

# Reconstruction of Ancestral Genomic Sequences Using Likelihood\*

Isaac Elias

isaac@nada.kth.se

Dept. of Numerical Analysis and Computer Science,  
KTH, Stockholm, SE-106 91, Sweden  
phone: +46 (0)8-5537 8574, fax: +46 (0)8-5537 8214

Tamir Tuller

tamirtul@post.tau.ac.il

School of Computer Science, Tel-Aviv University, Israel

## Abstract

A challenging task in computational biology is the reconstruction of genomic sequences of extinct ancestors, given the phylogenetic tree and the sequences at the leafs. This task is best solved by calculating the most likely estimate of the ancestral sequences, along with the most likely edge lengths. We deal with this problem and also the variant in which the phylogenetic tree in addition to the ancestral sequences need to be estimated. The latter problem is known to be NP-hard, while the computational complexity of the former is unknown. Currently, all algorithms for solving these problems are heuristics without performance guarantees. The biological importance of these problems calls for developing better algorithms with guarantees of finding either optimal or approximate solutions.

We develop approximation, fixed-parameter tractable (FPT), and fast heuristic algorithms for two variants of the problem: when the phylogenetic tree is known and when it is unknown. The approximation algorithm guarantees a solution with a log-likelihood ratio of 2 relative to the optimal solution. The FPT has a running time which is polynomial in the length of the sequences and exponential in the number of taxa. This makes it useful for calculating the optimal solution for small trees. Moreover, we combine the approximation algorithm and the FPT into an algorithm with arbitrary good approximation guarantee (PTAS).

We tested our algorithms on both synthetic and biological data. In particular, we used the FPT for computing the most likely ancestral mitochondrial genomes of *hominidae* (the great apes), thereby answering an interesting biological question. Moreover, we show how the approximation algorithms find good solutions for reconstructing the ancestral genomes for a set of *lentiviruses* (relatives of HIV).

Availability: Supplementary material of this work is available at  
<http://www.nada.kth.se/~isaac/publications/aml/aml.html>

---

\*Paper submitted for publication.

# 1 Introduction

A fundamental problem in molecular biology is to infer the evolution of a set of genomic sequences. Tightly linked with this problem are our assumptions of how sequences evolve. Based on our beliefs we state an optimization criterion by which the "correct" ancestral sequences should be selected. In this work, we try to find the most likely estimate of the ancestral sequences. We deal with two variants of the problem; when the phylogenetic tree is known, Small AML, and when only the sequences of the extant species are known, Big AML. This criterion for inferring ancestral sequences was first introduced by (Barry and Hartigan (1987)) about 20 years ago and is known as Ancestral Maximum Likelihood (AML) (a.k.a. most parsimonious likelihood).

The task of estimating the most likely ancestral sequences requires a probabilistic model of sequence evolution. The most elementary model is the two state Cavender-Farris-Neyman model (Cavender (1978); Farris (1973); Neyman (1971)). In this model, there are two symbols (0 and 1) and each edge in the tree is associated with a mutation probability. In this statistical framework, each tree and ancestral sequences has a probability of generating the observed sequences at the leafs. Thus, to compute the most likely ancestral sequences we need to either be given or estimate the tree and its edge lengths.

Altogether there are three variants of inferring the most likely ancestral sequences. The simplest variant is when the phylogenetic tree and its edge lengths are known. This variant can be solved in polynomial time by dynamic programming (Pupko *et al.* (2000)). Here we deal with Small AML, when the tree is known and its edge lengths are unknown, and Big AML, when the tree is unknown. For Small AML there is neither a known polynomial algorithm nor a proof of NP-hardness. The usual methods for solving this Small AML are heuristics such as hill climbing or EM (Friedman *et al.* (2002)). These methods guarantee nothing about the solution they find. In this work, we present an approximation algorithm which guarantees a solution with log-likelihood ratio less than 2 relative to the optimal solution. We show how the algorithm can be extended to get an arbitrary good approximation guarantee in polynomial time (PTAS). Moreover, we give an algorithm which is fixed-parameter tractable (FPT) in the size of the tree. This algorithm is guaranteed to find the optimal solution and has a running time which is polynomial in the length of the sequences while it is exponential in the size of the tree. This property makes the FPT very useful for finding the most likely ancestral sequences in small trees. We demonstrate how our FPT can find the most likely ancestral sequences for the mitochondrial genomes of hominidae. This question was originally asked by (Barry and Hartigan (1987)) who were the first to propose a heuristic for AML. Indeed, the solution of our FPT has much higher likelihood than the solution found in (Barry and Hartigan (1987)). Further, we use our approximation algorithms and heuristics for finding ancestral sequences with high likelihood for a set of *lentiviruses* (viruses from the HIV family). This shows how our algorithms can be used to solve interesting real world biological problems.

In Big AML only the sequences at the leafs are known. Thus, both the phylogenetic tree and the ancestral sequences need to be estimated. This variant is known to be NP-hard (Addario-Berry *et al.* (2004)). We show that, with the right modifications, the FPT and the 2-approximation algorithm are applicable also to this variant of the problem. We note that AML is an inconsistent estimator of the tree (no matter how long sequences are sampled the AML tree will not converge to the "true" tree) even though it is the best estimator of the ancestral sequences.

Mathematically, AML is similar to two common optimization criteria for reconstructing phylogenetic trees, Maximum Parsimony (MP) (Fitch (1971)) and Maximum Likelihood (ML) (Felsenstein (2003)). When the tree structure is unknown, all these problems are NP-hard (Day (1983); Foulds and Graham (1982); Chor and Tuller (2005)). The aim of MP is to find the phylogeny and ancestral sequences that explains the evolution using the fewest mutational events. ML requires, similarly to AML, an assumption of sequence evolution. However, as opposed to both AML and MP, the ML criterion does not include ancestral sequences. The aim of ML is instead to find the tree with highest probability averaged over all possible assignments to the internal vertices. One advantage of ML over MP is its consistency. There are cases where no matter how many samples of the original tree we use, MP will not find the right tree. However, there are no such cases for ML (Felsenstein (2003)). As was mentioned, AML is also an inconsistent estimator of the phylogenetic tree. There are similarities between AML and ML that have been observed by (Steel and Penny (2000)). Our work can, therefore, be seen as a step toward a deeper understanding of ML, which is the preferred method for estimating phylogenetic trees.

