

Fast and Reliable Reconstruction of Phylogenetic Trees with Very Short Edges*

(Draft)

Ilan Gronau[†] Shlomo Moran[‡] Sagi Snir[§]

January 17, 2008

Abstract

Phylogenetic reconstruction methods deal with the problem of reconstructing a tree describing the evolution of a given set of species; this is usually done using sequences of characters (e.g. DNA) extracted from these species. A central goal in the design of these methods is to be able to guarantee accurate reconstruction (with high probability) from short input sequences. Under common models of evolution, the required sequence length is known to depend on the depth of the tree as well as on its minimal edge-weight. Fast converging reconstruction algorithms are considered state-of-the-art in this context, as they require asymptotically minimal sequence length in order to accurately reconstruct the entire tree. However, when the tree in question contains very short edges, the sequence length required for complete (and accurate) topological resolution may be too long for practical purposes. This calls for *adaptive* fast converging algorithms studied in this paper, which, given input sequences of any length, correctly reconstruct as many edges of the evolutionary tree as possible.

In this paper we present a fast converging reconstruction algorithm which returns a partially resolved topology containing no false edges and *all* “sufficiently long” edges of the underlying phylogenetic tree. The weight of edges our algorithm guarantees to reconstruct is determined by the input sequence length and the depth of the tree; it does not depend, however, on the minimal edge-weight of the tree, and in this aspect it is strictly stronger than any previously known edge-reconstruction guarantee. This fact, together with the optimal complexity of our algorithm (linear space and quadratic-time), makes it appealing for practical use.

1 Introduction.

Phylogenetic reconstruction is the task of figuring out the evolutionary history of a given set of extant species (taxa). This history is usually described by an edge-weighted tree whose internal vertices represent past speciation events (extinct species) and whose leaves correspond to the given set of taxa. The (positive) weight of an edge in this tree describes the amount of evolutionary change between the two speciation events it connects. Reconstruction methods typically receive as input an alignment of sequences, each corresponding to a different taxon, and they are expected to yield a tree which closely depicts the true phylogenetic tree. In this work we only consider the *topology* of the output tree, which is essentially its unrooted and unweighted description (disregarding all zero-weight edges). The output topology is said to be *fully resolved* if the degree of each internal vertex is exactly three. We compare the output topology with the original phylogenetic tree by comparing

*A preliminary version of this paper appeared in the proceedings of SODA08

[†]Technion - Israel Institute of Technology, Haifa, 32000Israel

[‡]Technion - Israel Institute of Technology, Haifa, 32000Israel

[§]Netanya Academic College, Netanya, Israel

the splits induced by their edges on the set of taxa. In other words, an edge of the original tree is said to be correctly reconstructed if the output tree contains an edge inducing the same split.

A major challenge in designing reconstruction algorithms is being able to guarantee a maximum amount of correct reconstruction from short alignments. Studies providing such reconstruction guarantees usually assume some (relatively simple) model of sequence evolution, such as the Cavender-Farris-Neyman (CFN) model for binary sequences [5, 15, 25], or the Jukes-Cantor model for DNA sequences [19]. Many results in this field focus on determining the minimal sequence length required for correct reconstruction of the entire phylogenetic tree. These results show that this sequence-length depends on the weight of the shortest edge in the tree and on some notion of tree-depth. It was already shown in [13] that correct reconstruction can be guaranteed (w.h.p.) from sequences of length:

$$k = O\left(\frac{\log(n)}{f^2} \cdot \exp(\text{depth})\right), \quad (1)$$

where n is the number of taxa, f is the minimal edge weight in the tree, and depth is the depth of the tree (see Definition 6.5). In [22, 24] it is shown that this is the optimal dependence (up to constants) in f and depth for general trees, and in [12] it is shown that the (exponential) dependence on depth can be eliminated from §1, when assuming an additional *specific* upper bound on edge weights. So for general trees, the algorithm suggested in [13] requires asymptotically optimal sequence length. Note that if g is an upper bound on edge weight in T , then $\text{depth} \leq g \log_2(n)$, and §1 reduces to the following polynomial dependence in n :

$$k = \frac{n^{O(g)}}{f^2}. \quad (2)$$

Much attention has been focused in recent years on designing other algorithms which provide a similar guarantee to that of [13] (see e.g. [14, 18, 7, 8, 20, 22, 23, 11]). Such reconstruction methods are often said to be *fast converging* (they are called *absolutely fast converging* if they do not require a prior knowledge of the lower and upper bounds on edge lengths [21]). Whereas fast converging algorithms minimize the sequence length required for correct reconstruction of any given tree, most of them do not attempt to maximize the amount of correct reconstruction when the input sequences are too short to ensure correct reconstruction of the entire tree. In fact, the great majority of fast converging algorithms fail to produce any meaningful output in such a case. This poses an obvious problem when trying to apply such algorithms on actual (and even simulated) data. The forest reconstruction algorithms of [23, 11] are the only fast converging algorithms known today which attempt to cope with this problem. When the sequences are too short for complete reconstruction, these algorithms return a forest of fully resolved edge-disjoint trees which are consistent (w.h.p) with the original tree. Whereas this approach allows correct reconstruction of the ‘shallow’ edges in the tree, the presence of some short edges may prevent such algorithms from reconstructing long edges situated deeper in the tree, as demonstrated by the following example.

Consider a full binary tree over $4n$ leaves obtained by taking n 4-leaf trees with an arbitrarily short internal edge, and attaching them (in the middle of the internal edge) to n leaves of a full binary tree whose edges all have some (moderate) fixed weight (see Fig. 1). The algorithms in [23, 11] are based on the disk-covering approach [18], in which the tree is reconstructed from sub-phylogenies spanning r -disks of taxa; the r -disk of x consists of all taxa which are at distance at most r from x . In our example, any non-trivial sub-phylogeny induced by an r -disk of some taxon x must contain a short edge. Now assume that the input sequences are too short to ensure reliable resolution of these short edges. Then the methods in [23, 11], and in fact any method which returns (w.h.p.) a forest of correct *fully resolved trees*, will return only trees which span at most three taxa, and hence will not resolve *any* of the internal edges of such a tree.

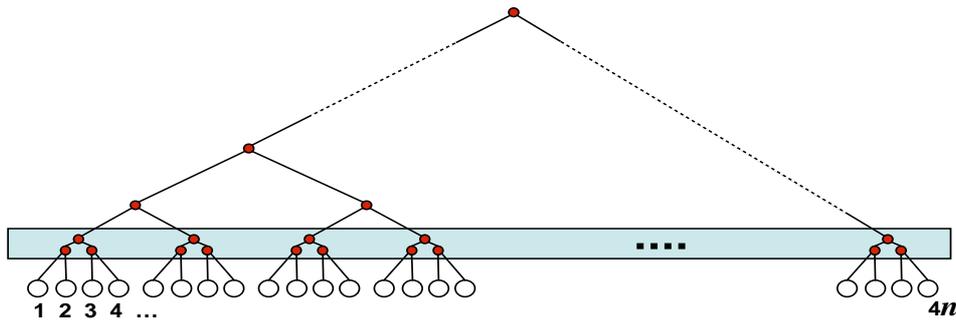


Figure 1: **Forest Reconstruction.** Consider the above full binary tree and assume that the edges in the highlighted layer are too short to be reconstructed by the input sequences. Then any disk-covering method which returns a forest of fully resolved trees will not be able to reconstruct any of the internal edges (even the longer ones).

The above example demonstrates the sensitivity of fast converging algorithms to the existence of short edges in the tree. This motivates the following extension of fast convergence:

Definition 1.1. A reconstruction algorithm is said to be adaptive fast converging if it accurately reconstruct all edges of weight greater than ε (w.h.p.), from sequences of length

$$k = \frac{n^{O(g)}}{\varepsilon^2} . \quad (3)$$

In this paper we present a technique which enables reconstruction of adaptive fast converging algorithms by contracting low-confidence edges (and thus introducing high degree vertices). This technique guarantees that: (1) the returned topology contains no false positives (i.e. edges inducing incorrect splits), and (2) the missing edges correspond only to “short” edges of the original tree. After introducing the general technique, we show how it can be applied on pairwise dissimilarity data (extracted from the input sequences) to obtain an adaptive fast converging algorithm. Contraction of low-confidence edges is an approach which exists in other known reconstruction methods, such as Buneman’s classic algorithm and its refinements (see eg [4, 2]). However, our technique results in a more efficient algorithm, and it is adaptive fast converging, and thus allows the reconstruction of much shorter edges than ones reconstructed by Buneman’s algorithm.

The algorithm presented in this paper follows the incremental approach which was introduced in [3], and later applied to fast converging algorithms in [9, 8, 20]. In this approach, the topology is constructed by attaching the taxa one by one. The goal in each iteration is to find the correct place of insertion of the new taxon in the current topology. This is typically done by querying the vertices of the current topology as to the direction (relative to that specific vertex) in which the new taxon should be attached. When a vertex in the current topology corresponds to a vertex of the original tree, such directional queries can be simply answered by querying quartet splits (see [9, 8]). Our directional queries may fail to return a direction when it is not confident enough in the answer. In such a case, the introduction of false edges is avoided by contracting some of the previously constructed edges. As a result, the topology maintained by our incremental algorithm may contain vertices which correspond to (contracted) subtrees of the original tree. This significantly complicates the task of reliably answering a directional query, but we manage to accomplish this task within the asymptotic optimal time complexity of $O(n^2)$.

The performance of our basic algorithm depends on a bound on the diameter of the contracted subtrees, which in extreme cases can be very large. In the final section of this paper we present a method which avoids this dependence, and thus imply the adaptive fast convergence of the algorithm.

In the next section we present the notations used in the paper. In Sections 3-6 we present our basic incremental algorithm in a ‘top-down’ fashion: In Section 3 we outline our algorithm and present the concept of *partial directional oracle* which is a model-independent primitive for constructing phylogenetic trees in the presence of noisy information. In Section 4 we prove some basic properties of our algorithm. In Section 5 we present an implementation of a directional oracle which uses a reliable *quartet oracle*. In Section 6 we present and analyze an implementation of our algorithm from a noisy version of the additive metric induced by the original tree. In Section 7 we analyze the time and space complexities of our algorithm, and in Section 8 we discuss the performance of our algorithm on a common model of sequence evolution. In the final section of this paper we present a modification of the basic algorithm (as presented in sections 3-6), which achieves adaptive fast convergence.

2 Definitions and Notations.

Trees: A tree T is an undirected connected and acyclic graph. $V(T)$ and $E(T)$ denote the sets of vertices and edges of T , respectively. $leaves(T)$ denotes the leaf-set of T , and $internal(T) = V(T) \setminus leaves(T)$ denotes the set of its internal vertices. For a vertex $v \in V(T)$, the *neighborhood* of v , $N_T(v)$, is the set of vertices adjacent to v in T . The neighborhood of a subset $U \subseteq V$ is defined by $N_T(U) = \cup_{u \in U} N(u) \setminus U$. The degree of a vertex, $deg_T(v)$, is the size of its neighborhood in T . The *parent* of a leaf x in T , $parent_T(x)$, is the unique vertex in $N_T(x)$. T is said to be a *phylogenetic tree* over taxon-set S if $leaves(T) = S$, the degree of every internal vertex is at least three, and every edge $e \in E(T)$ is associated with a strictly positive weight $w(e)$. The weight function w induces a metric $D_T = \{d_T(u, v)\}_{u, v \in V(T)}$ over $V(T)$, s.t. $d_T(u, v)$ is the total weight of $path_T(u, v)$ – the (unique) simple path connecting u and v in T . The diameter of T ($diam(T)$) is the maximum weight of a simple path in T .

A *subtree* of a tree T is a connected subgraph of T . The notion of distances is generalized for subtrees as follows: for two vertex disjoint subtrees t_1, t_2 of T , $d_T(t_1, t_2)$ denotes the total weight of $path_T(t_1, t_2)$, which is the (unique) shortest path in T connecting a vertex in t_1 and a vertex in t_2 . Let t_1, t_2, t_3 be mutually disjoint subtrees of T . We say that t_2 *separates* t_1 from t_3 if $path_T(t_1, t_3)$ intersects t_2 . If t_2 does not separate t_1 from t_3 we say that t_1 and t_3 are *on the same side of* t_2 (see Fig. 2). In general, we use lower case t 's to denote subtrees of a tree T , and whenever the tree T is clear from context, the subscript T may be removed from the corresponding notation.

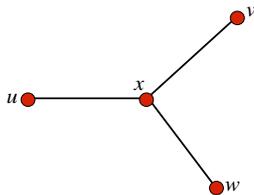


Figure 2: In the above tree, x separates u from v and from w . v , x and w are all on the same side of u .

Induced sub-phylogenies: Let T be a phylogenetic tree over a set of taxa S , and let $S' \subseteq S$. $T(S')$, the phylogenetic tree induced by T on S' , is obtained by taking the minimal subtree of T which spans S' , and removing all vertices of degree two by iteratively replacing the two edges which touch such a vertex with a single edge. Note that every vertex in $V(T(S'))$ corresponds to a vertex of T and every edge in $E(T(S'))$ corresponds to a simple path in T (see Fig. 3). Edge-weights in $T(S')$ are determined according to the weight of corresponding paths in T . Therefore, $T(S')$ induces a sub-metric to the metric D_T induced by T , and D_T is used to describe distances induced by all sub phylogenies of T .

