultrametrics, with the practical advantages and flexibility of neighbor joining algorithms.

We present a characterization of edge-weighted trees using LCA distances. The approach we study preserves many nice properties of ultrametrics, but it bypasses the need to transform additive distances to ultrametric distances (via the Farris transform). This consequently provides a neighbor joining criterion, and a family of algorithms, which are simpler and more efficient than Saitou&Nei’s NJ. One variant of this family is shown to find the unique dominant LCA-matrix for any nonnegative symmetric matrix, and then used to provide a simpler version of the known 3-approximation of an arbitrary metric by an additive metric.

Our family of algorithms, unlike NJ, is pivotal, in the sense that when the input is not consistent with some tree, the output tree may depend on a root-taxon selected by the algorithm. We present experimental results on data generated according to a well accepted evolutionary model. These experiments indicate that for the right selection of the root, the tree returned by our algorithm is likely to be topologically closer to the true tree than the one returned by NJ. The experimental results also indicate that selecting a root-taxon with smaller evolutionary distance is likely to produce a more accurate tree. A somewhat surprising phenomenon demonstrated by our results is that in this evolutionary model, trees which best approximate the input distances are usually not the trees which best approximate the correct topology.
Chapter 1

Introduction

Phylogenetic reconstruction methods attempt to find the evolutionary history of a given set of species (taxa). This history is usually described by an edge-weighted tree, where edges correspond to different branches of evolution, and the weight of an edge corresponds to the amount of evolutionary change on that particular branch. Distance-based phylogenetic reconstruction methods try to find this tree using estimates on distances along the tree; these distances correspond to total weight of paths. Most commonly used methods construct trees from pairwise-distance matrices, i.e. estimates on weights of paths connecting taxon-pairs.

Distance-based methods typically deal with two scenarios: accurate data and noisy data. In the first scenario we assume that our input is consistent with some tree. It is important in this case that the distance-estimates we use as input hold sufficient information for accurate reconstruction of the tree. Algorithms which achieve this are said to be consistent. In the second (more realistic) scenario, there may be no tree consistent with the input. In such a case the goal is to reconstruct a tree fitting the input in some way.

A distance metric consistent with some tree is called an additive metric [4]. It is known that such a metric has a unique tree (with strictly positive internal edge weights) describing it. After the introduction of this notion in [4], numerous algorithms were proposed, which reconstruct the tree given its additive metric [4, 27, 30]. One class of such algorithms reduces the construction of trees from additive distances to construction of ultrametrics. This reduction (also known as the Farris transform [13]), implies $O(n^2)$ algorithms for constructing trees from additive metrics, and is useful for obtaining various properties of additive metrics (e.g. [16, 1]).

Another common technique for constructing phylogenetic trees, which usually has larger complexity but proved to be reliable in practice, is the neighbor joining scheme. Neighbor-joining is an agglomerative clustering approach, in which at each stage two neighbor-elements are joined to one cluster; this new cluster then replaces them in the set of elements. This approach is used in hierarchical-clustering algorithms such as UPGMA [28], which is also used to
reconstruct ultrametric trees. The neighbor-joining scheme was first used to reconstruct trees from additive metrics by the ADDTREE algorithm [27]. Later, Saitou and Nei proposed the famous neighbor-joining algorithm (NJ) [26, 18]. Since then the neighbor-joining scheme was used in numerous algorithms (such as BIONJ [15], NJML [23] and Weighbor [3] to name a few) which were developed in hope of outperforming the original NJ algorithm on noisy (non-additive) input matrices.

When the input matrix is noisy, one approach is to return a tree whose implied metric is ‘close’ to the input under a certain distance norm (such as the $\ell_p$ norm). Finding the closest tree, however, was shown to be NP-hard for several such norms ($\ell_1, \ell_2$ in [6] and $\ell_\infty$ in [1]). A 3-approximation algorithm for $\ell_\infty$ is given by [1]. This is the only constant-rate approximation known to us in this area, and is based on the reduction of additive distances to ultrametrics (by the above mentioned Farris transform). Another approach in dealing with noisy input, is to test the performance of tree reconstruction algorithms on actual data. Since real phylogenetic data are scarce, it is common to use data generated by simulation of evolution under certain accepted probabilistic models [14, 11, 22]. Notable is the large-scale simulated datasets which have been generated during the past years by Gascuel [7, 8, 25] (see Chapter 3 for full detail).

In recent years, several works suggest using distance estimates other than pairwise distances to reconstruct trees. Levy et al, for instance, describe in [21] how to reconstruct a tree from estimates on total-weights of all subtrees spanning sets of $m$ taxa (where $m$ is some constant). They generalize Saitou&Nei’s NJ algorithm to deal with such input, and prove that their algorithm remains consistent. This generalized NJ algorithm is shown on simulated data to have a better success-rate than other common methods. In [25] the use of taxon-triplets is suggested to improve the estimates of pairwise distances. Instead of estimating the distance between two taxa $i, j$ by the ‘most-likely’ tree (path) spanning them, another taxon is introduced, and this distance is estimated using the tree spanning the triplet $i, j, k$.

In this paper we introduce a characterization of tree metrics by LCA-distances, i.e. distances from a selected root to the least common ancestors of all pairs of leaves. These distances can be obtained either from a pairwise distance matrix, or via triplet distances estimated directly from biological data, as proposed by [21, 25]. In both works it is suggested that distance-estimates obtained directly over triplets (and in the case of [21] on even larger taxon-subsets), more accurately portray the original tree, compared to estimates obtained over pairs. This, in turn, potentially leads to more accurate reconstruction.

The approach we present is strongly related to the Farris transform. However, it bypasses the need to transform additive distances to ultrametric distances, and implies a neighbor joining criterion, and a neighbor joining algorithm, which is simpler and more efficient than Saitou&Nei’s NJ algorithm. We use a variant of this algorithm to provide a simple proof of the 3-approximability of arbitrary metrics by additive metrics. Unlike NJ and other neighbor joining
algorithms, the neighbor joining algorithms based on LCA distances are pivotal, in the sense that when the input is not consistent with some tree, the output tree may depend on a root-taxon selected by the algorithm [5].

We present experimental results of two variants of this algorithm on the aforementioned simulated datasets [7, 25]. These results indicate that for the right selection of the root, the tree returned by the algorithm is likely to be topologically closer to the true tree than the one returned by NJ. The simulations also indicate that selecting a root-taxon with smaller evolutionary distance is likely to produce more accurate trees. An interesting phenomenon observed in our experiments is that trees which best approximate the input distances are not the same trees which best approximate the correct topology.