AML has been used as the criteria for finding ancestral sequences in many works, here we describe some of them. (Pagel (1999)) demonstrated and gave many examples for how to use AML for estimating

ancestral states. (Koshi and Goldstein (1996)) used AML for reconstructing the ancestral ribonuclease sequence. (Yang *et al.* (1995)) analyzed the lysozyme c sequences of six mammals by using the AML and MP criteria, they showed that AML is superior to MP. (Hudek and Brown (2005)) develop a method for aligning genomic sequences by first inferring ancestral sequences. (Krishnan *et al.* (2004)) tested AML by analyzing its ability to infer ancestral sequence distributions from primate mtDNA sequences and from simulated data. The famous Phylip package (Felsenstein (1993)) also has the ability to reconstruct ancestral genomic sequences by likelihood. Gaschen *et al.* (Gaschen *et al.* (2002)) use AML for selecting vaccine for HIV-1.

The rest of the paper is organized as follows. In Section 2, we formalize our notation and define the variants of AML that this paper deals with. Thereafter, in Section 3, we prove some properties that AML solutions have and use these to construct the FPT. In Section 4, we bound the likelihood of the optimal solution and develop the approximation algorithms. In Section 5, we extend our result to the Jukes-Cantor model. In Section 6, we examine our methods on synthetical and biological datasets. We finish with conclusions and some open questions in Section 7.

## 2 Preliminaries

In this paper, we are concerned with reconstructing the evolutionary history for a set of taxa  $\mathcal{T}$  containing  $|\mathcal{T}| = n$  sequences. The sequences are over the binary alphabet and have length  $m$ . In Section 5, we extend the results to include larger alphabets.

### Phylogeny

A *phylogenetic tree* is an unrooted binary tree  $T$  together with a *leaf labelling* function  $\lambda$ , with the vertex set  $V(T)$  and the edge set  $E(T)$ . The leaf labelling function is a bijection between the leaf set  $L(T)$  and a set of taxa  $\mathcal{T}$ . A *full labelling* of a phylogeny  $\hat{\lambda}$  is a labelling of all vertices of the tree such that the labelling at the leafs are the same as in the leaf labelling, i.e., for all  $l \in L(T)$   $\lambda(l) = \hat{\lambda}(l)$ .

### Likelihood

The Cavender-Farris-Neyman model (Cavender (1978); Farris (1973); Neyman (1971)) is the simplest probabilistic model of sequence evolution assumes that sites evolve independently and identically from the root down toward the leafs. Each edge in the tree has an associated mutation probability which specifies with what probability a site changes. We denote the edge probability for the edge  $e$  by  $p(e)$  and make the natural restriction  $0 \leq p(e) < \frac{1}{2}$ . In general, if two sequences of length  $m$  have hamming distance  $h$  and the mutation probability between these two sequences is  $p$  then the likelihood of observing two sequences as divergent as them is given by  $p^h \cdot (1 - p)^{m-h}$ . It is not difficult to see that the likelihood is maximized by  $p = \frac{h}{m}$ . Hence, the likelihood of observing a fully labeled tree  $(T, \hat{\lambda})$  with edge probability function  $p$  is

$$L(T, \hat{\lambda}, p) = \prod_{e \in E(T)} p(e)^{h(e)} \cdot (1 - p(e))^{m-h(e)},$$

where  $h(e)$  is the *hamming distance* of the two labels incident with the edge. Moreover, since the most likely edge probabilities are given by the hamming distance of the labels we have

$$\max_p L(T, \hat{\lambda}, p) = \prod_{e \in E(T)} \left(\frac{h(e)}{m}\right)^{h(e)} \cdot \left(1 - \frac{h(e)}{m}\right)^{m-h(e)}.$$

As a consequence, we exclude the edge probabilities and denote  $\max_p L(T, \hat{\lambda}, p)$  by  $L(T, \hat{\lambda})$  and the likelihood of the edge  $e$  by  $L(h(e))$ . This leads us to the definitions of the two problems we are concerned with:

- Given a set of sequences  $\mathcal{T}$  the (Big) *Ancestral Maximum Likelihood* (AML) problem is that of finding the tree  $T$  together with the most likely full labelling  $\hat{\lambda}$  such that each leaf is uniquely labeled by one of the sequences in  $\mathcal{T}$ , i.e.,  $\max_{(T, \hat{\lambda})} L(T, \hat{\lambda})$ .
- Given a phylogenetic tree  $(T, \lambda)$  the *Small Ancestral Maximum Likelihood* (Small AML) problem is that of finding the most likely full labelling, i.e.,  $\max_{\hat{\lambda}} L(T, \hat{\lambda})$ .

## Splits

Every position  $i \in [1, m]$  in the leaf sequences describes a bipartition of the taxa such that taxa with the symbol '0' in the position are in one partition and those with symbol '1' are in the other. Such bipartitions are called *splits* and we define the split associated with position  $i$  as

$$s_i = \{A, B\} \text{ s.t. } \forall t \in \mathcal{T} \begin{cases} t \in A, & \text{if } t[i] = '0' \\ t \in B, & \text{if } t[i] = '1' \end{cases}$$

Some positions may describe the same split and therefore we denote the number of times that split  $s$  occurs by  $\#s$  and denote the set of unique splits by  $S$ , i.e.,

$$\sum_{s \in S} \#s = m \quad \text{and} \quad |S| \leq m.$$

It is important to note that two positions describe the same split when the bipartition is the same and that this is not dependent on the exact symbols. That is, if  $s_i$  is defined as above and  $s_i = s_j$  then there are two cases, either

$$\forall t \in A \ t[j] = '0' \quad \text{or} \quad \forall t \in A \ t[j] = '1'.$$

## FPT

An algorithm is said to be *fixed-parameter tractable* (FPT) in the parameter  $k$  if its running time is  $O(f(k) \cdot m^c)$ , where  $m$  is the length of the input,  $f$  is some arbitrary function, and  $c$  a constant. In particular, FPT algorithms are of interest for problems which do not have known polynomial time algorithms and may provide a way for solving small instances.