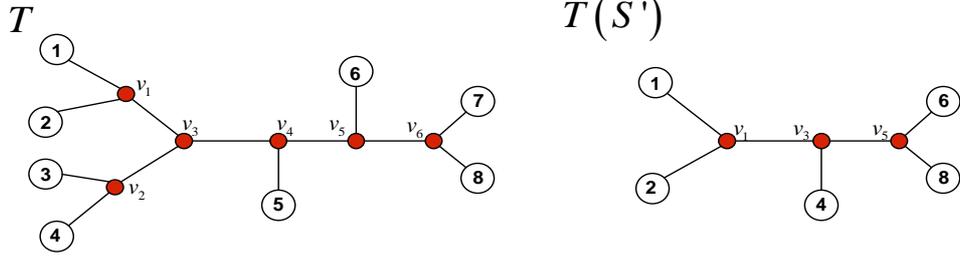


Figure 3: **Induced Sub-phylogeny.** Left: A tree over the taxon-set $S = \{1, \dots, 8\}$. Right: The sub phylogeny induced by $S' = \{1, 2, 4, 6, 8\}$. Notice that the edge (v_3, v_5) in $T(S')$ corresponds to the path $(v_3 - v_4 - v_5)$ in T .

Internal Edge Contraction: The contraction of an edge $e = (u, u') \in E(T)$ replaces e with a single vertex v s.t. $N(v) = N(\{u, u'\})$. In such a case we say that edge e was contracted into vertex v . \hat{T} is said to be an *internal (edge) contraction* of T if it is obtained from T by a series of contractions of internal edges (see Fig. 4). Note that if \hat{T} is an internal contraction of T then $leaves(T) = leaves(\hat{T})$. All edge contractions considered in this paper are internal. \hat{T} is said to be an ε -contraction of T if each (internal) edge in T which is contracted in \hat{T} has weight at most ε . An internal contraction of T to \hat{T} induces a mapping between vertices of $internal(\hat{T})$ and vertex disjoint subtrees of $internal(T)$: each vertex \hat{v} of $internal(\hat{T})$ corresponds either to a vertex in $internal(T)$, or to a subtree consisting of the internal edges contracted into \hat{v} . The subtree of T corresponding to a vertex \hat{v} of \hat{T} is denoted by $t_{\hat{v}}$. Edge weights in a contraction \hat{T} of T are determined by the weight of the corresponding edges in T . This means that for neighboring vertices \hat{u}, \hat{v} in \hat{T} , we have $d_{\hat{T}}(\hat{u}, \hat{v}) = d_T(t_{\hat{u}}, t_{\hat{v}})$, but otherwise, $d_T(t_{\hat{u}}, t_{\hat{v}})$ is generally larger than $d_{\hat{T}}(\hat{u}, \hat{v})$ (see Fig. 4). We complete this section with two simple observations which are used later (often implicitly):

Lemma 2.1. *Let t be a subtree of T s.t. $V(t) \subseteq internal(T)$. Then the tree \hat{T} obtained from T by replacing t with a single vertex \hat{v} such that $N_{\hat{T}}(\hat{v}) = N_T(V(t))$ is an edge contraction of T . If in addition the weights of all edges in t are at most ε , then \hat{T} is an ε -contraction of T .*

Lemma 2.2. *[transitivity of ε -contraction] Let T_1, T_2, T_3 be three phylogenetic trees over S . If T_1 is an ε -contraction of T_2 and T_2 is an ε -contraction of T_3 , then T_1 is an ε -contraction of T_3 .*

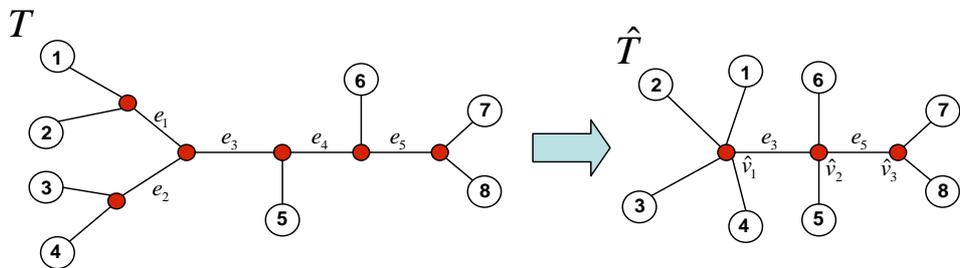


Figure 4: **Internal Edge Contraction.** The tree \hat{T} is obtained from T by contracting e_1, e_2 into \hat{v}_1 and e_4 into \hat{v}_2 . Note that $d_{\hat{T}}(\hat{v}_1, \hat{v}_2) = w(e_3) = d_T(t_{\hat{v}_1}, t_{\hat{v}_2})$, whereas $d_{\hat{T}}(\hat{v}_1, \hat{v}_3) = w(e_3) + w(e_5)$ and $d_T(t_{\hat{v}_1}, t_{\hat{v}_3}) = w(e_3) + w(e_4) + w(e_5)$.

3 Basics of the Incremental Algorithm.

Our incremental reconstruction algorithm works in iterations, where the objective in each iteration is to extend an edge contraction of $T(S')$ for some $S' \subseteq S$ to an edge-contraction of $T(S' \cup \{x\})$ for some $x \in S \setminus S'$.

Procedure *Incremental_Reconstruct*(S):

- Select $x_0, x_1 \in S$, and initialize: $\hat{T} \leftarrow (x_0, x_1)$; $S' \leftarrow \{x_0, x_1\}$.
- While $S' \neq S$ do:
 1. Select a taxon $x \in S \setminus S'$ and set $S' \leftarrow S' \cup \{x\}$.
 2. *Attach_Taxon*(\hat{T}, x)

The crucial part of this process is pinpointing the *anchor* of x , which is the location of $parent(x)$ in the current topology, as defined below.

Definition 3.1 (Anchor). *Let \hat{T} be an edge contraction of $T(S')$, and let $x \in S \setminus S'$. The anchor of x in \hat{T} , $anchor_{\hat{T}}(x)$, is defined as follows: Let p_x be the parent of x in $T(S' \cup \{x\})$. If p_x is included in $t_{\hat{v}}$ for some $\hat{v} \in V(\hat{T})$ (either by being a vertex in $t_{\hat{v}}$ or by lying on a path in T which corresponds to an edge of $t_{\hat{v}}$), then $anchor_{\hat{T}}(x) = \hat{v}$. Otherwise, $anchor_{\hat{T}}(x)$ is the unique edge $(\hat{u}, \hat{v}) \in E(\hat{T})$ for which p_x is an internal vertex in $path_T(t_{\hat{u}}, t_{\hat{v}})$.*

The anchor of x in \hat{T} is found by querying vertices in \hat{T} for the location of x . These queries are posed to a *directional oracle* which receives the new taxon x and a vertex \hat{v} of \hat{T} and is expected to output the neighbor \hat{u} of \hat{v} which indicates the direction of x with respect to \hat{v} (if such a neighbor exists).

Definition 3.2 (Partial Directional Oracle). *Let T be a phylogenetic tree over S . A partial directional oracle for T is a function $PDO = PDO_T$ which receives queries of the form (\hat{T}, \hat{v}, x) , where*

- \hat{T} is an edge-contraction of $T(S')$, for some $S' \subset S$,
- $\hat{v} \in V(\hat{T})$,
- $x \in S \setminus S'$,

and outputs either a vertex $\hat{u} \in N_{\hat{T}}(\hat{v})$ or ‘null’. The output must satisfy two requirements:

- If $\hat{v} \in leaves(\hat{T})$, then $PDO(\hat{T}, \hat{v}, x) = parent_{\hat{T}}(\hat{v})$.
- If $PDO(\hat{T}, \hat{v}, x) = \hat{u}$, then $t_{\hat{u}}$ and x are on the same side of $t_{\hat{v}}$ in $T(S' \cup \{x\})$ (this property is also referred to as the *truthfulness* of the oracle). We sometimes abuse notation and simply say that \hat{u} is on the same side of \hat{v} as x .

Our algorithm uses the partial directional oracle to compute the *insertion zone* of x in \hat{T} .

Definition 3.3 (Insertion Zone). *Given an edge-contraction \hat{T} of $T(S')$, some taxon $x \in S \setminus S'$, and a partial directional oracle PDO , the insertion zone of x in \hat{T} , denoted by $\hat{t}_{inz}(\hat{T}, x, PDO)$ (or simply \hat{t}_{inz} when \hat{T}, x and PDO are clear from context), is the subtree of \hat{T} defined by the following procedure:*

Procedure *Find_Insertion_Zone*(\hat{T}, x, PDO):

1. $\hat{t}_{inz} \leftarrow \hat{T}$
2. For every edge $(\hat{u}, \hat{v}) \in E(\hat{t}_{inz})$ s.t. $PDO(\hat{T}, \hat{v}, x) = \hat{u}$ do:
 - Delete from \hat{t}_{inz} all the vertices which are separated from \hat{u} by \hat{v} .

A simple inductive argument using the truthfulness of PDO implies that the above procedure returns a subtree of \widehat{T} which includes $anchor_{\widehat{T}}(x)$ as below (see Fig. 5):

Observation 3.4. $\hat{t}_{inz} = \hat{t}_{inz}(\widehat{T}, x, PDO)$ is a subtree of \widehat{T} which satisfies the following:

1. $anchor_{\widehat{T}}(x)$ is either an internal vertex or an edge (internal or external) of \hat{t}_{inz} .
2. For each leaf \hat{v} of \hat{t}_{inz} , $PDO(\widehat{T}, \hat{v}, x) = parent_{\hat{t}_{inz}}(\hat{v})$.
3. For each internal vertex \hat{v} of \hat{t}_{inz} (if any), $PDO(\widehat{T}, \hat{v}, x) = \text{'null'}$.

We conclude this general overview by describing how to attach x to \widehat{T} given its insertion zone.

Procedure $Attach_Taxon(\widehat{T}, x)$:

1. $\hat{t}_{inz} \leftarrow \hat{t}_{inz}(\widehat{T}, x, PDO)$.
2. If \hat{t}_{inz} is a single edge (\hat{u}, \hat{v}) , then attach x to \widehat{T} by introducing a new internal vertex \hat{p}_x and replacing (\hat{u}, \hat{v}) with the three edges (\hat{u}, \hat{p}_x) , (\hat{v}, \hat{p}_x) , (x, \hat{p}_x) .
3. If \hat{t}_{inz} has a single internal vertex \hat{v} (i.e. $V(\hat{t}_{inz}) = \{\hat{v}\} \cup N(\hat{v})$), then add to \widehat{T} the edge (\hat{v}, x) .
4. Else (i.e. \hat{t}_{inz} has at least one internal edge), contract all internal edges of \hat{t}_{inz} into a new vertex \hat{v} and add to \widehat{T} the edge (\hat{v}, x) .

There are two types of edge contraction which may occur during an execution of $Attach_Taxon(\widehat{T}, x)$, which inserts a taxon $x \in S \setminus S'$ to the current topology \widehat{T} over S' . The first are *explicit* contractions, which are implied by contractions of internal edges of \hat{t}_{inz} (including the case that such an internal edge is the anchor of x). In each iteration however at most one *implicit* contraction may occur as well, when the anchor of x is an *external* edge (\hat{u}, \hat{v}) of \hat{t}_{inz} (eg when v is a leaf of \hat{t}_{inz} - see Fig. 6). All the contracted edges (due to explicit or implicit contractions) appear in an intermediate topology \widehat{T}^{+x} , which is the following natural extension of \widehat{T} to a contraction of $T(S' \cup \{x\})$:

- If the anchor of x in \widehat{T} is a vertex $\hat{v} \in V(\widehat{T})$ then \widehat{T}^{+x} is obtained by adding an edge (\hat{v}, x) to \widehat{T} .
- Otherwise, the anchor of x in \widehat{T} is an edge $(\hat{u}, \hat{v}) \in E(\widehat{T})$, and \widehat{T}^{+x} is obtained by replacing the edge (\hat{u}, \hat{v}) with the three edges (\hat{u}, \hat{p}_x) , (\hat{p}_x, \hat{v}) , (\hat{p}_x, x) .

In the second case, if \hat{v} is a leaf of \hat{t}_{inz} , then (\hat{u}, \hat{v}) is not contracted but (\hat{u}, \hat{p}_x) is contracted. This is illustrated in Fig. 6, which describes the edges contracted in \widehat{T}^{+x} in the scenario given in Fig. 5b.

We conclude this section by a definition and an observation which imply the correctness of our insertion procedure.

Definition 3.5. Let \widehat{t} be an (internal edge) contraction of $T(S')$ and $x \in S \setminus S'$. Let further \tilde{t} be a subtree of \widehat{T} . We say that $anchor_{\widehat{T}}(x)$ touches \tilde{t} if it is either a vertex in $V(\tilde{t})$, or it is an edge with at least one endpoint in $V(\tilde{t})$.

The following observation follows directly from the definition of internal edge contraction.

Observation 3.6. Let \widehat{T} , x and \tilde{t} be as in Definition 3.5. Then $anchor_{\widehat{T}}(x)$ touches $\tilde{t} \iff$ the tree \widehat{T}_{post} obtained from \widehat{T} by contracting \tilde{t} to a new vertex \hat{v} and adding an edge (\hat{v}, x) is a contraction of $T(S' \cup \{x\})$.