In the next section we provide the needed notations and definitions. In Section 2.1 we present our characterization and in Section 2.2 we analyze theoretical properties of the resulted pivotal neighbor joining algorithms. In Section 2.3 we use a variant of these algorithms to provide a simple proof of the 3-approximation result from [1]. In Chapter 3 we briefly describe the probabilistic model of evolution used in [7, 25], and then present and analyze experimental results of our neighbor joining algorithms on these simulated datasets.

1.1 Definitions and Notations

Let $S$ be a finite set (the set of taxa). A phylogenetic tree over $S$ is an undirected weighted tree $T = (V, E, w : E \rightarrow \mathbb{R}^+)$ whose leaves are the elements of $S$. An edge is external if one of its endpoints is a leaf, and is internal otherwise. Let $r, i, j$ be three (not necessarily distinct) vertices in a tree $T$. $D_T(i, j)$, the distance in $T$ between $i$ and $j$, is the length of the path connecting $i$ and $j$ in $T$. Similarly, $D_T(r; ij)$ is the length of the path connecting $r$ and the center vertex of the 3-finger claw spanning $r, i, j$ (see Figure 1.1); when $T$ is rooted at $r$, this center vertex is the least common ancestor of $i$ and $j$, (note that $D_T(r; ii) = D_T(r, i)$). A matrix over $S$ is a square matrix $M$ whose rows and columns are indexed by the elements of $S$. For a subset $S' \subseteq S$, $M(S')$ denotes the principal submatrix of $M$ induced by the indices in $S'$. For matrices $M_1, M_2$ over $S$, $M_1 \leq M_2$ means that $\forall i, j : M_1(i, j) \leq M_2(i, j)$. 

3
Figure 1.1: **Distance estimates in trees.** The path connecting taxa $i, j$ is marked. $D_T(i, j)$ is the total weight of edges in this path. $v$ is the center vertex of the 3-finger claw spanning $r, i, j$. The path connecting $r, v$ is marked by a dashed line. $D_T(r; ij) = D_T(r, v)$ is the total weight of edges in this path.
Chapter 2

Deepest lca Neighbor Joining

2.1 Characterization of Trees Using LCA-Distances

The main concept we use in this work is that of an LCA-matrix, which will be shown to be equivalent to a representation of a weighted tree by distances from a root-taxon to the least common ancestors of all pairs of leaves. For a given edge-weighted tree $T$ over a set of taxa $S$ and a taxon $r \in S$, the LCA-matrix of $T$ from $r$, $LCA_r T$, is a matrix over $S \setminus \{r\}$ defined by $LCA_r T(i, j) = D_T(r; ij)$. The properties of such matrices are given in the following definition:

**Definition 2.1.** A symmetric non-negative matrix $L$ over a set $S$ is said to be an LCA-matrix iff

1. for all taxa $i \in S$, $L(i, i) = \max_{j \in S} L(i, j)$
2. For every triplet of distinct taxa $(i, j, k)$ in $S$, $L(i, j) \geq \min\{L(i, k), L(j, k)\}$ (this property will be termed the 3-point condition).

Matrices satisfying the above 3-point condition are referred to in [16] as min-ultrametrics. This condition can be phrased as follows: In every three entries of $L$ of the form $(L(i, j), L(i, k), L(j, k))$, the minimal value must appear at least twice. While min-ultrametrics are not metrics (e.g., they do not satisfy the triangle inequality), they share many nice properties of ultrametric distances, with the additional advantage that requiring Property 1 enables representation of general tree metrics (rather than only ultrametric trees).

**Theorem 2.2.** Let $S$ be a set of cardinality $n - 1$. A symmetric non-negative matrix $L$ over $S$ is an LCA-matrix iff there exists a weighted tree $T$ over a set of $n$ taxa $S \cup \{r\}$ s.t. $L = LCA_r T$, i.e. for all $i, j \in S$, $D_T(r; ij) = L(i, j)$.

**Proof.** $\Leftarrow$ Suppose that $T$ is a weighted tree over the taxon-set $S \cup \{r\}$, and let $L = LCA_r T$. Assume w.l.o.g. that $T$ has no zero-weight edges. It is clear that $\forall i, j \in S : D_T(r, i) \geq D_T(r; ij)$, which implies Property 1 of the definition. Now observe the subtree spanned by $r, i, j, k$. If its topology is a star, then $L(i, j) = L(i, k)$. Therefore, $L(i, j) = L(i, k)$ and $L(j, k) = L(i, k)$, which implies the 3-point condition.
Figure 2.1: **The 3-point condition for LCA-distances.** Observe the subtree spanning $r, i, j, k$ (marked edges).  

a) If its topology is a star (4-finger claw), with center-vertex $v$, then $D_T(r; ij) = D_T(r; ik) = D_T(r; jk) = D_T(r; v)$.  

b) Otherwise, w.l.o.g. $i$ is paired up with $r$ as illustrated, and $D_T(r; ij) = D_T(r; ik) = D_T(r, v) < D_T(r, u) = D_T(r; jk)$.

$L(i, k) = L(j, k)$, and the minimum value appears in $\{L(i, j), L(i, k), L(j, k)\}$ three times. If $i$ is paired up with $r$ in this quartet then $L(i, j) = L(i, k) < L(j, k)$ (see Figure 2.1), and the minimum value appears twice. The same can be argued for the other two possible topologies of this subtree, proving Property 2.

$\Rightarrow$ To prove the other direction, suppose $L$ satisfies both conditions. We will use the following lemma, which follows immediately from the three points condition:

**Lemma 2.3.** Let $L$ be an LCA-matrix over $S$, and let $\hat{i}, \hat{j}$ be two distinct elements of $S$ s.t. $\forall k \neq \hat{i}, \hat{j} : L(\hat{i}, \hat{j}) \geq \max\{L(\hat{i}, k), L(\hat{j}, k)\}$. Then $\forall k \neq \hat{i}, \hat{j} : L(\hat{i}, k) = L(\hat{j}, k)$.