## PTAS

An algorithm is said to be a *polynomial time approximation scheme* (PTAS) if it, for any input and any rational value  $r > 1$ , returns an  $r$ -approximate solution. The algorithm must be polynomial in the length of the input but may also depend on  $1/(r - 1)$ . That is, the better the approximation, the larger may be the running time.

## 3 Equal Evolution, Ancestral Sequence in Stars, and Fixed-Parameter Tractability

We start this section with the important notion that equal splits evolve identically. That is, if two positions  $i$  and  $j$  describe the same bipartition of the sequences then mutations in these positions occur on the same edges. Subsequently, this observation is used to construct the FPT algorithm and to find ancestral sequences in small stars.

**Lemma 3.1 (Equal Evolution).** *If two positions  $i$  and  $j$  describe the same split  $s_i = s_j$  then in any most likely labelling  $\lambda^*$  the evolution of these two sites are the same. That is, for any  $\lambda^*$ , maximizing  $\max_{\hat{\lambda}} L(T, \hat{\lambda})$ , one of (i) and (ii) holds:*

$$\begin{aligned} (i) \quad & \forall v \in V(T) \ \lambda^*(v)[i] = \lambda^*(v)[j] \\ (ii) \quad & \forall v \in V(T) \ \lambda^*(v)[i] \neq \lambda^*(v)[j]. \end{aligned}$$

*Proof.* The proof is by contradiction. For a phylogeny  $T$  let  $\lambda^*$  be a most likely full labelling and assume that  $s_i = s_j$ , but that the evolution in these two sites are not the same. Since the ancestral sequences are given by  $\lambda^*$ , the most likely edge-lengths are given by  $h(e)/m$ . There are two cases:

1. If the likelihood in site  $i$  is higher than in site  $j$  (or vice versa). Then the likelihood can be increased by simply changing the evolution in site  $j$  such that it is the same as in site  $i$ , without changing the edge-lengths.

2. Otherwise the likelihood in site  $i$  and  $j$  are the same. Again, without changing the edge-lengths, we can switch the evolution in  $j$ . This time the likelihood remains unchanged. However, since the most likely edge-lengths are given by  $h(e)/m$ , the old edge-lengths cannot be optimal.

□

Using the same arguments as in Lemma 3.1, it is easy to show that the equal evolution property holds also for non-binary trees.

**Observation 3.1.** *The equal evolution property holds also for non-binary trees.*

The equal evolution property can be drawn further. We say that  $T'$  is a *binary subtree* of the tree  $T$  if it can be attained from  $T$  by cutting edges in  $T$  (trees in this paper are binary and unrooted). Let  $\lambda^*$  be a most likely full labelling of  $T$  and consider a binary subtree  $T'$ .  $\lambda^*$  labels all vertexes of  $T$  and in particular this means that it labels all leafs of  $T'$ . Therefore, by the Lemma 3.1, all equal splits in the leaf sequences of  $T'$  have equal evolution in  $T'$ .

**Corollary 3.2 (Equal Evolution Cont.).** *Every most likely labelling of  $(T, \lambda^*)$  labels all binary subtrees of  $T$  such that the equal evolution property holds also for these subtrees.*

### 3.1 AML Solutions for Small Stars are Almost Parsimonious

A  $k$ -star is a tree with  $k$  leafs and one internal vertex. Here we show that the most likely ancestral sequence in 3-stars and 4-stars either are parsimonious or one of the leaf sequences. We also show that this is not the case in 5-stars. This result is used in our heuristic improvement algorithm in Section 4.4 and also in our improved FPT algorithm in Section 3.2. The results for 4-stars and 5-stars have been deferred to the appendix.

**Lemma 3.3 (AML for 3-star).** *In a 3-star the most likely ancestral sequence is either the parsimonious sequence or one of the leaf labels.*

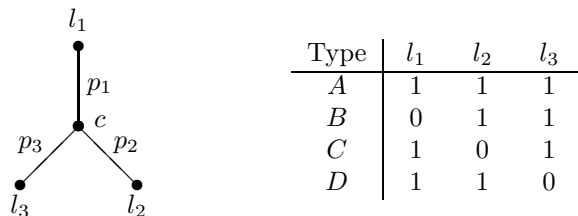


Figure 1: The 3-star with leaf labels  $l_1$ ,  $l_2$ , and  $l_3$ , and mutation probabilities  $p_1$ ,  $p_2$ , and  $p_3$ . The table to the right shows the four possible splits of the taxa.

*Proof.* As in Figure 1 let  $l_1$ ,  $l_2$ , and  $l_3$  be the three leaf labels. The first observation we make is that if all three leafs agree in a site then the most likely ancestral sequence has the same symbol. In terms of the split types in the table this can be reformulated as: In every site of type A the most likely ancestral sequence has the same symbol as the leaf sequences. In other words, the ancestral sequence is parsimonious in all sites of type A.

Assume that the most likely ancestral sequence is non-parsimonious. Further, assume, without loss of generality, that it is non-parsimonious in sites of type B. We show that if this is the case, then it is necessarily parsimonious in all sites of type C and D.

Let  $p_1$ ,  $p_2$ , and  $p_3$  be the mutation probabilities. Then the likelihood of sites of type B is  $(1 - p_1)p_2p_3$  and this is higher than  $p_1(1 - p_2)(1 - p_3)$ , which is the likelihood for the parsimonious solution in such sites. Consequently,  $p_1 < p_2$  and  $p_1 < p_3$ , otherwise the likelihood would be higher if sites of type B are given the parsimonious symbol. Thus, sites of type C and D have to be parsimonious; if they would not be either  $p_2 < p_1$  or  $p_3 < p_1$  which is not possible.