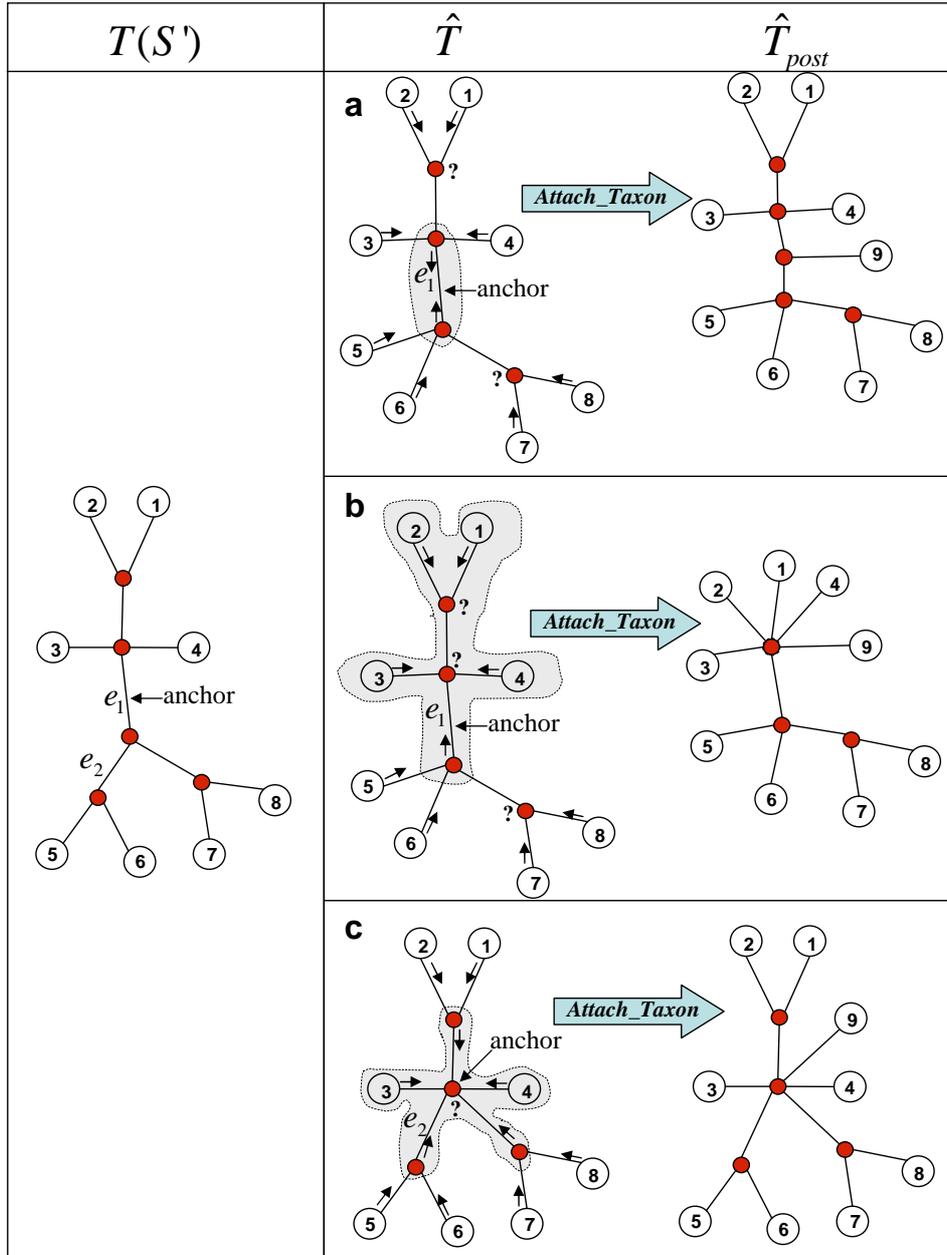


Figure 5: This figure illustrates a single insertion iteration of the algorithm. In this example we have $S' = \{1, \dots, 8\}$ and $x = 9$. Left: the induced sub-phylogeny $T(S')$, in which the anchor of x (edge e_1) is indicated. Right: three scenarios for the insertion of x into \hat{T} . The left tree depicts the current topology \hat{T} before insertion, with answers to directional queries and the insertion zone \hat{t}_{inz} marked by a grey cloud. The right tree depicts the topology \hat{T}_{post} after the insertion. **(a,b)** The current topology \hat{T} is obtained from $T(S')$ by contracting e_2 , so the anchor of x in \hat{T} is e_1 . **(a)** \hat{t}_{inz} consists a single edge, so \hat{T}_{post} is obtained from \hat{T} by splitting this edge to two and connecting x to the new internal vertex resulting from this. **(b)** \hat{t}_{inz} contains one internal edge, which is contracted into a new internal vertex to which x is connected. **(c)** The current topology \hat{T} is obtained from $T(S')$ by contracting e_1 , so the anchor of x in \hat{T} is the vertex into which e_1 is contracted. \hat{t}_{inz} contains a single internal vertex to which x is connected.

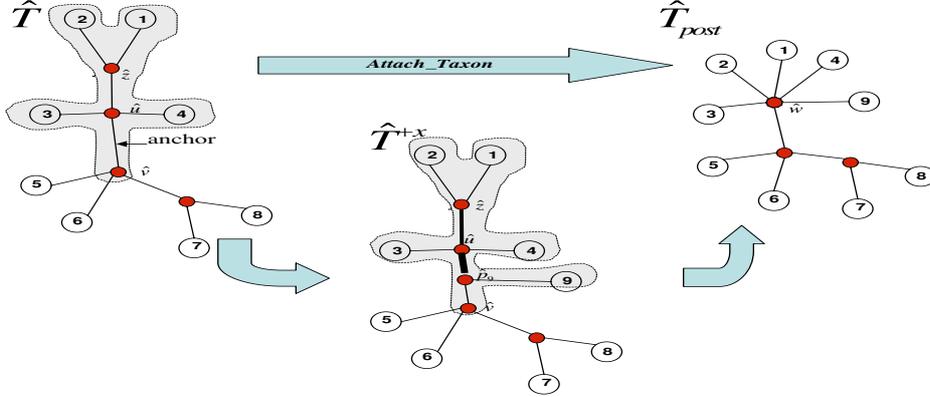


Figure 6: **explicit and implicit contractions** The insertion of taxon $x = 9$ into \widehat{T} as presented in Fig. 5b results in contraction of two edges in \widehat{T}^{+x} into the vertex \hat{w} in \widehat{T}_{post} : An explicit contraction of (\hat{u}, \hat{z}) (medium bold) as internal edge of the ε -environment of x , and an implicit contraction of (\hat{u}, \hat{p}_9) (bold), which occurs since the anchor of x is the external edge (u, v) of \hat{t}_{inz} .

Lemma 3.7. *Let \widehat{T} be a contraction of $T(S')$ and let $x \in S \setminus S'$. Then the tree \widehat{T}_{post} obtained by $Attach_Taxon(\widehat{T}, x)$ is a contraction of $T(S' \cup \{x\})$.*

Proof. The lemma holds trivially if \widehat{T}_{post} is obtained by step 2 of the procedure. It holds also when \widehat{T}_{post} is obtained by step 3 or 4 due to Observation 3.6 and the fact that, by Observation 3.4(1), $anchor_{\widehat{T}}(x)$ touches the subtree spanned by $internal(\hat{t}_{inz})$. \square

4 ε -Reliable Directional Oracles.

We now formalize the properties of the partial directional oracle which guarantee that all contracted edges are of weight smaller than ε .

Definition 4.1 (ε -environment). $\hat{t}_{env}(\widehat{T}, x, \varepsilon)$, the ε -environment of x in \widehat{T} , is the maximal subtree of \widehat{T}^{+x} which includes x and whose internal edges have weight at most ε .

Definition 4.2 (ε -reliability). A partial directional oracle PDO is said to be ε -reliable for (\widehat{T}, x) , if the insertion zone of x determined by PDO is included in the ε -environment of x , i.e.: $V(\hat{t}_{inz}) \subseteq V(\hat{t}_{env}(\widehat{T}, x, \varepsilon))$. Specifically, any internal vertex of \hat{t}_{inz} is also an internal vertex of $\hat{t}_{env}(\widehat{T}, x, \varepsilon)$.

Lemma 4.3. *If the partial directional oracle PDO used in the calculation of \hat{t}_{inz} is ε -reliable for (\widehat{T}, x) , then \widehat{T}_{post} is an ε -contraction of $T(S' \cup \{x\})$.*

Proof. By Lemma 3.7, \widehat{T}_{post} is a contraction of $T(S' \cup \{x\})$. Hence, by the assumption that \widehat{T} is an ε -contraction of $T(S')$, we only need to prove that all the edges contracted during the current iteration are of weight smaller than ε . Consider such a contracted edge (\hat{u}, \hat{v}) . Then each endpoint of e is either an internal vertex of \hat{t}_{inz} and hence, by Definition 4.2 also

of $\hat{t}_{env}(\hat{T}, x, \varepsilon)$, or it is p_x , the parent of x , which by definition is also an internal vertex of $\hat{t}_{env}(\hat{T}, x, \varepsilon)$. Thus e is an internal edge of $\hat{t}_{env}(\hat{T}, x, \varepsilon)$, and hence its weight is smaller than ε . \square

Lemma 4.3 provides the main inductive argument used in the proof of our main result (Theorem 6.9). It is important to note that the validity of this argument requires that our partial directional oracle is ε -reliable in every iteration of the algorithm.

5 An Efficient Implementation of the Partial Directional Oracle Using Quartet Queries.

In this section we present our partial directional oracle and give explicit conditions on \hat{T} and x under which it is ε -reliable. Our directional oracle *PDO* uses queries on *quartet splits*. A taxon-quartet $q = \{a, b, c, d\} \subseteq S$ defines an induced sub-topology $T(q)$ which is either a star or a split $(x, y|z, w)$, where $\{x, y, z, w\} = q$, and $T(q)$ has a single internal edge (of positive weight) separating $\{x, y\}$ from $\{z, w\}$. Reconstructing phylogenetic trees from quartet splits dates back as far as [4]. In order to allow quartet queries over noisy data, we introduce the concept of a *partial quartet oracle*, which may fail to return a split when it is not confident in the result (even when the topology of the quartet is not a star).

Definition 5.1 (Partial Quartet Oracle). *Let T be a phylogenetic tree over S . A partial quartet oracle for T is a function which receives a quartet $q \subseteq S$ and returns either the (correct) split of q in T , or ‘null’.*

Recall that a partial directional oracle, when queried on a vertex \hat{v} and taxon x , returns either ‘null’ or a neighbor \hat{u} of \hat{v} which is on the same side of \hat{v} as x . The procedure described below seeks such a neighbor via a series of queries to a partial quartet oracle *PQO*. These quartet queries include the new taxon x and three other taxa s_1, s_2, s_3 of \hat{T} which represent three different directions corresponding to three different neighbors of \hat{v} in \hat{T} , defined as follows:

Definition 5.2 (Directional Representatives). *Let (\hat{u}, \hat{v}) be an edge in \hat{T} . A taxon s of \hat{T} is a directional representative of $(\hat{v} \rightarrow \hat{u})$ if $\hat{u} \in \text{path}_{\hat{T}}(\hat{v}, s)$. The directional representative of $(\hat{v} \rightarrow \hat{u})$ is denoted by $s_{\hat{v}}(\hat{u})$.*

Our partial directional oracle contains two main phases:

Triplets Tournament: In this phase, the set of all possible directions (represented by $N_{\hat{T}}(\hat{v})$) is iteratively screened to end up with at most one candidate direction. In each iteration a quartet is queried and as a result at least two directions are eliminated from the set of candidates. If the tournament results in an empty candidate set, then the directional oracle returns ‘null’. Otherwise, the tournament results in a single surviving candidate. The following validation phase is needed in order to guarantee that the surviving candidate (if exists) indeed indicates the correct direction.

Validation: Validation of the direction represented by the surviving neighbor \hat{u} is done by another series of quartet queries which contain both x and $s_{\hat{v}}(\hat{u})$. If all quartet queries positively validate this direction (meaning that they put x and $s_{\hat{v}}(\hat{u})$ on the same side of the split), then \hat{u} is returned. Otherwise, the directional oracle returns ‘null’.