Using Lemma 2.3 above we prove by induction on the number of taxa $n$, that there is a tree $T$ over $S \cup \{r\}$ satisfying the theorem. If $n = 2$ then $L$ is a scalar $[w]$, and $T$ consists of a single edge of weight $w$. If $n > 2$, let $L(\hat{i}, \hat{j}) = \max_{i \neq j} \{L(i, j)\}$ be a maximal off-diagonal entry of $L$. Clearly, $L(\hat{i}, \hat{j})$ satisfies the assumption of Lemma 2.3. Now, let $v$ be a new vertex, and $L'$ be
the symmetric matrix over $S' = S \setminus \{ i, j \} \cup \{ v \}$, defined as follows:

\[
L'(k, l) = L(k, l) \quad k, l \neq v \\
L'(k, v) = L(k, i) \quad k \neq v \\
L'(v, v) = L(i, j)
\]

Since all the entries of $L'$ except $L'(v, v)$ are identical to the corresponding entries of $L(S \setminus \{ j \})$ (where index $v$ in $L'$ corresponds to index $i$ of $L$), Property 2 of Definition 2.1 holds for $L'$ as it holds for $L$. For the same reason Property 1 holds for all indices in $S' \setminus \{ v \}$. Property 1 holds for $v$ since $L'(v, v) = L(i, j) \geq \max_{k \in S' \setminus \{ i, j \}} L(i, k) = \max_{k \in S' \setminus \{ v \}} L'(v, k)$.

By the induction hypothesis, there is a tree $T'$ over a set of taxa $S' \cup \{ r \}$ s.t. for all $i, j \in S'$, $D_T(r; ij) = L'(i, j)$. Let $T$ be the tree obtained from $T'$ by making $v$ an internal vertex with two daughters $i, j$, where $w(v, i) = L(i, i) - L(i, j) \geq 0$ and $w(v, j) = L(j, j) - L(i, j) \geq 0$. It remains to show that $\forall i, j \in S : L(i, j) = D_T(r; ij)$. For all $i, j \in S \setminus \{ i, j \}$ this holds by the induction hypothesis and the fact that $L(i, j) = L'(i, j)$. For $k \neq i, j$ we have that:

\[
D_T(r; jk) = D_T(r; ik) = D_T(r; vk) = L'(v, k) = L(i, k) = L(j, k).
\]

It remains to prove the equality for the entries $(i, j), (i, i), (j, j)$. By the definition of $T'$ we have that $D_T(r; ij) = D_T(r, v) = L'(v, v) = L(i, j)$. Finally, $D_T(r, i) = D_T(r, v) + w(v, i) = L(i, i)$ by the definition of $w(v, i)$, and by a similar argument $D_T(r, j) = L(j, j)$. \hfill \Box

The above proof implies the following $O(n^2)$ algorithm for tree reconstruction using an LCA-matrix $L$ as input: Start by computing the maximal off-diagonal entry for each row of $L$, and then proceed via $n - 1$ iterations, where at each iteration a neighbors pair $i, j$ are are connected to their parent $v$ and the matrix $L$ is reduced as follows:

1. Find a maximal off-diagonal entry $L(i, j)$, by scanning the $n$ row-maxima
2. Reduce the matrix $L$ to $L'$ by replacing rows $i, j$ by row $v$ as described.
3. Recompute the maximal off-diagonal entries for each row of $L'$, and set $L \leftarrow L'$.

Observe that computing the maximal off-diagonal entry for a row $k \neq v$ in $L'$ is doable in constant time: If the maximal off-diagonal entry in row $k$ of $L$ is $m$, then the maximal off-diagonal entry in $L'$ is $\max \{ m, L'(k, v) \}$. Thus each iteration requires $O(n)$ time.

2.2 The DLCA Algorithm

Since in practice we are rarely able to obtain accurate distance estimates, it is crucial that reconstruction algorithms produce meaningful output even when
the input is not consistent with any tree metric. The algorithm implied by
the proof of Theorem 2.2 is easily adjusted for noisy input matrices. In fact
it fits in the frame of the ‘neighbor-joining’ approach, which is widely used for
reconstructing phylogenies from noisy data.

Neighbor-joining algorithms consist of a main loop which has three stages:

1. Neighbor selection: Select a pair of taxa \(i, j\) optimizing some criterion.

2. Reduction: Replace the taxon-pair with a new vertex \(v\) in the taxon-
   set. Define distances from \(v\) to all other taxa, and recursively solve the
   problem on this smaller set.

3. Neighbor connection: In the returned solution, make \(v\) an internal
   vertex and connect \(i\) and \(j\) to \(v\) with two edges of some defined length.

In each recursive call the number of taxa is reduced by one, until the stopping
condition is met. This process eventually yields a full-binary tree rooted at \(r\).
To obtain a canonical undirected version of the tree, we contract all zero-weight
internal edges. Our algorithm can be presented in the neighbor joining scheme
as follows:

**Deepest LCA Neighbor Joining (DLCA):**
Input: A symmetric nonnegative matrix \(L\).

1. **Stop condition:** If \(L = [w]\) return a tree consisting of a single edge of
   weight \(w\), rooted at \(r\).

2. **Neighbor selection:** Select a pair of distinct taxa \(i, j\), s.t. \(L(i, j)\) is a
   maximal off-diagonal value in \(L\).

3. **Reduction:** Remove \(i, j\) from the taxon-set and add \(v\). Define \(L(v, v) =
   L(i, j)\), and \(\forall k \neq v : L(v, k) = \frac{1}{2}(L(i, k) + L(j, k))\). Recursively call DLCA
   on the resulting matrix.

4. **Neighbor connection:** In the returned tree, add \(i\) and \(j\) as daughters
   of \(v\), with edge-weights: \(w(v, i) = \max\{0, L(i, i) - L(i, j)\}\) and
   \(w(v, j) = \max\{0, L(j, j) - L(i, j)\}\).

Notice that if \(L\) is an LCA-matrix, then by Lemma 2.3 \(\forall k : L(i, k) = L(j, k)\)
in step 2 of the algorithm, thus follows the equivalence of this algorithm to the
one described in the proof of Theorem 2.2. Hence we can claim:

**Theorem 2.4.** The DLCA algorithm is consistent. I.e, if the input matrix \(L\)
is \(\text{LCA}_T\), for some edge-weighted tree \(T\) and a taxon \(r\) of that tree, then the
DLCA algorithm returns \(T\).