Since the most likely ancestral sequence is non-parsimonious only in sites of type B, it agrees with the label  $l_1$ , i.e.  $p_1 = 0$ . Thus, we have shown that  $c$  is either the parsimonious solution,  $l_1$ ,  $l_2$ , or  $l_3$ . □

In particular, the 3-star lemma states that if sites of type  $B$  are non-parsimonious then  $p_1 = 0$ ,  $p_2 = (\#B + \#C)/m$ , and  $p_3 = (\#B + \#D)/m$ . Hence, due to Corollary 3.2 and the 3-star lemma each 3-star in an AML solution is either parsimonious or has an edge with mutation probability 0. The result is the following corollary:

**Corollary 3.4.** *If there is no edge with mutation probability 0 in the AML solution, then the AML solution is the most parsimonious solution of highest likelihood.*

### 3.2 Fixed-Parameter Tractability in the Number of Taxa

Since AML is NP-hard, we cannot hope to find a practical algorithm for solving big instances of the problem. The point with FPT algorithms is to not give up, but instead focus on solving small instances. The FPT we present here is fixed-parameter tractable in the number of taxa. This means that it has a running time  $O(f(n) \cdot m^c)$ , i.e., it is exponential in the number of taxa, but only polynomial in the length of the sequences. With this algorithm we can solve AML for small number of taxa no matter how long the input sequences are.

Consider the small AML problem with  $n$  taxa. By Lemma 3.1, in any optimal solution equal splits have equal evolution. Altogether there cannot be more than  $2^{n-1}$  splits of the taxa, no matter how long the sequences are. Moreover, since there are  $n-2$  internal vertices, there are  $2^{n-2}$  possible assignments for each split. Therefore, the total number of possible optimal solutions is no more than  $(2^{n-2})^{2^{n-1}}$ . Moreover, since each internal vertex have  $2^{2^{n-1}}$  possible labels, using bottom up dynamic programming the FPT can be improved to  $O(n \cdot 2^{2^n})$ .

The exponential part of the FPT grows very fast and in practice this means that it becomes difficult to solve instances with 5 taxa. However, according to Lemma 3.3 every 3-star in an optimal labelling is either parsimonious or one of the leaf sequences of the star. This can be used to our advantage! As an example, consider the tree with five taxa in Figure 8. Without improvements the FPT has to, in worst case, perform an exhaustive search over all  $5 \cdot 2^{2^5} = 5 \cdot 2^{32}$  labellings. However, according to Lemma 3.3 once the label at vertex  $I_2$  has been decided the labels at  $I_1$  and  $I_3$  can easily be deduced by checking the four different possibilities for each of them. Altogether the second approach need not check more than  $8 \cdot 2^{2^4} = 2^{17}$  labellings; the  $2^{2^4}$  different labels of  $I_2$  and four alternatives each for  $I_1$  and  $I_3$ .

The arguments above resulted in an FPT for small AML. However, by merely applying the FPT for small AML on all possible binary trees of  $n$  taxa we get an FPT also for the big version of AML.

## 4 Approximation Algorithms

In this section, we give a 2-approximation for AML and a PTAS for small AML. These algorithms are based on approximation algorithms for other Steiner tree problems and the super additivity of the log-likelihood function. Moreover, we give some bounds on the log-likelihood score. We start with these bounds.

### 4.1 Bounding the AML Score with the Parsimony Score and Consistency

Here the parsimony score is used to give an upper and a lower bound for the AML score of a phylogenetic tree. We prefer to deal with the normalized log-likelihood function which is the same function up to a multiplicative factor  $m$ . It is defined as  $\log\text{LN}(T, \hat{\lambda}) = \sum_{e \in E(T)} \log\text{LN}(h(e))$ , where  $\log\text{LN}(h(e))$  is the normalized log-likelihood of edge  $e$ :  $\log\text{LN}(h(e)) = \frac{h(e)}{m} \log\left(\frac{h(e)}{m}\right) + \left(1 - \frac{h(e)}{m}\right) \log\left(1 - \frac{h(e)}{m}\right)$ . We prefer this function, since it is easy to prove that the normalized log-likelihood of an edge is a convex function (see Figure 2).

**Lemma 4.1.**  $x \log(x) + (1-x) \log(1-x)$  is a convex function in the interval  $(0, \frac{1}{2})$ .

*Proof.* The second derivative is  $\frac{1}{x} + \frac{1}{1-x}$  and it is positive in the interval. Thus, by definition (Cover and Thomas (1991)) the function is convex.  $\square$

We now prove that the log-likelihood function is a super additive function. This lemma is very central to our approximation algorithms.

**Lemma 4.2.** *Let  $f(x) = x \log(x) + (1-x) \log(1-x)$ . Then  $f(x_1) + \dots + f(x_n) \leq f(\sum_i x_i)$  for  $\sum_i x_i < 1/2$ .*

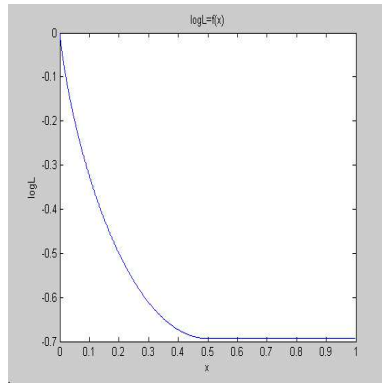


Figure 2: The normalized log-likelihood function for an edge,  $\max_p \frac{h(e)}{m} \log(p) + (1 - \frac{h(e)}{m}) \log(1 - p)$  in the interval  $0 < \frac{h(e)}{m} < 1$ , when the edge probability is restricted as  $p < \frac{1}{2}$ .