Claim 5.3. *The time required for the execution of $PDO(\hat{T}, \hat{v}, x)$ is linear in $\text{deg}_{\hat{T}}(\hat{v})$.*

Proof. Each iteration of the triplets tournament phase takes $O(1)$ time, and at least two neighbors of \hat{v} are eliminated from the candidate set at each iteration. So this phase ends

after at most $\frac{1}{2}deg_{\widehat{T}}(\hat{v})$ iterations. The validation phase simply scans the neighborhood of \hat{v} , and so it ends after at most $O(deg_{\widehat{T}}(\hat{v}))$ time. \square

Lemma 5.4. *If PQO is a partial quartet oracle for T and all directional representatives satisfy Definition 5.2, then procedure PDO described above is a (truthful) partial directional oracle for T .*

Proof. Consider a valid input instance $(\widehat{T}, \hat{v}, x)$ for PDO. If \hat{v} is a taxon, then PDO returns the unique neighbor of \hat{v} in \widehat{T} , as required. So assume \hat{v} is an internal vertex of \widehat{T} . It is sufficient to show that for any vertex $\hat{u} \in N_{\widehat{T}}(\hat{v})$ which is not on the same side of \hat{v} as x , there exists a vertex which fails \hat{u} at step 5b of the validation phase. If there is a vertex $\hat{u}' \in N(\hat{v})$ which is on the same side of \hat{v} as x , then \hat{u}' fails the validation of \hat{u} when chosen either as \hat{u}_2 or as \hat{u}_1 (see Fig. 7a). If there is no such vertex, then \hat{v} is the anchor of x in \widehat{T} , and p_x – the parent of x in $T(leaves(\widehat{T}) \cup \{x\})$ – is contained in $t_{\hat{v}}$. In this case, for each choice of \hat{u}_1 there exists $\hat{u}_2 \in N_{\widehat{T}}(\hat{v}) \setminus \{\hat{u}, \hat{u}_1\}$ s.t. $path_T(p_x, t_{\hat{u}})$ separates $t_{\hat{u}_1}$ from $t_{\hat{u}_2}$ (see Fig. 7b). This means that the quartet $\{x, s_{\hat{v}}(\hat{u}), s_{\hat{v}}(\hat{u}_1), s_{\hat{v}}(\hat{u}_2)\}$ is not split by $(x, s_{\hat{v}}(\hat{u}) | s_{\hat{v}}(\hat{u}_1), s_{\hat{v}}(\hat{u}_2))$, and hence the validation of \hat{u} fails. \square

The Partial Directional Oracle – PDO(\widehat{T}, \hat{v}, x):

1. Initialize candidate set $C \leftarrow N_{\widehat{T}}(\hat{v})$.
2. If $C = \{\hat{u}\}$ (\hat{v} is a taxon), return \hat{u} .
3. Otherwise ($|C| \geq 3$), proceed as follows:
4. **Triplets Tournament:**
 - While $|C| > 1$ do:
 - If $|C| = 2$, then $C \leftarrow C \cup \{\hat{u}\}$, for some $\hat{u} \in N_{\widehat{T}}(\hat{v}) \setminus C$.
 - Select some triplet $\{\hat{u}_1, \hat{u}_2, \hat{u}_3\} \subseteq C$ and invoke $PQO(\{x, s_{\hat{v}}(\hat{u}_1), s_{\hat{v}}(\hat{u}_2), s_{\hat{v}}(\hat{u}_3)\})$.
 - If output is ‘null’, then remove $\hat{u}_1, \hat{u}_2, \hat{u}_3$ from C .
 - Otherwise, the output is $(x, s_{\hat{v}}(\hat{u}_i) | s_{\hat{v}}(\hat{u}_j), s_{\hat{v}}(\hat{u}_k))$ (where $\{i, j, k\} = \{1, 2, 3\}$), then remove \hat{u}_j, \hat{u}_k from C .
 - If the tournament results in $C = \emptyset$, return ‘null’.
5. **Validation:** $C = \{\hat{u}\}$ for some $\hat{u} \in N_{\widehat{T}}(\hat{v})$.
 - (a) Select some vertex $\hat{u}_1 \in N_{\widehat{T}}(\hat{v}) \setminus \{\hat{u}\}$.
 - (b) For every $\hat{u}_2 \in N_{\widehat{T}}(\hat{v}) \setminus \{\hat{u}, \hat{u}_1\}$, invoke $PQO(\{x, s_{\hat{v}}(\hat{u}), s_{\hat{v}}(\hat{u}_1), s_{\hat{v}}(\hat{u}_2)\})$.
 - If output is $(x, s_{\hat{v}}(\hat{u}) | s_{\hat{v}}(\hat{u}_1), s_{\hat{v}}(\hat{u}_2))$, then continue.
 - Otherwise, stop and return ‘null’.
 - (c) Return \hat{u} (if it survived all rounds).

The truthfulness of PDO is not enough: in order to get meaningful results, we need in addition that in some cases it will return a valid direction (not ‘null’). For this, we need to classify the cases where our partial quartet oracle guarantees to return a valid split. Under most common models of evolution, the reliability of quartet splits inferred from estimated distances increases when the diameter of the quartet decreases and the length of the internal edge increases [13, 23, 11]. This motivates the definition of a (r, ε) -reliable quartet oracle below¹.

¹In Section 6 we present an (r, ε) -reliable distance-based quartet oracle. A somewhat similar oracle is

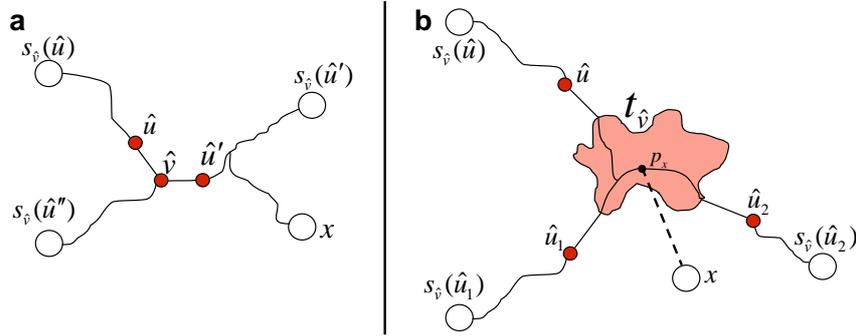


Figure 7: **Proof of Lemma 5.4.**

Definition 5.5 ((r, ε) -Reliable Quartet Oracle). A partial quartet oracle is said to be (r, ε) -reliable if it returns the split of the input quartet q whenever $\text{diam}(T(q)) \leq r$ and the single internal edge of $T(q)$ has weight greater than ε .

The following lemma specifies the cases in which PDO is guaranteed to return the correct direction. These cases depend on the (r, ε) -reliability of PQO .

Lemma 5.6. Let \hat{T} be a contraction of $T(S')$, and $\{s_{\hat{v}}(\hat{u}') : \hat{u}' \in N(\hat{v})\}$ be directional representatives (see Definition 5.2). Assume further that the following holds for a vertex $\hat{v} \in \text{internal}(\hat{T})$ and taxon $x \in S \setminus S'$:

1. \hat{v} has a neighbor \hat{u} in \hat{T} which is on the same side of \hat{v} as x .
2. PQO is an (r, ε) -reliable quartet oracle for T , where r, ε satisfy the following:
 - (a) $\varepsilon < \min\{d_T(t_{\hat{v}}, t_{\hat{u}}), d_T(t_{\hat{v}}, p_x)\}$, where p_x is the parent of x in $T(S' \cup \{x\})$.
 - (b) $r \geq \max\{d_T(y, z)\}$, for all taxon-pairs $\{y, z\} \subset \{x\} \cup \{s_{\hat{v}}(\hat{u}') : \hat{u}' \in N(\hat{v})\}$.

Then $PDO(\hat{T}, \hat{v}, x) = \hat{u}$.

Proof. Consider all taxon-quartets queried by PDO of type $q = \{x, s, s_1, s_2\}$, where s is the directional representative of $(\hat{v} \rightarrow \hat{u})$ in \hat{T} . By condition 2b we have that $\text{diam}(T(q)) \leq r$. It is easy to see that the path in T corresponding to the internal edge of $T(q)$ intersects $t_{\hat{v}}$, and in addition it either contains p_x (see Fig. 8a) or intersects $t_{\hat{u}}$ (see Fig. 8b). Hence, by condition 2a above, the weight of this path is greater than ε . Therefore, by the (r, ε) -reliability of PQO we get that $PQO(q) = (x, s \mid s_1, s_2)$, which implies that \hat{u} survives all rounds of the triplets tournament and the validation phase. \square

Corollary 5.7. Let \hat{T} be an internal contraction of $T(S')$ for some $S' \subset S$, and let x be a taxon in $S \setminus S'$. Assume that for every $\hat{v} \in \text{leaves}(\hat{t}_{\text{env}}(\hat{T}, x, \varepsilon))$ we have:

1. $d_T(t_{\hat{v}}, x) \leq r_1$.
2. $\text{diam}(t_{\hat{v}}) \leq r_2$.
3. $\forall \hat{u} \in N_{\hat{T}}(\hat{v}) : d_T(t_{\hat{v}}, s_{\hat{v}}(\hat{u})) \leq r_3$.

Then if PQO is (r, ε) -reliable for $r \geq r_2 + r_3 + \max\{r_1, r_3\}$, PDO is ε -reliable for (\hat{T}, x) .

also implicit in [11].

Proof. In order to establish the ε -reliability of PDO , we must show that the insertion zone of x in \widehat{T} is contained in the ε -environment of x in \widehat{T} . By Observation 3.4(1,3), all we need to prove is that $PDO(\widehat{T}, \hat{v}, x) \neq \text{'null'}$ for every leaf \hat{v} of $\hat{t}_{env} = \hat{t}_{env}(\widehat{T}, x, \varepsilon)$. This clearly holds if \hat{v} is a taxon. Thus it suffices to show that each leaf \hat{v} of \hat{t}_{env} which is an internal vertex of \widehat{T} satisfies conditions 1, 2a and 2b of Lemma 5.6.

The definitions of r, r_1, r_2, r_3 imply condition 2b of Lemma 5.6. By the definition of ε -environment, condition 1 is satisfied by \hat{u} , the unique neighbor of \hat{v} in \hat{t}_{env} . Let p_x be the parent of x in $T(S' \cup \{x\})$. Since \hat{v} is an internal vertex of \widehat{T} which is a leaf of \hat{t}_{env} , the weight of the external edge of \hat{t}_{env} adjacent to \hat{v} is larger than ε . If (\hat{v}, \hat{u}) is the anchor of x in \widehat{T} , then this edge is (\hat{v}, p_x) and $d_T(t_{\hat{v}}, t_{\hat{u}}) \geq d_T(t_{\hat{v}}, p_x) > \varepsilon$; otherwise this edge is (\hat{v}, \hat{u}) and $d_T(t_{\hat{v}}, p_x) \geq d_T(t_{\hat{v}}, t_{\hat{u}}) > \varepsilon$. Thus condition 2a of Lemma 5.6 is satisfied as well. \square

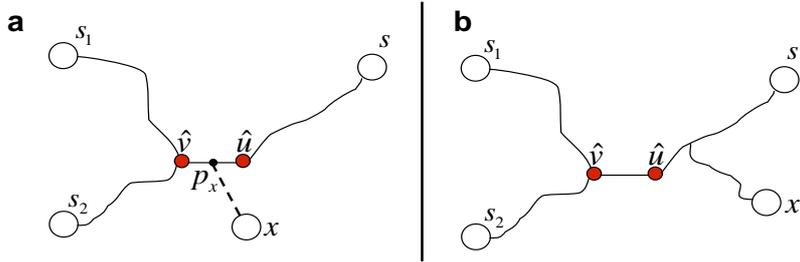


Figure 8: **Proof of Lemma 5.6.** (a) If the edge (\hat{v}, \hat{u}) is the anchor of x in \widehat{T} , then the weight of the internal edge of $T(q)$ is at least $d_T(t_{\hat{v}}, p_x)$. (b) Otherwise, its weight is at least $w(\hat{v}, \hat{u}) = d_T(t_{\hat{v}}, t_{\hat{u}})$.

Note: Corollary 5.7 assumes a bound (r_2) on the diameter of contracted subtrees. In Section 9 we present a variant of our algorithm which is independent on this bound. In Section 6 we discuss this issue in more details.

6 Applying the Algorithm on Noisy Tree Metrics.

In this section we specify conditions which guarantee that our partial directional oracle is ε -reliable in every iteration of the incremental algorithm. As commonly done in phylogenetic reconstruction, we assume that the input to the algorithm is a *dissimilarity matrix* \widehat{D} over S ($\widehat{D} = \{\hat{d}(i, j)\}_{i, j \in S}$), which is a noisy version of the metric D_T induced by the phylogenetic tree T . We model this noise by an arbitrary non-decreasing function of the pairwise distances, as follows:

Definition 6.1. Two dissimilarity matrices D_1, D_2 over S are said to be α -close, for some function $\alpha : \mathbb{R}^+ \rightarrow \mathbb{R}^+ \cup \{\infty\}$, if for every taxon-pair $\{i, j\} \subseteq S$, we have

$$|d_1(i, j) - d_2(i, j)| \leq \alpha(\min\{d_1(i, j), d_2(i, j)\}) .$$

Given a noise function α , let us denote $B_\alpha^+(d) = d + \alpha(d)$ and $B_\alpha^-(d) = d - \alpha(d)$. The following basic observation is used extensively in our analysis.

Observation 6.2. Assume that D_1 and D_2 are α -close dissimilarity matrices. Then:

- $B_\alpha^-(d_1(i, j)) \leq d_2(i, j) \leq B_\alpha^+(d_1(i, j))$,
- $B_\alpha^-(d_2(i, j)) \leq d_1(i, j) \leq B_\alpha^+(d_2(i, j))$.

For our analysis we assume that the input matrix \widehat{D} and the tree-metric D_T are α -close for some efficiently computable non-decreasing function α . Our partial quartet oracle $FPM_{\widehat{D},\alpha}(q)$ is the following modified version of the well-known four-point method (FPM) (see e.g. [13]).

The Partial Quartet Oracle – $FPM_{\widehat{D},\alpha}(q)$:
 Let $q = \{a, b, c, d\}$, and assume a labelling of the four taxa which satisfies:

$$\hat{d}(a, b) + \hat{d}(c, d) \leq \min\{\hat{d}(a, c) + \hat{d}(b, d), \hat{d}(a, d) + \hat{d}(b, c)\} .$$

 – Return $(a, b|c, d)$, if:

$$B_{\alpha}^{-}(\hat{d}(a, c)) + B_{\alpha}^{-}(\hat{d}(b, d)) > B_{\alpha}^{+}(\hat{d}(a, b)) + B_{\alpha}^{+}(\hat{d}(c, d)) \quad (4)$$

 – Otherwise, return ‘null’.