The simple neighbor-selection criterion over LCA-distances used by the DLCA
algorithm relates to a similar criterion in ultrametric trees over pairwise dis-
tances. In ultrametric trees we know that the taxon-pair closest to each other
are neighbors. This is the basis of many ultrametric reconstruction algorithms,
such as UPGMA. In such trees, pairwise distances inversely correlate with the depth of the lca, so the the taxon-pair closest to each other is the taxon-pair with deepest lca. However when considering distances to lca’s, we no longer need to assume an ultrametric tree. We can therefore use this neighboring criterion to reconstruct general trees. In general, any algorithm which reconstructs ultrametric trees from pairwise distances can be converted to an algorithm which reconstructs general trees from LCA-distances. The DLCA algorithm as described above is actually such a conversion of the WPGMA algorithm. The algorithm can be generalized in two points without compromising its correctness:

- **Neighbor selection:** Rather than selecting the maximal off-diagonal entry of $L$, we can select any taxon pair satisfying Lemma 2.3. This criterion, in fact, gives us a simple method to pinpoint all neighbor-pairs in $T$ (which do not include $r$). We do not have to choose the ‘deepest’ pair in the tree, as suggested by the original description of DLCA. In this paper we confine ourselves to the original, deepest pair selection.

- **Reduction:** Since for the selected neighbors $i, j$ it holds that $\forall k \neq i, j: \text{LCA}_T(i, k) = \text{LCA}_T(j, k)$, we can set $L(v, k)$ to any value of the form $\alpha L(i, k) + (1 - \alpha)L(j, k)$. Our simulations show that the performance of this algorithm on noisy input matrices could be affected by the selected reduction. We will be interested in two variants:
  
  1. The original (mid-point reduction) variant: $L(v, k) = \frac{1}{2}(L(i, k) + L(j, k))$
  2. The maximal-value reduction: $L(v, k) = \max\{L(i, k), L(j, k)\}$

We found out that this scheme has already been used in [29]. The reduction step in the algorithm proposed there uses a weighted average $L(v, k) = \alpha L(i, k) + (1 - \alpha)L(j, k)$, where $\alpha, 1 - \alpha$ correlate with the number of offspring-taxa of $i, j$ respectively. This algorithm can be seen as the conversion of UPGMA to LCA-distances, as previously discussed. Similarly, the ‘maximal-value’ variant relates to an algorithm proposed in [19, 12]. The latter is discussed in further detail later this section.

The running time of the algorithm may depend on the reduction stage. Specifically, it remains $O(n^2)$ if $L(v, k)$ is set to $\max\{L(i, k), L(j, k)\}$ (maximal-value), since this enables simple updating of the maximal off-diagonal values as previously described. However, in other cases we may have to sort the entries of each row, therefore adding a factor of $\log(n)$ to the complexity, which becomes $O(n^2 \log(n))$.

Next we present some connection between the input matrix, and trees constructed by the algorithm. Specifically, We show that the ‘maximal-value reduction’ yields a tree whose LCA-matrix is the unique dominant LCA-matrix to the input matrix $D$. Our definitions and analysis are similar to those used in [19, 12] for ultrametrics.
Definition 2.5. An LCA-matrix \( L \) is said to be dominant to a matrix \( D \) if \( D \leq L \) and for every LCA-matrix \( L' \) s.t. \( D \leq L' \leq L \) it follows that \( L = L' \).

It is not hard to see that if \( L_1, L_2 \) are two LCA-matrices dominant to a matrix \( D \), then the matrix \( L \) defined by \( L(i,j) = \min\{L_1(i,j), L_2(i,j)\} \) is also an LCA-matrix dominant to \( D \), which implies that for each matrix \( D \) there is a unique matrix \( L_{\text{dom}} \) dominant to \( D \) (see e.g. [19]). Thus for each LCA-matrix \( L' \geq D \) we have that \( L' \geq L_{\text{dom}} \) as well. Therefore \( L_{\text{dom}} \) is the LCA-matrix closest to \( D \) from above, under any distance-metric \( d \) which satisfies the following intuitive requirement: if \( |D_1 - D| \geq |D_2 - D| \) then \( d(D, D_1) \geq d(D, D_2) \) (this includes, for instance, all the \( \ell_p \) norms).

Next we show how to transform \( L_{\text{dom}} \) into an LCA-matrix closest to \( D \) under the \( \ell_\infty \) norm.

Lemma 2.6. Given a matrix \( D \), and its unique dominant LCA-matrix \( L_{\text{dom}} \), denote by \( \varepsilon = \|D, L_{\text{dom}}\|_\infty = \max_{i,j}\{L_{\text{dom}}(i,j) - D(i,j)\} \). Then \( L_\infty \) defined by \( L_\infty(i,j) = \max\{L_{\text{dom}}(i,j) - \frac{\varepsilon}{2}, 0\} \) is an LCA-matrix closest to \( D \) under the \( \ell_\infty \) norm.

Proof. First, it is rather easy to see that \( L_\infty \) is an LCA-matrix, and that \( \|D, L_\infty\|_\infty = \frac{\varepsilon}{2}. \) Now, given any LCA-matrix \( L \), define by \( \varepsilon_L = \|D, L\|_\infty. \) We need to prove that \( \frac{\varepsilon}{2} \leq \varepsilon_L. \) Denote by \( L' \) the matrix defined as follows: \( L'(i,j) = L(i,j) + \varepsilon_L. \) It is again easy to verify that \( D \leq L' \), and that \( \|D, L'\|_\infty \leq 2\varepsilon_L. \) Now, since \( L' \) is an LCA-matrix, and \( L_{\text{dom}} \) is the dominant LCA-matrix of \( D \), we have \( D \leq L \leq L'. \) This means that \( \varepsilon = \|D, L_{\text{dom}}\|_\infty \leq \|D, L'\|_\infty \leq 2\varepsilon_L. \)

Notice that the transformation in Lemma 2.6 does not change the topology of the tree corresponding to the LCA-matrices (though it may set some edge-weights to zero). A similar transformation (which does not change tre-topology) yields an LCA-matrix closest to \( D \) under the maximal distortion measure, which is the maximal ratio between corresponding distances in the two metrics. This means that for each matrix \( D \) there is a unique tree topology - the one defined by the unique dominant LCA-matrix - which minimizes the various distance-measures to \( D \) described above. Note that edge weights may vary in the different trees according to the specific distance-measure. We conclude this section by proving that this tree is the one constructed by the maximal-value variant of the DLCA algorithm.