*Proof.* By Lemma 4.1, we know that  $f$  is negative in the interval,  $f'$  is negative, and  $f''$  is positive (see Figure 2). Assume w.l.o.g. that  $0 < x_1 < \dots < x_n$ . We start by showing that  $f(x_1) + f(x_2) < f(x_1 + x_2)$ . Since  $f$  has a continuous second derivative,  $f'' > 0$ , and since  $f(0) = 0$  we have

$$\frac{f(x_1)}{x_1} < \frac{f(x_2)}{x_2} < \frac{f(x_1 + x_2)}{x_1 + x_2}.$$

Consequently,  $f(x_1) < \frac{x_1}{x_2} f(x_2)$  and  $f(x_2) < \frac{x_2}{x_1 + x_2} f(x_1 + x_2)$ . Thus,

$$f(x_1) + f(x_2) < \left(\frac{x_1}{x_2} + 1\right) f(x_2) < \left(\frac{x_1}{x_2} + 1\right) \frac{x_2}{x_1 + x_2} f(x_1 + x_2) = f(x_1 + x_2).$$

It is an easy matter to extend the proof by induction. □

We denote the parsimony score of  $(T, \lambda)$  by  $h(T)$  ( $\lambda$  is implicit), i.e., the least number of mutations needed to explain the sequences at the leafs. The following lemma states that the best possible AML solution is one in which all  $h(T)$  mutations occur on one single edge; all other edges have mutation probability zero. Moreover, it states that the likelihood is bounded from below by the solution in which the  $h(T)$  mutations are evenly distributed over the edges in the tree.

**Lemma 4.3.** *The likelihood of the AML solution for a phylogenetic tree  $(T, \lambda)$  is bounded by*

$$|E(T)| \cdot \log \text{LN} \left( \frac{h(T)}{|E(T)|} \right) \leq \max_{\hat{\lambda}} \log \text{LN}(T, \hat{\lambda}) \leq \log \text{LN}(h(T)).$$

*Proof.* First note that

$$|E(T)| \cdot \log \text{LN} \left( \frac{h(T)}{|E(T)|} \right) \leq \sum_{e \in E(T)} \log \text{LN}(\text{hp}(e)),$$

where  $\text{hp}(e)$  denotes the number of mutations over edge  $e$  in a most parsimonious solution. This equation holds by Jensen's inequality for convex functions, which states that if  $f(x)$  is convex and  $\lambda_1 + \dots + \lambda_n = 1$ ,  $\lambda_i > 0$  then

$$f(\lambda_1 x_1 + \dots + \lambda_n x_n) \leq \lambda_1 f(x_1) + \dots + \lambda_n f(x_n).$$

Since moreover no parsimonious solution can be more likely than the most likely ancestral labelling, i.e.,

$$\sum_{e \in E(T)} \log \text{LN}(\text{hp}(e)) \leq \max_{\hat{\lambda}} \log \text{LN}(T, \hat{\lambda}),$$

the left inequality of the lemma holds.

Furthermore,

$$\max_{\hat{\lambda}} \log \text{LN}(T, \hat{\lambda}) = \sum_{e \in E(T)} \log \text{LN}(h\lambda^*(e)) \leq \log \text{LN}\left(\sum_{e \in E(T)} h\lambda^*(e)\right)$$

where  $h\lambda^*(e)$  is the number of mutations over edge  $e$  in the most likely labelling and the right inequality follows from Lemma 4.2. Since  $\sum_{e \in E(T)} h\lambda^*(e) \geq h(T)$ , the right inequality of the lemma holds trivially.  $\square$

Maximum Likelihood is known to be a consistent estimator of phylogenetic trees, while Maximum Parsimony is known as an inconsistent estimator (Felsenstein (1978)). Lemma 4.3 provides the intuition why AML is not a good criterion for estimating phylogenetic trees, though it is the best estimator of ancestral sequences. According to the lemma the AML score may be higher for labellings with some edges having mutation probability 0 and some with higher probability, compared to the parsimonious solution. The presence of zero edges in the AML solution makes it an inconsistent estimator of phylogenetic trees. That is, there are examples for which AML cannot be used to find the "true" tree even if infinitely long sequences are sampled from the tree. In the appendix we provide such an example.

**Observation 4.1.** *Ancestral Maximum Likelihood is an inconsistent estimator of phylogenetic trees.*

## 4.2 The Most Likely Spanning Tree - A 2-approximation for AML

It is common knowledge that the minimum spanning tree is a 2-approximation for minimum Steiner tree problems in metric spaces, e.g. maximum parsimony and tree alignment. Here we show that the most likely spanning tree (MLST) is a 2-approximation for AML. However, in the current setting the sequences are not from a metric space. Instead this result is based on the super additivity of the log-likelihood function, Lemma 4.2.

Both ML and AML are known to be NP-hard (Chor and Tuller (2005); Addario-Berry *et al.* (2004)), but no algorithms with guaranteed performance have been given. Unfortunately, the most likely spanning tree only provides a 2-approximation for AML. This is because the ML function is non-additive.

**Lemma 4.4.** *If  $(T^*, \lambda^*)$  is the optimal AML solution then there is a spanning tree  $(T^S, \lambda^S)$  such that*

$$2 \log(L(T^*, \lambda^*)) \leq \log(L(T^S, \lambda^S)) \leq \log(L(T^*, \lambda^*)),$$

where  $\lambda^S$  is the leaf labeling and  $\lambda^*$  the most likely labelling.