Lemma 6.3. *Assume that \widehat{D} is α -close to D_T . Then for every $r \in \mathbb{R}^+$, $FPM_{\widehat{D},\alpha}$ is an (r, ε_r) -reliable partial quartet oracle for T , where $\varepsilon_r = 4\alpha(B_{\alpha}^{+}(r))$.*

Proof. First we show that $FPM_{\widehat{D},\alpha}$ is a truthful quartet oracle, meaning that if the inequality in §4 holds, then $(a, b|c, d)$ is the correct quartet split in T . This is established by showing that $d_T(a, b) + d_T(c, d) < d_T(a, c) + d_T(b, d)$.

$$\begin{aligned} d_T(a, b) + d_T(c, d) &\leq B_{\alpha}^{+}(\hat{d}(a, b)) + B_{\alpha}^{+}(\hat{d}(c, d)) \\ \text{(by §4)} \quad &< B_{\alpha}^{-}(\hat{d}(a, c)) + B_{\alpha}^{-}(\hat{d}(b, d)) \leq d_T(a, c) + d_T(b, d) . \end{aligned}$$

We are left to show that if (a) $T(q) = (a, b|c, d)$, (b) $\text{diam}(T(q)) \leq r$, and (c) the weight of the single internal edge in $T(q)$ is larger than $\varepsilon_r = 4\alpha(B_{\alpha}^{+}(r))$, then the quartet split $(a, b|c, d)$ is returned. Hence we have to prove that (a),(b),(c) imply §4. Notice that (b) implies that $\hat{d}(i, j) \leq B_{\alpha}^{+}(r)$ for all $\{i, j\} \subset q$. Using the monotonicity of α and Observation 6.2 we get:

$$\begin{aligned} B_{\alpha}^{-}(\hat{d}(a, c)) + B_{\alpha}^{-}(\hat{d}(b, d)) &\geq \hat{d}(a, c) + \hat{d}(b, d) - 2\alpha(B_{\alpha}^{+}(r)) \geq \\ d_T(a, c) + d_T(b, d) - 4\alpha(B_{\alpha}^{+}(r)) &> \quad \text{(by (a),(c) above)} \\ d_T(a, b) + d_T(c, d) + 4\alpha(B_{\alpha}^{+}(r)) &\geq \hat{d}(a, b) + \hat{d}(c, d) + 2\alpha(B_{\alpha}^{+}(r)) \geq \\ B_{\alpha}^{+}(\hat{d}(a, b)) + B_{\alpha}^{+}(\hat{d}(c, d)). & \quad \square \end{aligned}$$

Lemma 6.3 establishes the (well known) fact, that in order to establish (r, ε) -reliability for small values of ε , we will have to make sure that the diameters of the quartets queried by our algorithm are kept as small as possible. For this we select the inserted taxon to be as close as possible to the current topology, and the directional representatives as close as possible to the corresponding edges, as specified below.

Order of Insertion: The algorithm starts by setting $S' \leftarrow \{x_0, x_1\}$ where x_0, x_1 are the closest taxa in \widehat{D} , and inserting them to \widehat{T} . In each consequent iteration it selects a taxon $x \in S \setminus S'$ closest to S' (where $\hat{d}(S', x) = \min_{y \in S'} \{\hat{d}(x, y)\}$).

Updating Directional Representatives: The algorithm holds two representatives $s_{\hat{u}}(\hat{u}), s_{\hat{v}}(\hat{v})$ for each edge (\hat{u}, \hat{v}) in \widehat{T} . Consider the updates required after inserting taxon x into \widehat{T} . Let y denote the taxon closest to x in \widehat{T} (under \widehat{D}), and let \hat{p}_x denote the parent of x in \widehat{T} . The following updates take place for the new external edge (x, \hat{p}_x) : $s_{\hat{p}_x}(x) \leftarrow x$ and $s_x(\hat{p}_x) \leftarrow y$. If an edge (\hat{u}, \hat{v}) is split to $(\hat{u}, \hat{p}_x), (\hat{p}_x, \hat{v})$, the following updates take place: $s_{\hat{v}}(\hat{p}_x), s_{\hat{p}_x}(\hat{u}) \leftarrow s_{\hat{v}}(\hat{u})$, and $s_{\hat{u}}(\hat{p}_x), s_{\hat{p}_x}(\hat{v}) \leftarrow s_{\hat{u}}(\hat{v})$. Finally, if contractions (of the internal edges of \hat{t}_{inz}) take place, then edges touching the new vertex (resulting from contraction) inherit the directional representatives of the corresponding external edges of \hat{t}_{inz} .

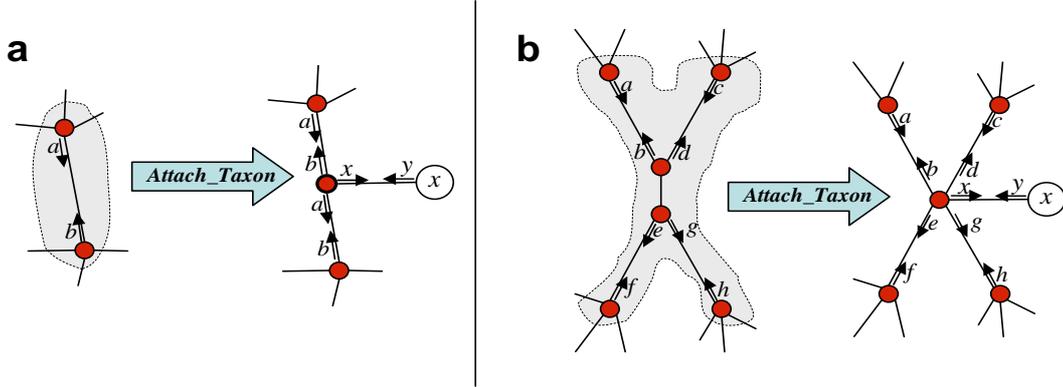


Figure 9: **Updating Directional Representatives.** (a) An edge is split during insertion of x . Its directional representatives (a, b in the figure above) are copied to the two resulting edges. (b) An edge is contracted during insertion of x . Directional representatives of external edges of the insertion zone (marked by a grey cloud) are copied to the edges touching the new internal vertex (into which the internal edge is contracted).

The bounds on the diameters of the queried quartets depend on two important properties of the input tree T : its ε -diameter and its depth, which are defined next.

Definition 6.4. For $\varepsilon \geq 0$, the ε -diameter of T (denoted by $\text{diam}(T, \varepsilon)$) is the maximum weight of a simple path in T consisting only of edges of weight at most ε .

Definition 6.5. The depth of a tree T (denoted by $\text{depth}(T)$) is given by:

$$\text{depth}(T) = \max_{v \in V(T), u \in N_T(v)} \{ \min\{d_T(v, s) : s \in \text{leaves}(T), u \in \text{path}_T(v, s)\} \}.$$

Lemma 6.6. Let $\emptyset \subset S' \subset S$. Then there is a taxon-pair $y \in S', x \in S \setminus S'$ s.t. $d_T(x, y) \leq 2\text{depth}(T)$.

Proof. Let T' be the subtree of T which spans S' (note that T' may have vertices of degree two). A vertex of T' is said to be full if $\text{deg}_{T'}(v) = \text{deg}_T(v)$. Observe that all leaves of T' are full, and that since $S' \neq S$, there exists a non-full vertex in T' . Now root T (and T') at some arbitrary vertex r , and let $v \in V(T')$ be a non-full (internal) vertex which maximizes $d_T(r, v)$. Let u be an arbitrary child of v in T' . Then, by the maximality of $d_T(r, v)$, u and all its descendants in T' are full. Hence there is a taxon $y \in S'$ which is a descendant of u in T' and $d_T(v, y) \leq \text{depth}(T)$. Also, since v is not full, it has a child $u' \in V(T) \setminus V(T')$ s.t. u' and all its descendants in T are not in T' , implying similarly that u' has a descendant taxon $x \in S \setminus S'$ s.t. $d_T(v, x) \leq \text{depth}(T)$. Thus $d_T(x, y) = d_T(x, v) + d_T(v, y) \leq 2\text{depth}(T)$. \square

In the analysis below, we use $B_\alpha^{+2}(d)$ to denote the function $B_\alpha^+(B_\alpha^+(d))$. We will use the following observation, which follows from the monotonicity of α :

Observation 6.7. $B_\alpha^{+2}(d) - d = \alpha(d) + \alpha(B_\alpha^+(d))$ is a non decreasing function of d .

Lemma 6.8. Assume that \widehat{D} is a dissimilarity matrix over S which is α -close to the tree-metric D_T , and let $S' \subset S$. Let $x \in S \setminus S'$ be a taxon closest to S' under \widehat{D} , and let $y \in S'$ be the taxon closest to x in S' . Finally, let p_x denote the parent of x in $T(S' \cup \{x\})$. Then,

1. $d_T(x, y) \leq B_\alpha^{+2}(2\text{depth}(T))$.
2. $d_T(x, p_x) \leq B_\alpha^{+2}(2\text{depth}(T)) - \text{depth}(T)$.

Proof. Throughout the proof we extensively apply Observation 6.2 and use $depth$ in short to denote $depth(T)$. By Lemma 6.6 there exist $x' \in S \setminus S'$ and $y' \in S'$ s.t. $d_T(x', y') \leq 2depth$. For such x', y' we have $\hat{d}(x', y') \leq B_\alpha^+(d_T(x', y')) \leq B_\alpha^+(2depth)$. Hence, since $\hat{d}(x, y) \leq \hat{d}(x', y')$, we get: $d_T(x, y) \leq B_\alpha^+(\hat{d}(x, y)) \leq B_\alpha^+(\hat{d}(x', y')) \leq B_\alpha^{+2}(2depth)$.

The second inequality is proved similarly. By the definition of $depth$ there must be a taxon $x' \in S \setminus S'$ s.t. $d_T(p_x, x') \leq depth$. Now notice that $d_T(x, p_x) - d_T(x', p_x) = d_T(x, y) - d_T(x', y)$, thus

$$d_T(x, p_x) = d_T(x', p_x) + [d_T(x, y) - d_T(x', y)] \leq depth + [d_T(x, y) - d_T(x', y)].$$

So it suffices to show that $d_T(x, y) \leq d_T(x', y) + B_\alpha^{+2}(2depth) - 2depth$. If $d_T(x', y) \geq 2depth$, then this is implied by the fact that $d_T(x, y) \leq B_\alpha^{+2}(2depth)$. So assume that $d_T(x', y) < 2depth$. Then we get:

$$\begin{aligned} d_T(x, y) &\leq B_\alpha^+(\hat{d}(x, y)) &\leq B_\alpha^+(\hat{d}(x', y)) &\leq B_\alpha^{+2}(d_T(x', y)) \\ \text{(by Observation 6.7 above)} &&& \leq d_T(x', y) + B_\alpha^{+2}(2depth) - 2depth. \end{aligned} \quad \square$$

We are now ready to present the main result of this section:

Theorem 6.9. *Consider a phylogenetic tree T over a taxon-set S . Let \hat{D} be a dissimilarity matrix which is α -close to the tree-induced metric D_T , for some non-decreasing function α . Assume that the following holds for some $\varepsilon \in \mathbb{R}^+$:*

$$\varepsilon \geq 4\alpha(B_\alpha^+(r)) \quad \text{for } r = 3B_\alpha^{+2}(2depth(T)) - 2depth(T) + 2diam(T, \varepsilon). \quad (5)$$

Then, algorithm Incremental_Reconstruct (Section 3) which uses the quartet oracle $FPM_{\hat{D}, \alpha}$ returns a topology which is an ε -contraction of T .

Sketch of Proof. We use $depth$ in short to denote $depth(T)$, and H_ε to denote $diam(T, \varepsilon)$. By Lemma 6.3, §5 implies that $FPM_{\hat{D}, \alpha}$ is an (r, ε) -reliable quartet oracle for $r = 3B_\alpha^{+2}(2depth) - 2depth + 2H_\varepsilon$. The proof is completed by proving (inductively) that the topology \hat{T} our algorithm holds throughout its execution satisfies the following properties:

1. \hat{T} is a ε -contraction of $T(S')$.
2. Every edge in \hat{T} has weight at most $B_\alpha^{+2}(2depth) - depth$.
3. For every directional representative $s_{\hat{v}}(\hat{u})$ in \hat{T} , we have $d(t_{\hat{v}}, s_{\hat{v}}(\hat{u})) \leq B_\alpha^{+2}(2depth)$.

The induction is pretty straightforward. Lemma 6.8 is used to prove conditions 2 and 3. Condition 1 is proved with Lemma 4.3 and Corollary 5.7, using the induction hypothesis to obtain the following bounds on r_1, r_2, r_3 :

$$r_1 = 2B_\alpha^{+2}(2depth) - 2depth + H_\varepsilon \quad / \quad r_2 = H_\varepsilon \quad / \quad r_3 = B_\alpha^{+2}(2depth).$$

The bounds on r_2, r_3 follow immediately from conditions 1 and 3. The bound on r_1 is obtained by observing the path in $T(S' \cup \{x\})$ connecting x and $t_{\hat{v}}$, where \hat{v} is an arbitrary leaf of \hat{t}_{env} : condition 2 implies a bound of $B_\alpha^{+2}(2depth) - depth$ on the weight of the first and last edges, and the weight of the rest of the path is bounded by H_ε since it consists of edges whose weight is at most ε . \square

6.1 Guaranteeing Reconstruction of Short Edges

In this section we give a closer look at the weight threshold ε , above which edges in T are guaranteed (correct) reconstruction by our algorithm. Such weight thresholds were provided for several other algorithms by Atteson in [1] (by what Atteson terms as ‘edge l_∞ radius’). However, since Atteson ties this threshold to the worst-case noise, the best threshold his

analysis provides is $2\alpha(\text{diam}(T))$ (achieved e.g. by Buneman’s algorithm [4] and DLCA [17]). To imply (adaptive) fast convergence in common models of evolutions (as will be discussed in Section 8), we need our algorithm to imply a tighter threshold $\varepsilon = 4\alpha(c \cdot \text{depth}(T))$ for some constant c . To achieve this stronger threshold, we use the following corollary of Theorem 6.9.