Theorem 2.7. Let \( D \) be a symmetric matrix over \( S \), and let \( T \) be the tree over \( S \cup \{v\} \) resulting from executing the DLCA algorithm with maximal-value reduction on \( D \). Then \( LCA_T \) is the unique dominant LCA-matrix of \( D \).

Proof. Denote by \( L = LCA_T \) the output suggested above. Note that \( L \) is an LCA-matrix due to Theorem 2.2. We prove by induction on the dimension of \( D \) \((n)\), that \( L \) is the dominant LCA-matrix of \( D \). If \( n = 1 \), then \( D = [w] \), and \( L = [w] \) as well, and the claim becomes trivial. If \( n > 1 \), then let \( i,j \) be the taxon-pair chosen in \( S \) by the DLCA algorithm. Denote by \( S' = S \setminus \{i,j\} \cup \{v\} \) the reduced taxon-set, by \( D' \) the reduced matrix, and by \( T' \) the tree returned
by the algorithm given $D'$ as input. If $L' = LCA_{T'}$, then by the induction hypothesis, $L'$ is the unique dominant LCA-matrix of $D'$. We will use this to show that $L$ is dominant to $D$.

First, we show that $L \geq D$ using the properties of the reduction:

$$\forall k, l \neq i, j : L(k, l) = L'(k, l) \geq D'(k, l) = D(k, l)$$

$$\forall k \neq i, j : L(k, i) = L(k, j) = L'(k, v) \geq D'(k, v) \geq D(k, i), D(k, j)$$

$$L(i, j) = L'(v, v) \geq D'(v, v) = D(i, j)$$

Now we need to show that if $M$ is an LCA-matrix and $D \leq M \leq L$, then $M = L$. We wish to do this by reducing $M$ to $M'$ (over $S'$) as described in the proof of Theorem 2.2. In order to do this we need to show that $i, j$ satisfy the conditions of Lemma 2.3. Assume this is the case, then from Lemma 2.3 we know that $\forall k \neq i, j : M(i, k) = M(j, k)$, and $M$ can be reduced to $M'$ as previously described. Note that $M'$ is an LCA-matrix and $D' \leq M' \leq L'$. From the induction hypothesis, $L'$ is dominant to $D'$, so $M' = L'$. It is easy to see now that $M = L$ as well:

$$\forall k, l \neq i, j : M(k, l) = M'(k, l) = L'(k, l) = L(k, l)$$

$$\forall k \neq i, j : M(k, i) = M(k, j) = M'(k, v) = L'(k, v) = L'(k, i) = L'(k, j)$$

$$M(i, j) = M'(v, v) = L'(v, v) = L(i, j)$$

We are left to prove that $i, j$ satisfy the conditions of Lemma 2.3, i.e. $\forall k \neq i, j : M(i, k) \geq \max\{M(i, k), M(j, k)\}$. For this end we need to introduce an additional claim to our induction:

$$\forall i : L(i, i) = \max_k \{D(i, k)\} \quad (2.1)$$

We now use this claim (on $L'$) to prove that $L(i, j) = D(i, j)$. Observe that $L(i, j) = L'(v, v) = \max_k \{D'(v, k)\}$. Now, for an arbitrary $k \neq v$, $D'(v, k) = \max\{D(i, k), D(j, k)\}$ due to the maximality of $D(i, j)$. Therefore, since $D'(v, v) = D(i, j)$, we get that $L(i, j) = \max_k \{D'(v, k)\} = D(i, j)$. Using this fact and our assumption-$D \leq M \leq L$, the following holds for every $k \neq i, j$:

$$M(i, k) \leq L(i, k) \leq D(i, j) \leq M(i, j), \quad \text{and similarly} \quad M(j, k) \leq M(i, j).$$

Finally, we prove the additional claim, i.e. $\forall k : L(k, k) = \max_l \{D(k, l)\}$. Notice that it holds for the base case. Now, assume it holds for $L'$, then:

$$\forall k \neq i, j : L(k, k) = L'(k, k) = \max_{l \in S'} \{D'(k, l)\} = \max_{l \in S} \{D(k, l)\}.$$

(The last equality is due to the reduction formula: $D'(k, v) = \max\{D(i, k), D(j, k)\}$)

$$L(i, i) = L(i, j) + w(v, i) = D(i, j) + \max\{0, D(i, i) - D(i, j)\} = \max\{D(i, j), D(i, i)\} = \max_{l \in S} \{D(i, l)\}.$$  

(The last equality is due to the maximality of $D(i, j)$)
The following observation (which will be used later) follows directly from the additional claim (2.1) we now proved for the dominant LCA-matrix:

**Observation 2.8.** If the matrix $D$ satisfies Property 1 of Definition 2.1, namely $\forall i : \max_k \{D(i,k)\} = D(i,i)$, and $L$ is the dominant LCA-matrix of $D$, then $\forall i : L(i,i) = D(i,i)$.

### 2.3 Reconstruction from Pairwise Distances

The DLCA algorithm can be used to reconstruct trees from pairwise distances, by transforming pairwise distances over a set $S$ into estimates of LCA-distances as follows:

**Definition 2.9.** Given a distance matrix $D$ over a set of taxa $S$, and a taxon $r \in S$, $L = \text{LCA}(D, r)$ is the matrix over $S \setminus \{r\}$ defined by $L(i,j) = \frac{1}{2}(D(r,i) + D(r,j) - D(i,j))$.

We can now execute any of the variants of the DLCA algorithm on $L = \text{LCA}(D)$, to get a tree. In such a case we say the algorithm is run on $D$ from taxon $r$. If $D$ is an additive distance matrix consistent with some tree $T$, then $L = \text{LCA}_T$, and it is obvious from previous discussion that the DLCA algorithm returns $T$. However, we are interested in understanding the potential of this method also on inputs which are not additive.

#### 2.3.1 The 3-approximation Algorithm

We show how the above scheme can be used to get an $O(n^2)$ 3-approximation algorithm for the closest additive metric under the $\ell_\infty$ norm. This result was originally presented in [1] using the Farris transform, however our formulation implies simpler analysis.

Given a metric (satisfying the triangle inequality) $D$ over a set $S$ of taxa, our algorithm acts as follows:

1. Choose some arbitrary taxon $r$, and calculate $L = \text{LCA}(D, r)$

2. Find an LCA-matrix $L^\infty$ closest to $L$ under $\ell_\infty$, which satisfies:
   $\forall i \in S, \; L^\infty(i,i) = L(i,i)$.