*Proof.* First note that the right inequality holds trivially. As in Figure 3A, order the leaves of  $T^*$  as  $l_1, \dots, l_n$  and consider the counterclockwise walk along the edges. In this walk, each edge is followed twice and therefore the log-likelihood of the walk, i.e. the sum of likelihoods for the edges along the walk, is  $2 \log(L(T^*, \lambda^*))$ . As in the figure, partition the walk into  $n$  paths as follows:

$$P(l_1, l_2) \rightarrow \dots \rightarrow P(l_{n-1}, l_n) \rightarrow P(l_n, l_1),$$

where  $P(l_x, l_y)$  denotes the path between  $l_x$  and  $l_y$ . Now consider the spanning tree  $T^S$  which connects successive leaves with an edge, i.e.  $l_i$  with  $l_{i+1}$  etc.. By super additivity, Lemma 4.2, the log-likelihood of the path  $P(l_i, l_{i+1})$  in  $T^*$  is less than the log-likelihood of the edge between  $l_i$  and  $l_{i+1}$  in  $T^S$ . Thus,

$$2 \log(L(T^*, \lambda^*)) \leq L(T^S, \lambda^S).$$

Although this spanning tree is not binary it can easily be converted into one by adding edges of length 0.  $\square$

Note that the spanning tree in the proof above cannot actually be constructed, unless we have the optimal tree. The most likely spanning tree can however easily be computed in polynomial time and is guaranteed to be within a factor 2 of the optimal solution. Further, it should be possible to improve the 2-approximation by using our FPT together with standard techniques for Steiner tree problems in metric spaces.

**Corollary 4.5.** *The most likely spanning tree is a 2-approximation of AML.*

One drawback of an MLST tree is that some vertices may be of high degree. Since our heuristic improvement algorithm, **ImprovedAML**, requires a binary tree as input it is important that a binary tree is constructed. As was described in the proof above, high degree vertices can be resolved arbitrarily without affecting the likelihood by adding edges of length 0. However, there are better ways of resolving the high degree vertices. Since the MLST tree have sequences assigned to each vertex, it is possible to perform the resolution of non-binary vertices using any other method that provides fully resolved trees; we chose to use the NJ algorithm.

### 4.3 A Polynomial Time Approximation Scheme for Small AML

In (Wang *et al.* (1996)), a PTAS is given for tree alignment. The idea of this algorithm is to, in polynomial time, optimally solve fixed sized subproblems and then combine these into a solution for the original problem. The PTAS we provide is based on the PTAS of Wang *et al.*, our FPT, and Lemma 4.2.

The concepts of the PTAS are defined on *rooted trees* but unrooted trees can be rooted on any arbitrary edge. At the first stage of the PTAS a full labelling is constructed in which each vertex is assigned the same label as one of its descendant leaves. Formally such a labelling  $\hat{\lambda}$  is called a *leaf lifting* and defined as

$$\forall v \in V(T) \exists l \in D(v) \text{ s.t. } \hat{\lambda}(v) = \lambda(l),$$

where  $D(v)$  is the set of descendant leaves of  $v$ . In other words, the labels at the leafs are lifted up into ancestral vertices. In Lemma 4.6 and Corollary 4.7, below, it is shown that the optimal leaf lifting is guaranteed to be a 2-approximation. Furthermore, computing the most likely leaf labelling is easily done in polynomial time with bottom-up dynamic programming in the tree.

As can be seen in the proof of Lemma 4.6, the super additivity is used to bound the likelihood of an edge in the lifted labelling with the likelihood of a path in the optimal tree. In fact, each edge in the lifted labelling is bounded by a unique path in the optimal tree. When all the paths bounding edges are combined into a walk then this walk traverses each edge in the optimal tree twice, which yields the 2-approximation ratio.

Using the FPT and super additivity it is straightforward to modify the proof of Wang *et al.*. Therefore, we choose to show the 2-approximation ratio which is a succinct way of conveying the line of reasoning in the proof of the PTAS. After the 2-approximation we discuss the proof of the PTAS in broad terms and the arguments needed to modify the proof of Wang *et al.*

**Lemma 4.6.** *For any arbitrary rooting of the binary tree  $T$  there is a leaf lifting  $\hat{\lambda}$  such that*

$$2 \log(L(T, \lambda^*)) \leq \log(L(T, \hat{\lambda})) \leq \log(L(T, \lambda^*)),$$

where  $\lambda^*$  is a most likely labelling.

*Proof.* First note that the right inequality holds trivially. Let  $\lambda^*$  be the optimal labelling of the phylogeny. Then the log-likelihood of the labelling is given by

$$\log(L(T, \lambda^*)) = \sum_{e \in E(T)} \log(L(h(e))),$$

i.e. the sum of log-likelihoods of all edges. As in the Figure 3A, order the leafs as  $l_1, \dots, l_n$  and consider the counterclockwise walk along the edges. In the walk each edge of the tree is followed twice and therefore the log-likelihood of this walk is  $2 \log(L(T, \lambda^*))$ . Further, partition the walk  $W$  into  $n$  paths as follows:

$$W = P(l_1, l_2) \rightarrow \dots \rightarrow P(l_{n-1}, l_n) \rightarrow P(l_n, l_1).$$

We now show that there is a leaf lifting  $\hat{\lambda}$  with likelihood at least as good as that of the walk, i.e.,

$$2 \cdot \log(L(T, \lambda^*)) \leq \log(L(T, \hat{\lambda})).$$

Without loss of generality, we infer a root somewhere on the path between  $l_1$  and  $l_n$ . In this rooted tree with labelling  $\lambda^*$  we define the *closest descendant leaf* of a vertex, denoted  $l(v)$ , as the descendant leaf to which the path from the vertex is most likely. Thereafter, we create the leaf lifting  $\hat{\lambda}$  by labelling each internal vertex by the label of its closest descendant leaf.

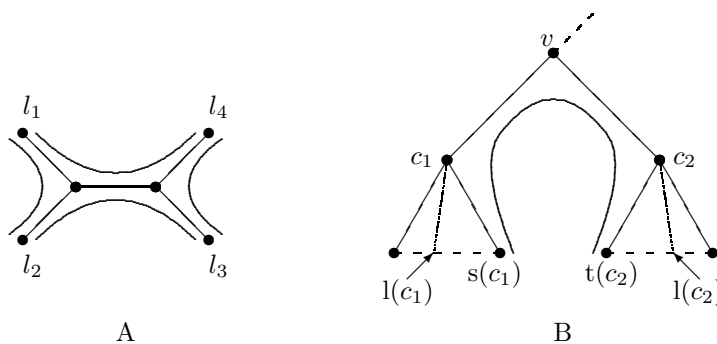


Figure 3: (A) The counterclockwise walk along the edges of the tree. (B) The path  $P(s(c_1), t(c_2))$ .