Corollary 6.10. *Let T, \widehat{D} and α be as in Theorem 6.9. Let $\varepsilon \in \mathbb{R}^+$ be s.t. $\varepsilon \geq 4\alpha(r')$, where $r' = 4\text{depth}(T) + 1.75\varepsilon + 2\text{diam}(T, \varepsilon)$. Then algorithm *Incremental_Reconstruct* (Section 3) returns an ε -contraction of T .*

Proof. According to Theorem 6.9, all we need to show is that $\varepsilon \geq 4\alpha(B_\alpha^+(r))$ for $r = 3B_\alpha^{+2}(2\text{depth}(T)) - 2\text{depth}(T) + 2\text{diam}(T, \varepsilon)$. The fact that $\varepsilon \geq 4\alpha(r')$ implies that for every $d \leq r'$, we have $B_\alpha^+(d) \leq d + \alpha(r') \leq d + \frac{\varepsilon}{4}$. So we get:

$$\begin{aligned} 2\text{depth}(T) \leq r' &\Rightarrow B_\alpha^+(2\text{depth}(T)) \leq 2\text{depth}(T) + \frac{\varepsilon}{4} \leq r' \Rightarrow \\ B_\alpha^{+2}(2\text{depth}(T)) &\leq 2\text{depth}(T) + \frac{\varepsilon}{2} \Rightarrow r \leq 4\text{depth}(T) + 1.5\varepsilon + 2\text{diam}(T, \varepsilon) \leq r' \Rightarrow \\ B_\alpha^+(r) &\leq 4\text{depth}(T) + 1.75\varepsilon + 2\text{diam}(T, \varepsilon) = r' \Rightarrow \varepsilon \geq 4\alpha(B_\alpha^+(r)). \quad \square \end{aligned}$$

Assume that T and ε are such that $\text{depth}(T) \geq 1.75\varepsilon + 2\text{diam}(T, \varepsilon)$. In this case, Corollary 6.10 implies that our algorithm is guaranteed to correctly reconstruct all edges in T which are longer than $\varepsilon = 4\alpha(5\text{depth}(T))$. The above assumption ($\text{depth}(T) \geq 1.75\varepsilon + 2\text{diam}(T, \varepsilon)$) is likely to hold whenever we are interested (a-priori) in weight-thresholds which are not too large (eg, when most of the edges of the tree are of weight larger than ε). In Section 9 we present a modification of our algorithm which guarantees a weight-threshold of $\varepsilon \leq 4\alpha(12\text{depth}(T))$ for *any* tree T . Whereas in practice we do not expect the modified algorithm to perform much better than the original one, its significance is in being adaptive fast convergent (see discussion in Section 8).

7 Complexity Analysis.

The space complexity of the algorithm (disregarding the space needed for storing the input dissimilarity matrix \widehat{D}) is obviously linear in n (the number of taxa), since the current topology \widehat{T} and the directional representatives can easily be maintained in linear space throughout the algorithm. The time complexity of the algorithm is quadratic, which is asymptotically optimal for algorithms reconstructing a phylogenetic tree with unbounded vertex-degrees, even from the exact tree-induced metric (see [10]). Each iteration involves selecting a taxon for insertion and applying *Attach_Taxon*. Note that computing the next taxon to be inserted (the one closest to S') can be done in linear time as in Dijkstra MST algorithm [6]. The most time consuming task in *Attach_Taxon* is computing the insertion zone. This can be done by querying the directional oracle on all vertices of \widehat{T} , and then pruning \widehat{T} in a DFS-traversal according to the queries’ results. The DFS-traversal and pruning is clearly linear in n . By Claim 5.3, the total time complexity of all queries to the directional oracle *PDO* is linear in the sum of vertex-degrees, which is also linear in n .

8 Reliable Reconstruction from Biological Sequences.

In this section we study the inter-taxa distances and the noise function induced by a stochastic process of sequence evolution. Since the model is stochastic, the resulting noise function α will be ‘probabilistic’ in the sense that it bounds the noise only with sufficiently high probability. We then use this noise function together with the results of Section 6 to establish the (inverse) relation between the input sequence length and weights of contracted edges.

8.1 Stochastic Model of Evolution

The results presented in this section assume the Cavender-Farris-Neyman [5, 15, 25] (CFN) model of *binary* sequence evolution, but they may be generalized to more complex models as well. The CFN model assumes a rooted tree T , whose edges are associated with symmetric *changing probabilities* $\{p_e\}_{e \in E(T)}$. The process of evolution is modelled by uniformly randomizing a binary state (0 or 1) at the root and propagating mutations along the tree edges according to their changing probabilities. A *site* is defined by the n random states generated by the above process at the leaves of the tree. Note that under a given model-tree, the probability distribution of a specific site is well defined. Repeating this process k times (independently), yields n binary sequences of length k (corresponding to k sites), which may serve as input to a *phylogenetic reconstruction method*.

The (additive) metric D_T induced by the model-tree T is defined by assigning the following *weight* to each edge e in T : $w_e = -\frac{1}{2} \ln(1 - 2p_e)^2$. For $u, v \in V(T)$, denote by p_{uv} the *compound changing probability* between u and v , which is the probability of observing different states in u and v . The corresponding tree metric D_T is given by the following equality:

$$d_T(u, v) = \sum_{e \in \text{path}(u, v)} w_e = -\frac{1}{2} \ln(1 - 2p_{uv}).$$

Given a pair of sequences (of length k) corresponding to taxa i, j , the *observed* compound changing probability \hat{p}_{ij} is estimated by the normalized hamming distance (i.e. the number of sites in the two sequences with different states divided by k). The *observed pairwise dissimilarities* are defined accordingly - $\hat{d}(i, j) = -\frac{1}{2} \ln(1 - 2\hat{p}_{ij})$. Our analysis is largely based on the following result:

Theorem 8.1. *Let D_T be a metric induced by a phylogenetic tree T , and let \hat{D} be an observed pairwise-dissimilarity matrix derived (as described above) from length- k sequences which evolved along T . Then with probability greater than $1 - \frac{1}{n}$, \hat{D} and D_T are α_k -close, where α_k is given by:³*

$$\alpha_k(d) = -\frac{1}{2} \ln \left[1 - e^{2d} \sqrt{\frac{6 \ln(n)}{k}} \right].$$

In Section 8.2 we provide a proof for this theorem, and in Section 8.3 we discuss its implication on the (adaptive) fast convergence of our algorithm.

8.2 A Noise Bound for the CFN Model

In bounding the noise for the CFN model, we concentrate on proving the following result:

Lemma 8.2. *For any $\delta > 0$, let $\alpha_{k, \delta}$ be defined as follows:*

$$\alpha_{k, \delta}(d) = -\frac{1}{2} \ln \left[1 - e^{2d} \sqrt{\frac{2}{k} \ln \left(\frac{2}{\delta} \right)} \right]. \quad (6)$$

Then the tree-induced metric D_T and the observed dissimilarity matrix \hat{D} are $\alpha_{k, \delta}$ -close with probability larger than $1 - \binom{n}{2} \delta$.

Note that Theorem 8.1 is obtained from Lemma 8.2 by substituting $\delta = \frac{2}{n^3}$. This lemma is proved by bounding the deviation of the observed dissimilarities from the tree-induced distances (see Lemma 8.4). The bound on this deviation follows the following basic bound, implied by Hoeffding's inequality.

²When assuming that the changes obey a Poisson distribution, this weight is the expected number of changes that occurred along e - see e.g. discussion in [16] pp. 156-157

³we use the convention that for $z \leq 0$, $\ln(z) = -\infty$.

Lemma 8.3 ([26], Theorem 4.5). *Let X_1, \dots, X_k be independent random variables which get the value 1 with probability p and 0 with probability $1 - p$, and let $\hat{X}_k = \frac{\sum_{i=1}^k X_i}{k}$. Then for every $\lambda > 0$,*

$$\Pr\left(\hat{X}_k - p > \lambda\right) \leq \exp(-2k\lambda^2) \quad (7)$$

$$\Pr\left(\hat{X}_k - p < -\lambda\right) \leq \exp(-2k\lambda^2) \quad (8)$$

It is not hard to see that for every taxon-pair i, j , \hat{p}_{ij} (as defined in Section 8.1) is an average of k random variables satisfying Lemma 8.3. Hence, the deviation of \hat{p}_{ij} from its expected value p_{ij} can be bounded using this lemma. Lemma 8.4 below translates this deviation to the deviation between tree-induced distances and observed dissimilarities.

Lemma 8.4. *Let d be the tree-induced distance between two taxa, and let \hat{d} be the observed dissimilarity between these two taxa. Then for any $\beta > 0$ we have:*

$$\Pr\left(d - \hat{d} > \beta\right) \leq \exp\left(-\frac{k}{2} \frac{(e^{2\beta} - 1)^2}{e^{4d}}\right) \quad (9)$$

$$\Pr\left(d - \hat{d} < -\beta\right) \leq \exp\left(-\frac{k}{2} \frac{(1 - e^{-2\beta})^2}{e^{4d}}\right). \quad (10)$$

Proof. Let p, \hat{p} be the real and observed compound changing probabilities between the two taxa mentioned in the lemma. First we establish §9:

$$\begin{aligned} \Pr\left(d - \hat{d} > \beta\right) &= \Pr\left(\frac{1}{2} \ln\left(\frac{1 - 2\hat{p}}{1 - 2p}\right) > \beta\right) = \\ &= \Pr\left(\frac{1 - 2\hat{p}}{1 - 2p} > e^{2\beta}\right) = \\ &= \Pr\left(1 + 2\frac{p - \hat{p}}{1 - 2p} > e^{2\beta}\right) = \\ &= \Pr\left(p - \hat{p} > \frac{1}{2}(e^{2\beta} - 1)(1 - 2p)\right) = \\ &= \Pr\left(p - \hat{p} > \frac{1}{2}(e^{2\beta} - 1)e^{-2d}\right). \end{aligned} \quad (11)$$

The inequality in §9 is now obtained by plugging $\lambda = \frac{1}{2}(e^{2\beta} - 1)e^{-2d}$ in §8. The inequality in §10 is similarly obtained by first showing (as in §11) that:

$$\Pr\left(d - \hat{d} < -\beta\right) = \Pr\left(p - \hat{p} < \frac{1}{2}(e^{-2\beta} - 1)e^{-2d}\right), \quad (12)$$

and then plugging $\lambda = \frac{1}{2}(1 - e^{-2\beta})e^{-2d}$ in §7. \square

Note that the bounds we get in §9 and in §10 are not identical: the RHS of §10 is greater than the RHS of §9, because $\beta > 0$ implies that $1 - e^{-2\beta} < e^{2\beta}(1 - e^{-2\beta}) = e^{2\beta} - 1$. Hence we get:

$$\Pr\left(|d - \hat{d}| > \beta\right) < 2 \exp\left(-\frac{k}{2} \frac{(1 - e^{-2\beta})^2}{e^{4d}}\right). \quad (13)$$

The bound in §14 below is obtained by noticing that d and \hat{d} are interchangeable in the proof of Lemma 8.4:

$$\Pr\left(|d - \hat{d}| > \beta\right) < 2 \exp\left(-\frac{k}{2} \frac{(1 - e^{-2\beta})^2}{e^{4 \min\{d, \hat{d}\}}}\right). \quad (14)$$

Proof of Lemma 8.2: The noise function $\alpha_{k,\delta}$ defined in §6 is obtained by extracting the value of β for which the RHS of §13 equals δ . Hence for every taxon pair $i, j \in S$, we get the following inequality by plugging in $\alpha_{k,\delta}(\min\{d(i, j), \hat{d}(i, j)\})$ as β in §14:

$$\Pr\left(|d(i, j) - \hat{d}(i, j)| > \alpha_{k,\delta}(\min\{d(i, j), \hat{d}(i, j)\})\right) < \delta. \quad (15)$$

Now, by applying a simple union bound, we get that D_T and \hat{D} are $\alpha_{k,\delta}$ -close with probability at least $1 - \binom{n}{2}\delta$. \square

8.3 Sequence Length Requirements

By combining Theorems 8.1 and 6.9 we are able to establish the relation between k and the weight of edges our algorithm contracts.

Theorem 8.5. *Let \hat{D} be a dissimilarity matrix obtained from n binary taxon-sequences of length k which evolved according to the CFN model along a phylogenetic tree T . Let $\varepsilon \in \mathbb{R}^+$ be s.t.*

$$\varepsilon \geq 4\alpha_k(B_{\alpha_k}^+(3B_{\alpha_k}^{+2}(2\text{depth}(T)) - 2\text{depth}(T) + 2H_\varepsilon)) \quad (16)$$

(where α_k is as defined in Theorem 8.1 and $H_\varepsilon = \text{diam}(T, \varepsilon)$).

Then when executed on \hat{D} , algorithm *Incremental_Reconstruct* (Section 3) returns an ε -contraction of T with probability larger than $1 - \frac{1}{n}$.