3. Return the tree $T^\infty$, represented by $L^\infty$.

Stage 2 of this algorithm can be achieved as follows: Since $D$ is a metric, $L$ is nonnegative and $\forall i \in S : D(r,i) = L(i,i) = \max_j L(i,j)$. By Theorem 2.7 and Observation 2.8, when run on $L$, the DLCA algorithm returns $L^{\text{dom}}$ (the dominant LCA-matrix of $L$) which satisfies $\forall i : L^{\text{dom}}(i,i) = L(i,i)$. $L^\infty$ is constructed from $L^{\text{dom}}$ as described in Lemma 2.6, except that $\frac{\varepsilon}{2}$ is subtracted only from the non-diagonal entries of $L^{\text{dom}}$. This gives us that $\forall i : D(r,i) = L(i,i) = L^\infty(i,i) = D_{T^\infty}(r,i)$. 

12
Theorem 2.10. Let $D$ be a metric over a set $S$ of taxa, and let $T^\infty$ be the edge-weighted tree constructed from $L^\infty$ in the above algorithm. Denote by $D_{T^\infty}$ the additive metric implied by $T^\infty$. Then for every additive metric $D'$:

$$||D, D_{T^\infty}||_\infty \leq 3 \cdot ||D, D'||_\infty$$

Proof. The proof consists of two simple claims:

Claim 2.11. $||D, D_{T^\infty}||_\infty = 2 \cdot ||L, L^\infty||_\infty$.

Proof. For an arbitrary taxon-pair $i, j$, we use the formula in Definition 2.9, and the fact that $D(r, i) = D_{T^\infty}(r, i)$ and $D(r, j) = D_{T^\infty}(r, j)$ to show that:

$$D(i, j) - D_{T^\infty}(i, j) = (D(r, i) + D(r, j) - 2L(i, j)) - (D_{T^\infty}(r, i) + D_{T^\infty}(r, j) - 2L^\infty(i, j)) = 2(L^\infty(i, j) - L(i, j))$$

which implies that $||D, D_{T^\infty}||_\infty = 2 \cdot ||L, L^\infty||_\infty$. \hfill $\Box$

Now, denote by $T'$ the edge-weighted tree which realizes $D'$, and let $L' = LCA_{T'}$. Note that $L'$ is an LCA-matrix due to Theorem 2.2.

Claim 2.12. $||L, L'||_\infty \leq \frac{3}{2} \cdot ||D, D'||_\infty$.

Proof. The proof simply follows from the fact that $L'(i, j) = \frac{1}{2}(D'(r, i) + D'(r, j) - D'(i, j))$ and $L(i, j) = \frac{1}{2}(D(r, i) + D(r, j) - D(i, j))$. \hfill $\Box$

Now, since by definition $||L, L^\infty||_\infty \leq ||L, L'||_\infty$, we have that:

$$||D, D_{T^\infty}||_\infty = 2 \cdot ||L, L^\infty||_\infty \leq 2 \cdot ||L, L'||_\infty \leq 3 \cdot ||D, D'||_\infty$$

Notice that this analysis is correct for every root-taxon. In [1] it is shown that in the worst case, there are distance matrices in which the bound of Theorem 2.10 holds with equality for all taxa. However, it is reasonable to suggest that on most real data there is a significant difference in the performance of the algorithm from different root-taxa. We demonstrate this fact in the next chapter.
Chapter 3

Simulations and Results

3.1 The Simulated Trees

When considering the DLCA algorithm until now, we did not discuss the initial step of choosing a root-taxon \( r \) from which to execute the algorithm. As mentioned earlier, when distance estimates are not consistent with some tree (as often is the case), the output may vary for different choices of a root. This choice, therefore, may bare crucial influence on the performance of the algorithm on actual data. In this chapter we explore this influence on simulated data.

We use two types of simulated data: random trees (sampled according to the Yule-Harding distribution) and model-trees. Both datasets were downloaded from the LIRMM ‘Methods and Algorithms in Bioinformatics’ (MAB) website http://www.lirmm.fr/mab/sommaire_english.php3. For model trees we use six trees over 12 taxa, as described in [7]. Three of these trees (A/B group: ‘AA’, ‘AB’ and ‘BB’) are ultrametric, and the other three (C/D group: ‘CC’, ‘CD’ and ‘DD’) are not (See Figure 3.1). These trees are based on smaller model-trees (‘A’, ‘B’, ‘C’ and ‘D’) proposed earlier in [23]. Each tree was scaled in 4 different ways corresponding to different rates of evolution, resulting in a total of 24 model-trees (see [25] for more detail). The ‘Random-trees’ dataset was obtained in [7] as follows: 2\( \times \)2,000 trees (over sets of 24 and 96 taxa) were generated using the stochastic speciation process described in [20], corresponding to the Yule-Harding distribution over trees [14, 11]. Edge-lengths were slightly modified in order to adjust deviation from molecular-clock. Each of the 4,000 tree was rescaled to obtain ‘slow’, ‘moderate’ and ‘fast’ rates of evolution (see details in [7]).

In both cases, SeqGen [24] was used to generate sequences, simulating the process of evolution along each tree, according to the Kimura two parameter model [22] with transition/transversion ratio of 2.0. For each random tree a single simulation was performed resulting in 500\( _b \) long sequences for all taxa. For each of the 24 different model-trees 2\( \times \)1,000 simulations were performed, resulting in 300\( _b \) and 600\( _b \) long sequences. For each such set of sequences we computed a pairwise distance-matrix using DNADIST from the PHYLIP (3.63)
package [17]. We obtained estimates on LCA-distances from these distance matrices using the transformation stated in Definition 2.9. Note that, as suggested in [21, 25], we could have obtained more accurate estimates directly over taxon-triplets. However, our tests indicate that the difference in estimates obtained by both approaches on our datasets is not significant.

For each distance matrix we hold the original (correct) tree from which it is originated, in order to test the accuracy of reconstruction. Accuracy is measured by the Robinson-Foulds (RF) distance [9] between the original and reconstructed topologies. Since they are both fully resolved, half the RF-distance is the number of edges of the original tree not appearing in the reconstructed topology. We term this number the RF-score. The RF-score of a reconstruction algorithm on a dataset is the average RF-score of all reconstructed topologies. Note that the smaller the RF-score, the better the algorithm is.