Consider an arbitrary internal vertex  $v$  in the tree, and let  $c_1$  and  $c_2$  be the children of that vertex. As each vertex is given the label of its closest descendant leaf, the total log-likelihood of the edges  $(v, c_1)$  and  $(v, c_2)$  is

$$\log \left( L(h(l(v), l(c_1))) \right) + \log \left( L(h(l(v), l(c_2))) \right),$$

where  $h$  is the hamming distance. Let  $s(x)$  and  $t(x)$  denote the rightmost and leftmost descendant leaf of vertex  $x$ , respectively. The idea now is to show that the log-likelihood of path, depicted in Figure 3B, from  $s(c_1)$  to  $t(c_2)$  when the tree is labeled by  $\lambda^*$  is upper bounded by the log-likelihood of the edges  $(v, c_1)$  and  $(v, c_2)$  when the vertices are labeled with  $\hat{\lambda}$ . Since  $l(v) = l(c_1)$  or  $l(v) = l(c_2)$  and the two cases are symmetric, we show:

$$\begin{aligned} & \log \left( L(P^{\lambda^*}(s(c_1), t(c_2))) \right) \\ & \leq \log \left( L(P^{\lambda^*}(l(v), s(c_1))) \right) \quad \text{if } l(v) = l(c_2) \\ & \leq \log \left( L(P^{\lambda^*}(l(v), l(c_1))) \right) \quad \text{Def. } l(c_1) \\ & \leq \log \left( L(h(l(v), l(c_1))) \right) \quad \text{Lemma 4.2} \end{aligned}$$

where the first two inequalities are from the definition of closest descendant leaf and the third from Lemma 4.2. Finally, the walk from  $l_1$  to  $l_n$ ,  $P(l_1, l_2) \rightarrow \dots \rightarrow P(l_{n-1}, l_n)$ , is made up by all such paths for each internal vertex. Thus, the log-likelihood of the leaf lifting is

$$\begin{aligned} \log(L(T, \hat{\lambda})) &= \sum_{v \in I(T)} \sum_{c_x = c_1, c_2} \log \left( L(h(l(v), l(c_x))) \right) \geq \log(L(W, \lambda^*)) \\ &= 2 \log(L(T, \lambda^*)). \end{aligned}$$

□

**Corollary 4.7.** *The most likely leaf labelling is a 2-approximation for small AML.*

### 4.3.1 A Polynomial Time Approximation Scheme

The general idea of the PTAS is to extend the leaf lifting technique with local optimization of fixed sized subtrees of height  $t$ . More specifically, the PTAS constructs a labelling in which the vertices on every  $t$ 'th level are assigned labels of descendant leaves and all remaining nodes are labeled optimally. That is, the tree is divided into fixed size subtrees of height  $t$ , called *tiles*, that are internally disjoint but may have overlapping leaves and roots. We refer to such divisions as *tilings*. Furthermore, we let the tiles at the bottom and the single tile at the top be of height  $< t$ . Consequently, there are  $t$  different tilings. We call a tile in which the leaves and the root have been assigned labels of descendant leaves for *components*. The PTAS computes a full labeling of the tree which is built up by an optimal combination of such components<sup>1</sup>. We denote a tiled labelling in which the bottom components are of height  $i$  with  $\lambda_t^i$ .

Here we assume w.l.o.g. that the tree is a complete binary tree, since the general case can be reduced to this case. Let  $g = 2^{t-1} + 1$  represent the number of leafs in a tile (including the root) and let  $n$  be the number of leaf labels. The PTAS works as follows:

<sup>1</sup>If the top component in the tree has height  $< t$  its root is not assigned a leaf label.

1. For  $i = 0 \dots t - 1$ 
  - (a) Construct all  $n^g$  possible components of a tiling in which the bottom components are of height  $i$ .
  - (b) Compute the likelihood and the optimal labelling of each such component with the FPT.
  - (c) Using bottom-up dynamic programming construct an optimal labelling  $\lambda_t^i$  of the tree using the components, where the components at the bottom are of height  $i$ .
2. Return  $\max_i \lambda_t^i$

It is not difficult to see that the running time of the PTAS is  $O(t(n^g \cdot g^{2^{2^g}} + n \cdot n^g \cdot n^g))$ . Next we give arguments that show that the PTAS, for each  $t > 1$ , is guaranteed to return a solution which is within a factor  $1 + 3/t$  of the optimal solution.

Consider a most likely labelling  $\lambda^*$  of the tree. We show that there is a labelling  $\lambda_t^i$  such that the internal vertices of the components are labeled by  $\lambda^*$  and such that the likelihood is within a factor  $1 + 3/t$  of the optimal solution. Note that compared to the labelling constructed by the PTAS each component is not labeled optimally and therefore the labelling returned by the PTAS has at least as high a likelihood.

We now give the arguments needed to modify the proof of (Wang *et al.* (1996)) for tree alignment. In tree alignment, the input is a phylogenetic tree  $T$  with leaf labeling function  $\rho$  assigning sequences to the leaves. Further, the objective is to find a full labeling  $\hat{\rho}$  minimizing:

$$\min_{\hat{\rho}} \sum_{(u,v) \in E(T)} d(\hat{\rho}(u), \hat{\rho}(v)),$$

where  $d$  is a string metric. Wang *et al.* define closest descendant leaf with respect to the string metric and the optimal labelling  $\rho^*$  of the tree. Further, they prove that there is a labelling  $\rho_t^i$  with components of height  $t$  such that the internal vertices of the components are labeled by  $\rho^*$  and such that the objective of the labelling is no more than  $1 + 3/t$  times the optimal. The proof of Wang *et al.* only relies on the triangle inequality of the string metric. Note that by identifying  $\rho^*$  as  $\lambda^*$  and  $d$  with the log-likelihood function this becomes equivalent to Small AML. That is, minimizing the sum of distance in tree alignment and using the triangle inequality is equivalent to maximizing the likelihood and using the super additive bound of Lemma 4.2.