Proof. This theorem is a direct result of Theorems 6.9 and 8.1 \square

Theorem 8.5 states a condition which allows us to calculate, for a given tree T and input sequence length k , an upper bound ε on the weight of edges our algorithm contracts. The following lemma provides a more explicit (but less tight) connection between k and ε .

Corollary 8.6. *Let $\varepsilon \in \mathbb{R}^+$, and assume that the sequence length satisfies:*

$$k \geq \frac{24 \ln(n) \cdot e^{16\text{depth}(T) + 8\varepsilon + 8H_\varepsilon}}{\varepsilon^2}. \quad (17)$$

Then algorithm *Incremental_Reconstruct* returns an ε -contraction of T with high probability.

Proof. Corollary 6.10 implies that it is sufficient to show that $\varepsilon \geq 4\alpha_k(r')$ for $r' = 4\text{depth}(T) + 1.75\varepsilon + 2H_\varepsilon$. By plugging in α_k as defined in Theorem 8.1, we get that this is equivalent to requiring:

$$k \geq \frac{6 \ln(n) \cdot e^{16\text{depth}(T) + 7\varepsilon + 8H_\varepsilon}}{(1 - e^{-\frac{\varepsilon}{2}})^2}. \quad (18)$$

So we are left to show that §17 implies §18. This is established by showing that

$$4 \frac{e^\varepsilon}{\varepsilon^2} \geq (1 - e^{-\frac{\varepsilon}{2}})^{-2}. \quad (19)$$

Substituting $\varepsilon = 2t$, §19 is equivalent to $\frac{e^{2t}}{t^2} \geq (1 - e^{-t})^{-2}$, or by taking square root to $\frac{e^t}{t} \geq (1 - e^{-t})^{-1}$, which is equivalent to the well known inequality $e^t - 1 \geq t$. \square

Theorem 8.7. *Algorithm *Incremental_Reconstruct* is fast converging.*

Proof. The proof rather straightforwardly follows from Corollary 8.6. In order to establish fast convergence, we need to prove that our algorithm reconstructs the correct topology of T with high probability from sequences of $\text{poly}(n)$ length, when edge-weights of T are assumed to be within some interval $[f, g]$ (see §2). Applying Corollary 8.6 with any $0 < \varepsilon < f$, implies that our algorithm returns the correct topology with high probability, whenever $k > \frac{24}{f^2} \ln(n) \cdot \exp(16\text{depth}(T) + 8f)$. As mentioned already in Section 1, $\text{depth}(T) \leq g \log_2(n)$, which gives us the desired polynomial-bounded on the required sequence length. \square

Note that our algorithm is also *absolute* fast converging since its input (consisting of \widehat{D} and the sequence length k) does not require any knowledge about the originating tree.

For *adaptive* fast convergence we need that sequences of length $k = \frac{\exp(\text{depth}(T))}{\varepsilon^2}$ suffice to guarantee correct reconstruction (w.h.p.) of all edges of weight greater than ε (see §3). Hence Corollary 8.6 implies adaptive fast convergence only when $\text{diam}(T, \varepsilon) = O(\text{depth}(T))$. As mentioned in Section 6.1, this restriction is likely to hold in most “normal looking” trees (which do not have large clusters of short edges). However, for our algorithm to be adaptive fast convergent we need to remove this restriction. This is done in the next section.

9 Eliminating Dependence on the Diameter of Contracted Subtrees.

In this section we present an alternative insertion procedure which does not depend on the diameter of contracted subtrees. For this we guarantee that the algorithm queries only quartets whose diameter is bounded by some linear function of $\text{depth}(T)$. In our discussion we focus on a single insertion iteration in which taxon x is inserted into the current topology \widehat{T} spanning taxon set S' . The main idea behind this alternative insertion procedure is to compute the insertion zone over a sub-phylogeny induced by a subset $S'' \subseteq S'$ containing only taxa which are close to x . This approach is justified by Lemmas 9.1 and 9.2.

Lemma 9.1. *Assume that $\widehat{T}(S'')$ is an internal contraction of $T(S'')$ and that PQO is (r, ε) -reliable for $r = \text{diam}(T(S'' \cup \{x\}))$. Then PDO is ε -reliable for $(\widehat{T}(S''), x)$.*

Proof. This lemma is directly implied by Lemma 5.6 and the discussion following it. \square

Lemma 9.2. *Let \widehat{T} be an internal contraction of $T(S')$, and Let $S'' \subseteq S'$. Then*

1. $\widehat{T}(S'')$, the subtree of \widehat{T} induced by S'' , is an internal contraction of $T(S'')$.
2. An edge $(u, v) \in E(T(S''))$ is contracted into a vertex \hat{v} of $\widehat{T}(S'')$ iff all the edges in $\text{path}_{T(S')} (u, v)$ are contracted into \hat{v} in \widehat{T} .

Sketch of Proof. We prove the lemma for the case where $S'' = S' \setminus \{s\}$ (the general case follows by induction on $|S' \setminus S''|$). To prove 1, we show that $\widehat{T}(S'')$ is an internal contraction of $T(S'')$ by mapping each internal vertex \hat{v} of $\widehat{T}(S'')$ onto a subtree $t''_{\hat{v}}$ of $T(S'')$, as follows. Consider an arbitrary vertex $\hat{v} \in \text{internal}(\widehat{T})$, and the subtree $t_{\hat{v}}$ of $T(S')$ contracted into it in \widehat{T} . If $V(t_{\hat{v}}) \subseteq V(T(S''))$, then $t_{\hat{v}}$ is also a subtree of $T(S'')$, and $t''_{\hat{v}} = t_{\hat{v}}$. If this is not the case, then $t_{\hat{v}}$ must contain p_s , the parent of s in $T(S')$ (since p_s is the only possible vertex in $\text{internal}(T(S')) \setminus \text{internal}(T(S''))$). Furthermore, in this case we must have that $\text{deg}_{T(S')} (p_s) = 3$, and $N_{T(S')} (p_s) = \{s, u, v\}$ for some edge $(u, v) \in E(T(S''))$ (and $(u, p_s), (p_s, v) \in E(T(S'))$). There are three possible cases:

1. $t_{\hat{v}}$ contains neither (u, p_s) nor (p_s, v) , in which case $V(t_{\hat{v}}) = \{p_s\}$, and $\hat{v} \notin V(\widehat{T}(S''))$.
2. $t_{\hat{v}}$ contains exactly one of the edges $(u, p_s), (p_s, v)$ - w.l.o.g. it is (u, p_s) . Then $t''_{\hat{v}} = t_{\hat{v}} \setminus \{(u, p_s)\}$.
3. $t_{\hat{v}}$ contains both (u, p_s) and (p_s, v) , in which case $t''_{\hat{v}} = t_{\hat{v}} \setminus \{(u, p_s), (p_s, v)\} \cup \{(u, v)\}$.

2 is implied by the fact that the edge (u, v) of $T(S'')$ is contracted in $\widehat{T}(S'')$ only in the third case. \square

In order to use the reduced subset S'' in the insertion of x into \widehat{T} , we need the ε -reliability of PDO for $(\widehat{T}(S''), x)$ (guaranteed by Lemma 9.1) to translate also to \widehat{T} . This is obtained by making sure that the anchor of x in $\widehat{T}(S'')$ is the same as its anchor in \widehat{T} (to be proved

in Theorem 9.7). To achieve this, we require S'' to *preserve* the anchor of x in $T(S')$, as specified by the following definitions and results.

Definition 9.3. Let v be an internal vertex of a phylogenetic tree T . The v -components of T are the connected components of $T(V \setminus \{v\})$.

Definition 9.4. Let T be a phylogenetic tree, let $v \in V(T)$ and let $\tilde{S} \subseteq \text{leaves}(T)$. We say that \tilde{S} preserves v in T , if either v is a taxon in \tilde{S} , or \tilde{S} has nonempty intersection with at least three of the v -components of T . \tilde{S} is said to preserve an edge (u, v) in T if it preserves both u and v in T .

Observation 9.5. A vertex v of $T(S')$ is also a vertex in $T(S'')$ iff S'' preserves v in $T(S')$. Similarly, an edge (u, v) of $T(S')$ is also an edge of $T(S'')$ iff S'' preserves the edge (u, v) in $T(S')$. (see Fig. 10)

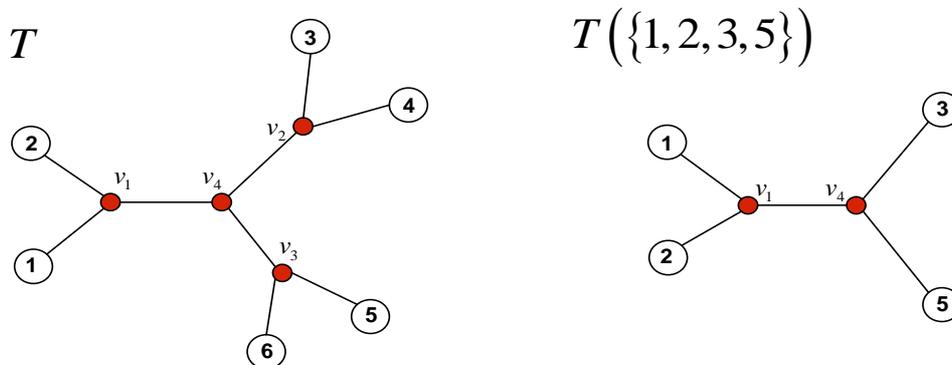


Figure 10: **Vertex and Edge Preservation.** The taxon subset $\{1, 2, 3, 5\}$ preserves the internal vertices v_1, v_4 and the edges $(1, v_1), (2, v_1), (v_1, v_4)$ in T .

Lemma 9.6. Let \hat{T} be an internal edge contraction of $T(S')$, and assume that S'' preserves $\text{anchor}_{T(S')}(x)$. Then $\text{anchor}_{\hat{T}}(x) = \text{anchor}_{\hat{T}(S'')}(x)$.

Proof. Assume that $\text{anchor}_{T(S')}(x)$ is an edge e (the proof for the case that $\text{anchor}_{T(S')}(x)$ is an internal vertex is similar but simpler). Then by Observation 9.5, e is also an edge in $T(S'')$. Assume first that e is contracted in \hat{T} into some vertex \hat{v} , implying that $\text{anchor}_{\hat{T}}(x) = \hat{v}$. Then by Lemma 9.2(2), e is also contracted into \hat{v} in $\hat{T}(S'')$, implying that $\text{anchor}_{\hat{T}(S'')}(x) = \hat{v}$ as well. Now assume that e is not contracted in \hat{T} , then (by Lemma 9.2(2)) it is also not contracted in $\hat{T}(S'')$. Thus, both \hat{T} and $\hat{T}(S'')$ contain an edge \hat{e} which is identical to e (i.e., \hat{e} and e induce the same splits in the corresponding trees). Hence \hat{e} is the anchor of x in both \hat{T} and $\hat{T}(S'')$. \square

Lemmas 9.1 and 9.6 suggest the following insertion procedure:

Procedure *Attach_Taxon_revised*(\widehat{T}, x):

1. Calculate a subset $S'' \subseteq S'$, s.t. $\text{diam}(T(S'' \cup \{x\}))$ is small, and S'' preserves the anchor of x in $T(S')$.
2. $\hat{t}_{inz}'' \leftarrow \hat{t}_{inz}(\widehat{T}(S''), x, PDO)$.
3. If \hat{t}_{inz}'' is a single edge (\hat{u}, \hat{v}) , then attach x to \widehat{T} by introducing a new internal vertex \hat{p}_x and replacing (\hat{u}, \hat{v}) with the three edges $(\hat{u}, \hat{p}_x), (\hat{v}, \hat{p}_x), (x, \hat{p}_x)$.
4. If \hat{t}_{inz}'' has a single internal vertex \hat{v} (i.e. $V(\hat{t}_{inz}'') = N(\hat{v}) \cup \{\hat{v}\}$), then add to \widehat{T} the edge (\hat{v}, x) .
5. Else (i.e. \hat{t}_{inz}'' has at least one internal edge), contract into a new vertex \hat{v} all edges in \widehat{T} which lie on paths corresponding to internal edges of \hat{t}_{inz}'' , and add to \widehat{T} the edge (\hat{v}, x) .

Implementation note: Before computing the insertion zone in step 2, the algorithm constructs $\widehat{T}(S'')$ and determines all directional representatives. Since directional representatives have to be taxa in S'' , they are not kept from one iteration to the next. The construction of $\widehat{T}(S'')$ (including calculation of directional representatives) may be achieved in linear time using two DFS scans of \widehat{T} .

Theorem 9.7. *Assume that \widehat{T} is an ε -contraction of $T(S')$, and let \widehat{T}_{post} be the topology resulting from *Attach_Taxon_revised*(\widehat{T}, x). Then if S'' preserves the anchor of x in $T(S')$ and PDO is ε -reliable for $(\widehat{T}(S''), x)$, then \widehat{T}_{post} is an ε -contraction of $T(S' \cup \{x\})$.*

Proof. If \hat{t}_{inz}'' consists of a single edge (\hat{u}, \hat{v}) , then all we need to show is that (\hat{u}, \hat{v}) is the anchor of x in \widehat{T} : This holds since by Observation 3.4(1), $\text{anchor}_{\widehat{T}(S'')}(x) = (\hat{u}, \hat{v})$, and by Lemma 9.6 $\text{anchor}_{\widehat{T}}(x) = \text{anchor}_{\widehat{T}(S'')}(x)$. So assume \hat{t}_{inz}'' contains some internal vertices. We prove first that if \widehat{T} is an internal contraction of $T(S')$, then \widehat{T}_{post} is an internal contraction of $T(S' \cup \{x\})$. For this, let \tilde{t} be the subtree of \widehat{T} spanned by $\text{internal}(\hat{t}_{inz}'')$. Then \widehat{T}_{post} is obtained by contracting \tilde{t} into a new vertex \hat{v} and attaching x to \hat{v} . Hence, by Observation 3.6, it suffices to show that $\text{anchor}_{\widehat{T}}(x)$ touches \tilde{t} . This last claim holds since $\text{anchor}_{\widehat{T}}(x) = \text{anchor}_{\widehat{T}(S'')}(x)$ (lemma 9.6) and $\text{anchor}_{\widehat{T}(S'')}(x)$ touches \tilde{t} in $\widehat{T}(S'')$ (Observation 3.4(1)).