3.2 Results on random trees

When executing the DLCA algorithm on an \( n \times n \) distance-matrix, \( n \) different trees can potentially be obtained. We first wish to see how good is the best tree obtained, how bad is the worst tree, and how the average-case behaves. Both variants of DLCA (mid-point and maximum-value) were tested and compared to Saitou&Nei’s NJ algorithm for reference. Summary of these results appear in Table 3.1 and Figure 3.2. It is notable that the ‘mid-point’ variant consistently outperforms ‘max-value’. From this point on we will concentrate on results obtained by the mid-point variant of DLCA. In Table 3.1 we can see that change in the scale of the trees (rate of evolution) has little influence on the relative performance of the algorithms. They all seem to perform better when the rate of evolution is faster. This behavior is also observed in [7].

The relative performance of the average-case root is not influenced dramatically by the size of the tree (number of taxa). In both 24-taxa and 96-taxa trees the RF-score of the average root (using mid-point reduction) is about 25% higher than the RF-score of NJ. On the other hand, the relative performance of the best-case root significantly deteriorates in bigger trees. In 24-taxa trees the RF-score of the best root (using mid-point reduction) is about 50% lower than the RF-score of NJ, and in 96-taxa trees it is merely 15% lower.

This observation can also be seen in Figure 3.2. We ranked the taxa in each data instance according to the RF-scores that DLCA achieves when using them as roots: rank = 1 for the taxon resulting in the lowest score, and rank = #taxa for taxon resulting in the highest score. In Figure 3.2 we see average RF-scores plotted against root-rank of the taxa. The results shown here correspond to trees with moderate rate of evolution, however very similar behavior can be seen for other rates as-well. We observe that in both cases the best topology obtained by DLCA (‘mid-point’ variant) is on average better than the one obtained by NJ. However, in 96-taxa trees a smaller portion of the taxa (8/96) yield lower average scores than NJ, compared with 24-taxa trees (8/24).
Table 3.1: **Average performance on random trees.** Results for NJ and DLCA are estimated using the RF-score. Each result is averaged over 2,000 random instances. DLCA was executed using both mid-point and max-value reductions. For each variant we show the score obtained by the best root, worst root, and average score over all roots. The upper table corresponds to random trees over 24 taxa, and the lower one to trees over 96 taxa.

### Results on model trees

It would be very helpful to know, prior to execution, from which taxa the DLCA algorithm is likely to yield a more accurate reconstruction. This could save us the trouble of running it from all taxa, and improve the chance for more accurate reconstruction. In order to approach this challenge, we used the ‘model trees’ dataset since it contains multiple (1,000) simulations for each tree. In each tree we looked for taxa which consistently lead to better reconstruction. Figure 3.3 summarizes the results achieved for all trees under the fastest rate of evolution (M.D = 2.0, see [25]), with 300b long sequences. Other settings yielded similar results.

Each taxon receives an RF-score, which is the average RF-score over all topologies reconstructed using DLCA from that taxon. It is apparent from our results that in ultrametric trees (A/B group) all taxa receive similar average RF-scores, which are notably higher than the score NJ receives. In such trees there does not seem to be a taxon which significantly outperforms the others. In the other group of trees (C/D), which are far from ultrametric, we see high fluctuations in scores. Some taxa significantly outperform others. In all three trees there are taxa which receive average RF-scores similar to NJ, whereas the
Table 3.2: Choosing a root-taxon. Results for NJ and DLCA (mid-point variant) are estimated by the RF-score. For DLCA we show the average score over all roots, and the score received by the root $i$ minimizing the criterion: $\max_{j \in S} \{D(i, j)\}$.

<table>
<thead>
<tr>
<th></th>
<th>AA</th>
<th>BB</th>
<th>AB</th>
<th>CC</th>
<th>DD</th>
<th>CD</th>
</tr>
</thead>
<tbody>
<tr>
<td>NJ</td>
<td>2.924</td>
<td>3.534</td>
<td>3.247</td>
<td>2.106</td>
<td>2.089</td>
<td>2.103</td>
</tr>
<tr>
<td>chosen taxon</td>
<td>3.688</td>
<td>4.420</td>
<td>4.000</td>
<td>2.219</td>
<td>2.181</td>
<td>2.262</td>
</tr>
</tbody>
</table>

rest receive much higher scores. It is easy to see that the taxa which yield more accurate reconstruction are the ones closer to the actual root of the tree. In ‘CD’, for instance, we see that taxa 1, 3, 4, 7, 9, 11 receive scores similar to NJ ($\sim 2$), taxa 2, 5, 8, 10 receive a much higher score ($\sim 4.4$), and taxa 6, 12 receive a somewhat better score ($\sim 3.8$). This correlates well with distances from the root of the tree: $\sim 4b$ for the first group, $\sim 13b$ for the second, and $9b$ for the last. Notice that taxon 3 receives a slightly higher score than 4. This difference, though small, seems to be significant, and it cannot be explained by distance from the root (as taxa 3,4 are equidistant from the root). There may be other factors which come into play here.

Given an input distance-matrix, therefore, we wish to guess which taxa are closer to the root of evolution. In a general model of evolution this may be impossible. However, in the model trees we studied, the taxa which are closer to the root are relatively closer to other taxa as-well. A taxon $i$ minimizing $\max_{j \in S} \{D(i, j)\}$ is likely to be close to the root. In Table 3.2 we can see that choosing a taxon via this criterion yields better results than choosing a taxon by random. In the A/B group this difference is rather small, however in the C/D group the improvement is much more significant. We tried using the same criterion to find ‘good taxa’ in random trees, however we did not reach a significant improvement from the average-case. Note, however, that in some scenarios of phylogenetic reconstruction, we may add an outgroup taxon, which we know to be relatively close to the root. In such a case we believe DLCA will perform best from that outgroup.

3.4 Reconstruction accuracy vs. best-fit to input

In case we do not know which taxa are more likely to yield a better tree, we can run DLCA from all taxa, and choose one of the $n$ resulting trees. However, since we do not hold the original topology, we cannot know which of them is more accurate. One approach is to test the fit of a tree to the input matrix. Hopefully, the better it fits the input, the closer it is to the original topology. We tested this assumption on our random trees over 24 taxa, by computing the $\ell_2$-distances from each input matrix to all trees reconstructed from it. Since we
### Table 3.3: Reconstruction accuracy vs. best-fit to input under $\ell_2$. The top row shows the average relative improvement in RF-scores when choosing the tree closest to the input matrix, compared with the average case (as shown in Table 3.1). The rest of the table shows in what ratio of the data DLCA achieves better fit to the input-matrix, compared with NJ, and the original tree. NJ is compared to both variants of DLCA (‘maximal value’ and ‘mid-point’). All results were obtained on the random-trees dataset over 24 taxa.