It remains to prove that all the contracted edges are of weight at most ε . Consider an edge e of $T(S' \cup \{x\})$ which is contracted in \widehat{T}_{post} . If e was contracted already in \widehat{T} then (by our assumption on \widehat{T}) $w(e) \leq \varepsilon$. Otherwise, by the ε -reliability of PDO for $(\widehat{T}(S''), x)$, e must lie on a path corresponding to an internal edge e' of $\hat{t}_{env}(\widehat{T}(S''), x, \varepsilon)$, and $w(e) \leq w(e') \leq \varepsilon$. \square

We are left to show how to find a reduced subset of taxa S'' s.t. $T(S'' \cup \{x\})$ has a small diameter and S'' preserves the anchor of x in $T(S')$. We assume, as in Section 6, that the algorithm has access to a dissimilarity matrix \widehat{D} which is α -close to the tree-induced metric D_T . We also assume the insertion order suggested in Section 6. Let S'_i be the set of attached taxa in the beginning of the i^{th} iteration, and let $x_i \notin S'_i$ be the taxon chosen for insertion at that stage. Then there is a taxon $y_i \in S'_i$ s.t. $\hat{d}(x_i, y_i) = \min\{\hat{d}(x', y') : x' \notin S'_i, y' \in S'_i\}$. Our definition of S'' relies on the following observation (which can be proved by induction on i):

Observation 9.8. *Let $\bar{d}_i = \max_{j \leq i} \{\hat{d}(x_j, y_j)\}$. Then $\text{depth}(T(S'_i \cup \{x_i\})) \leq B_\alpha(\bar{d}_i)$.*

Lemma 9.9. Consider the i^{th} iteration in which taxon $x = x_i$ is inserted into \widehat{T} (where $S' = \text{leaves}(\widehat{T})$). Then the following set preserves the anchor of x in $T(S')$:

$$S'' = \left\{ s \in S' : \hat{d}(x, s) \leq B_\alpha^+ (3B_\alpha^+(\bar{d}_i)) \right\}$$

Proof. Assume that the anchor of x in $T(S')$ is an edge (u, v) (the other case is proved similarly). Then we need to show that S'' preserves both u and v in $T(S')$. We will show w.l.o.g. that it preserves v . This is done by proving that each v -component of $T(S')$ contains a taxon s s.t. $\hat{d}(x, s) \leq B_\alpha^+ (3B_\alpha^+(\bar{d}_i))$. Notice that each v -component contains a taxon s s.t. $d_T(v, s) \leq \text{depth}(T(S'))$. By decomposing path connecting x and s in $T(S' \cup \{x\})$ into 3 parts we get:

$$d_T(x, s) \leq w(x, p_x) + w(p_x, v) + d_T(v, s) \leq 3\text{depth}(T(S' \cup \{x\})) \leq 3B_\alpha^+(\bar{d}_i) .$$

The lemma then follows since $\hat{d}(x, s) \leq B_\alpha^+(d_T(x, s))$. \square

The final part left in the analysis is to bound the diameter of $T(S'' \cup \{x\})$. Let s_1, s_2 be two arbitrary taxa in S'' . Then $\hat{d}(x, s_1), \hat{d}(x, s_2) \leq B_\alpha^+ (3B_\alpha^+(\bar{d}_i))$, and we get:

$$d_T(s_1, s_2) \leq d_T(s_1, x) + d_T(s_2, x) \leq B_\alpha^+(\hat{d}(s_1, x)) + B_\alpha^+(\hat{d}(s_2, x)) \leq 2B_\alpha^{+2} (3B_\alpha^+(\bar{d}_i)) \quad (20)$$

Now, Lemma 6.8(1) implies that $\bar{d}_i \leq B_\alpha^+(2\text{depth}(T))$ in every iteration of the algorithm. This gives us the following bound:

$$\text{diam}(T(S'' \cup \{x\})) \leq 2B_\alpha^{+2} (3B_\alpha^{+2}(2\text{depth}(T))) . \quad (21)$$

We conclude this section by outlining the adjustment of the the main results from Sections 6 and 8 to the case when *Attach_Taxon_revised* is used in each iteration of *Incremental_Reconstruct*.

Theorem 9.10. Consider a phylogenetic tree T over a taxon-set S . Let \widehat{D} be a dissimilarity matrix which is α -close to the tree-induced metric D_T , for some non-decreasing function α . Then the modified algorithm returns a topology which is an ε -contraction of T where

$$\varepsilon = 4\alpha (B_\alpha^+ (2B_\alpha^{+2} (3B_\alpha^{+2}(2\text{depth}(T)))))) . \quad (22)$$

Proof. We show (by induction) that in every iteration of the algorithm, the current topology \widehat{T} is an ε -contraction of $T(S')$. Consider an arbitrary iteration where x is inserted into \widehat{T} (which is assumed to be an ε -contraction of $T(S')$). Let S'' be the reduced subset calculated by *Attach_Taxon_revised* (as defined in Lemma 9.9). The definition of ε and the bound established on $\text{diam}(T(S'' \cup \{x\}))$ imply by Lemma 6.3 that $FPM_{\widehat{D}, \alpha}$ is an (r, ε) -reliable quartet oracle for $r = \text{diam}(T(S'' \cup \{x\}))$. And so by Lemma 9.1, PDO is an ε -reliable directional oracle for $(\widehat{T}(S''), x)$. Hence, from Theorem 9.7 (together with Lemma 9.9) we get that $\widehat{T}_{\text{post}}$ is an ε -contraction of $T(S' \cup \{x\})$. \square

Adjusting Theorem 9.10 to the probabilistic setting of Section 8 yields the following:

Theorem 9.11. Let \widehat{D} be a dissimilarity matrix obtained from n binary taxon-sequences of length k which evolved according to the CFN model along a phylogenetic tree T . Then when executed on \widehat{D} , the modified algorithm returns with probability larger than $1 - \frac{1}{n}$ an ε -contraction of T , where

$$\varepsilon = 4\alpha_k (B_{\alpha_k}^+ (2B_{\alpha_k}^{+2} (3B_{\alpha_k}^{+2}(2\text{depth}(T)))))) . \quad (23)$$

We now turn to prove the adaptive fast convergence of the modified algorithm. To establish this we have to determine the sequence length k required for correct reconstruction of edges whose weight is greater than ε . First we simplify equation §23 similarly to the way Corollary 6.10 simplifies equation §5.

Lemma 9.12. *Let $\varepsilon \in \mathbb{R}^+$ be s.t. $\varepsilon \geq 4\alpha (12\text{depth}(T) + \frac{17}{4}\varepsilon)$, then $\varepsilon \geq 4\alpha (B_\alpha^+ (2B_\alpha^{+2} (3B_\alpha^{+2}(2\text{depth}(T)))))$.*

Proof. Similar to the proof of Corollary 6.10. □

Theorem 9.13. *The incremental algorithm using `Attach_Taxon_revised` is adaptive fast converging.*

proof outline. In order to establish adaptive fast convergence it is enough to show that assuming edge weights in T are not greater than g , sequences of length $k = \frac{n^{O(g)}}{\varepsilon^2}$ are sufficient for the modified algorithm to return (w.h.p.) an ε -contraction of T (see §3). According to Theorem 9.11 and Lemma 9.12, ε contraction is guaranteed w.h.p. if $\varepsilon \geq 4\alpha_k (12\text{depth}(T) + \frac{17}{4}\varepsilon)$. Using similar arguments to the ones used in the proof of Corollary 8.6, we get that this is ensured whenever:

$$k \geq \frac{24 \ln(n) \cdot e^{48\text{depth}(T)+18\varepsilon}}{\varepsilon^2} . \quad (24)$$

Now assuming g is an upper bound on edge weights, we have $\text{depth}(T) \leq g \log_2(n)$ and w.l.o.g. $\varepsilon \leq g$ (otherwise all edge weights are smaller than ε , and hence any execution of our algorithm outputs an ε -contraction of T). Hence the required sequence length is smaller than

$$\frac{24 \ln(n) \cdot n^{48g \log_2 e} \cdot e^{18\varepsilon}}{\varepsilon^2} = \frac{n^{O(g)}}{\varepsilon^2},$$

as required. □

Acknowledgement:

The first author would like to thank Elchanan Mossel for a very helpful discussion at the early stages of this research.

References

- [1] K. Atteson. The performance of neighbor-joining methods of phylogenetic reconstruction. *Algorithmica*, 25:251–278, 1999.
- [2] V. Berry and D. Bryant. Faster reliable phylogenetic analysis. In *RECOMB '99: Proceedings of the third annual international conference on Computational molecular biology*, pages 59–68, 1999.
- [3] W. Beyer, M. Singh, T. Smith, and M. Waterman. Additive evolutionary trees. *J Theor Biol*, 64(2):199–213, January 1977.
- [4] P. Buneman. The recovery of trees from measures of dissimilarity. *Mathematics in the Archeological and Historical Sciences*, pages 387–395, 1971.
- [5] J. Cavender. Taxonomy with confidence. *Math Biosci*, 40:271–280, 1978.
- [6] T. Cormen, C. Leiserson, R. Rivest, and C. Stein. *Introduction to Algorithms*. MITP, 2001. 2nd edition.
- [7] M. Cryan, L. Goldberg, and P. Goldberg. Evolutionary trees can be learned in polynomial time in the two-state general markov model. *SIAM Journal on Computing*, 31(2):375–397, 2001.
- [8] M. Csürös. Fast recovery of evolutionary trees with thousands of nodes. *Journal of Computational Biology*, 9(2):277–297, 2002.

- [9] M. Csürös and M. Kao. Recovering evolutionary trees through harmonic greedy triplets. In *SODA: ACM-SIAM Symposium on Discrete Algorithms*, pages 261–270, 1999.
- [10] J. Culberson and P. Rudnicki. A fast algorithm for constructing trees from distance matrices. *Information Processing Letters*, 30(4):215–220, February 1989.
- [11] C. Daskalakis, C. Hill, A. Jaffe, R. Mihaescu, E. Mossel, and S. Rao. Maximal accurate forests from distance matrices. In *RECOMB '06: Proceedings of the tenth annual international conference on Computational molecular biology*, pages 281–295, 2006.
- [12] C. Daskalakis, E. Mossel, and S. Roch. Optimal phylogenetic reconstruction. In *STOC '06: Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, pages 159–168, 2006.
- [13] P. Erdos, M. Steel, L. Szekely, and T. Warnow. A few logs suffice to build (almost) all trees (I). *Random Structures and Algorithms*, 14:153–184, 1999.
- [14] P. Erdos, M. Steel, L. Szekely, and T. Warnow. A few logs suffice to build (almost) all trees (II). *Theoretical Computer Science*, 221:77–118, 1999.
- [15] J. Farris. A probability model for inferring evolutionary trees. *Systematic Zoology*, 22:250–256, 1973.
- [16] J. Felsenstein. *Inferring Phylogenies*. Sinauer Associated, Inc., Sunderland, MA, 2004.
- [17] I. Gronau and S. Moran. Neighbor joining algorithms for inferring phylogenies via LCA-distances. *Journal of Computational Biology*, 14(1):1–15, 2007.
- [18] D. Huson, S. Nettles, and T. Warnow. Disk-Covering, a fast-converging method for phylogenetic tree reconstruction. *J Comp Biol*, 6:369–386, 1999.
- [19] T. Jukes and C. Cantor. Evolution of protein molecules. In H.N. Munro, editor, *Mammalian Protein Metabolism*, pages 21–132. Academic Press, New York, 1969.
- [20] V. King, L. Zhang, and Y. Zhou. On the complexity of distance-based evolutionary tree reconstruction. In *SODA: ACM-SIAM Symposium on Discrete Algorithms*, pages 444–453, 2003.
- [21] B. Moret, K. St. John, and T. Warnow. Absolute convergence: true trees from short sequences. In *SODA: ACM-SIAM Symposium on Discrete Algorithms*, pages 186–195, 2001.
- [22] E. Mossel. Phase transitions in phylogeny. *Trans Amer Math Soc*, 356:2379–2404, 2004.
- [23] E. Mossel. distorted metrics on trees and phylogenetic forests. *ACM Transactions on computational biology and bioinformatics*, 4:108–116, 2007.
- [24] E. Mossel and M. Steel. *How much can evolved characters tell us about the tree that generated them?*, chapter 14. 2005.
- [25] J. Neymann. Molecular studies of evolution: A source of novel statistical problems. In S. Gupta and Y. Jackel, editors, *Statistical Decision Theory and Related Topics*, pages 1–27. Academic Press, New York, 1971.
- [26] L. Wasserman. *All of Statistics*. Springer, New York, 2004.