<table>
<thead>
<tr>
<th></th>
<th>Slow</th>
<th>Moderate</th>
<th>Fast</th>
</tr>
</thead>
<tbody>
<tr>
<td>Improvement from average</td>
<td>6.72%</td>
<td>10.28%</td>
<td>12.72%</td>
</tr>
<tr>
<td>DLCA beats NJ (MID) (MAX)</td>
<td>93%</td>
<td>94%</td>
<td>93%</td>
</tr>
<tr>
<td>DLCA beats original tree</td>
<td>92%</td>
<td>90%</td>
<td>91%</td>
</tr>
</tbody>
</table>

Wish to test the reconstructed topology, we disregard the edge-weights assigned during reconstruction, and use least-squares estimates of edge weights instead (as suggested in [27]). These weights minimize, for a given topology, the $\ell_2$-distance from the input matrix. A similar approach was taken by Farach and Cohen in [5] regarding $\ell_1$ and $\ell_\infty$ (using Linear Programming to re-calculate edge weights). The results are presented in the upper row of Table 3.1. We see that the average RF-score of the topology ‘closest’ to the input matrix is much closer the one received by the average-case root, than the one received by the best topology. The improvement in RF-score, compared with the average case, is no more than 13%. Therefore, among the topologies reconstructed by DLCA, that which fit best the input matrix, is rarely a good candidate for approximating the original topology.

We also checked to see if DLCA yields a better $\ell_2$-fit to the input matrix, compared with NJ. Indeed, for more than 90% of the distance-matrices in our dataset, the best tree reconstructed by DLCA is closer to the input matrix than the one reconstructed by NJ. Interestingly, more than 90% of the times this tree is even closer to the input than the original tree. This supports the observation we made earlier, that best-fit to the input matrix rarely implies the most accurate reconstruction. Farach and Cohen show by simulations [5] that their pivotal-algorithm (SP-DP) yields a better fit to the input matrix than NJ under $\ell_1, \ell_\infty$. Note that the SP algorithm is equivalent to the ‘maximal-value’ variant of DLCA, and DP is a variant of SP which allows for midpoints between two taxa to act as roots, as well as the taxa themselves. We see here that ‘maximal-value’ also yields good results considering $\ell_2$. However, it seems the ‘mid-point’ variant is slightly better than it. This fact adds up with our initial observation that the ‘mid-point’ variant is superior to ‘maximal-value’ regarding the RF-score, despite the good theoretical properties we proved for the latter in Sections 2.2, 2.3.
Figure 3.1: Model trees. Short branches are of uniform length ($a$ for the A/B group and $b$ for the C/D group). Lengths of long branches are specified. Note that trees in the A/B group are ultrametric, and trees in C/D are not. Taxa of trees in the C/D group are labelled by 1-12.
Figure 3.2: **Performance from various roots on random trees.** The horizontal line corresponds to the score obtained by NJ. The x-axis corresponds to root-rank. rank=1 means that we chose the root yielding the smallest RF-score, and rank=\#taxa means we chose the worst-case root. Graphs describe performance on 24-taxon and 96-taxon random trees (both with moderate rate evolution).
Figure 3.3: **Characterizing good roots.** Summary of results obtained for all 6 model trees. In each graph the first 12 bars correspond to all 12 taxa. The last bar corresponds to NJ. Each taxon receives an RF-score, which is the average RF-score over all topologies reconstructed using DLCA from that taxon.
Chapter 4

Discussion and Conclusion

In this paper we presented a characterization of edge-weighted trees using LCA-distances. LCA-distances of trees obey a 3-point condition dual to the 3-point ultrametric condition. This duality enables us to provide a simple neighbor-joining criterion (deepest least common ancestor). Using this criterion we defined a family of efficient neighbor joining algorithms (DLCA), with $O(n^2 \log(n))$ running time. These algorithms are pivotal in the sense that when the input is not consistent with some tree, the output may vary according to the choice of root-taxon (from which LCA-distances are estimated).

We studied two variants of this family, denoted by ‘mid-point’ and ‘maximal-value’. The ‘maximal-value’ variant was shown to have some desirable theoretical properties; it can be implemented in $O(n^2)$ time, and yields a topology best-fitting the estimated LCA-distances under several interesting measures of fitness (see discussion following Lemma 2.6). Using these properties, this variant was shown to provide a new simple $O(n^2)$ 3-approximation algorithm for the closest additive metric under the $\ell_\infty$ norm (originally shown in [1]).

We used simulated data to test the performance of both variants of DLCA against Saitou&Nei’s NJ algorithm. Using the fact that the DLCA algorithm is pivotal, we obtained $n$ potentially different trees from each data-instance. We see that the pivotal approach almost always yields topologies better fitting the input matrix, than NJ. This is shown here (Table 3.3) regarding the $\ell_2$ norm, and was shown earlier in [5] regarding $\ell_1, \ell_\infty$ norms.

However, the main goal of our experiments was to test the accuracy of topological reconstruction. In this aspect we observe that the ‘mid-point’ variant is significantly superior to ‘maximal-value’. As in the case of approximating the input matrices, in most cases at least one of the $n$ resulting trees is closer to the original topology than the one returned by NJ.

A natural problem raised by this scenario is that since in real data we do not know the original topology, there is no obvious way to select an output tree which best approximates it. This calls for some heuristic solution, which is able to pinpoint either a topology is close to the original one, or a root-taxon which is likely to yield such a topology. A possible solution is to use some other
optimization criterion (like maximum parsimony of the underlying sequences). Our results indicate that fitness to the input distance matrix (specifically under $\ell_2$), which a-priori could have served as such a criterion, does not seem to imply similarity to the original topology, and hence may be inappropriate for the specified evolutionary model.

On the other hand, we were able to illustrate on several model trees that taxa closer to the origin of evolution are more likely to yield better topologies. This observation justifies introducing an outgroup-taxon close to that origin. The approach of selecting roots as midpoints between pairs of taxa, which is used in the DP algorithm introduced in [5], can be useful in exploring this direction as well.
Bibliography