The proof of Wang *et al.* is divided into four lemmas. Only one of these lemmas relies on the triangle inequality and need to be changed to serve our purpose. Below we describe these lemmas and thereafter we give an intuitive explanation of the proof.

- Using the triangle inequality, Lemma 4 of (Wang *et al.* (1996)) bounds the cost of each component in  $\rho_t^i$  by the cost of the associated tile in the optimal labelling and the cost of paths to the closest descendant leafs of each of the leaf labeled vertices. To modify the proof we simply need to replace the use of the triangle inequality with Lemma 4.2.
- Lemma 5 of (Wang *et al.* (1996)) states that there is a one-to-one mapping from internal vertices down to leafs such that the paths from the internal vertices down to their mapped leafs are disjoint. This lemma is only based on algebraic verification and does not need to be changed to serve our purpose.
- Lemma 6 and Lemma 7 of (Wang *et al.* (1996)) states the existence of a labelling  $\rho_t$  with approximation ratio  $1 + 3/t$ . These lemmas depend only on the definition of closest descendant leaf and need not be changed to serve our purpose.

Intuitively the  $1 + 3/t$  approximation ratio can be explained as follows. Altogether there are  $t$  different tiled labellings,  $\rho_t^0, \dots, \rho_t^{t-1}$ , and each of them assigns leaf labels to different vertices. Since all other vertices are labeled with  $\rho^*$ , only the edges incident to vertices that have been assigned leaf labels cost more than in the optimal labelling. Lemma 4 of (Wang *et al.* (1996)) bounds the cost of each component in  $\rho_t^i$  by the cost of the associated tile in the optimal labelling and the cost of paths to the closest descendant leafs of each leaf labeled vertex. Since each vertex has three neighbors, the cost of the edges incident to leaf labeled vertices increases by no more than three times the cost of the path to the closest descendant leaf. Furthermore, since each internal vertex has a unique path down to a leaf, the cost of all  $t$  labellings is  $\leq (t + 3) \cdot \sum_{(u,v) \in E(T)} d(\rho^*(u), \rho^*(v))$ . Consequently, as desired, one of the  $\rho_t^i$  has cost  $\leq (1 + 3/t) \sum_{(u,v) \in E(T)} d(\rho^*(u), \rho^*(v))$ .

**Theorem 4.8.** *Small Ancestral Maximum Likelihood has a PTAS.*

## 4.4 A Local Improvement Heuristic

The solutions provided by our approximation algorithms above have guaranteed performance ratios. This however does not rule out that there are simple ways of improving the solutions. Here we describe a local improvement heuristic which takes a full labelling and improves it iteratively using the FPT.

### ImprovedAML

Input:  $T, \hat{\lambda}, k$ .

1. For each binary subtree  $T'$  of with  $k$  leaves,
  - Find the most likely ancestral sequence of  $T'$  using the FPT.
2. Continue until no improvement is made.

In its most simple form, the improvement algorithm above is used for  $k = 3$ , i.e., each binary subtree is a 3-star. In this case, by Lemma 3.3, there are only 4 sequences to check and the algorithm is very fast. However, as in the PTAS case, the larger  $k$  we use the slower the algorithm gets. In the experimental section, we use the improvement heuristic to improve solutions provided by the approximation algorithms and the parsimonious solution.

## 5 Extending Results to the Jukes-Cantor Model

In this section, we extend our results to the Jukes-Cantor (JC) model (Jukes and Cantor (1969)). The JC model is similar to CFN model, but can deal with more than two characters. The JC model can be formalized for alphabets of any size  $c$  (for DNA sequences,  $c = 4$ ; for proteins,  $c = 20$ ). It assumes that all sites evolve independently and identically according to a Markov model with  $c$  states, each representing a nucleotide. The Markov model is symmetric and all state transitions are equally likely. Therefore, for our purposes, it suffices to associate each edge  $e$  with a mutation probability  $p(e)$ . This means that the probability of no mutation is  $1 - p(e)$  and the probability of observing any specific mutation, e.g.  $A \rightarrow G$ , is given by  $p(e)/(c - 1)$ . Hence, the likelihood of observing a fully labeled tree  $(T, \hat{\lambda}, p)$  is given by

$$L(T, \hat{\lambda}, p) = \prod_{e \in E(T)} \left( \frac{p(e)}{c - 1} \right)^{h(e)} (1 - p(e))^{m - h(e)}$$

We start by arguing that the log-likelihood function in the JC model is convex. As in the proof of Lemma 4.1, it is a simple matter to prove that the second derivative of the normalized log-likelihood is positive. Similarly it is easy to see that function is super additive. Moreover, since Lemma 4.3 only relies on the convexity and super additivity of the function the bounds hold also in the JC model.

### 5.1 Extending the Results for Small Stars to the JC model

Recall that Lemma 3.3 states that the most likely ancestral sequence in a 3-star either is one of the leaf sequences or the most likely parsimonious sequence. The proof in the JC model is almost identical; the only difference is that there may be more than one parsimonious sequence. Therefore, we show how the most likely parsimonious sequence can be found.

Let the three leaf sequences be  $l_1, l_2$ , and  $l_3$ . Moreover, assume that there are  $x$  positions in which the parsimonious choice is not unique, i.e., the three leaves all have different nucleotides in these  $x$  positions. The other  $m - x$  positions have a unique parsimonious choice. Let  $h'_{l_1}, h'_{l_2}$ , and  $h'_{l_3}$  represent the Hamming distance for each of the three leaf sequences to the parsimonious sequence in the  $m - x$  parsimonious positions. Assume further, w.l.o.g., that  $h'_{l_1} = \min(h'_{l_1}, h'_{l_2}, h'_{l_3})$ . By super additivity of the log-likelihood, Lemma 4.2, the most likely parsimonious sequences agrees with  $l_1$  in all of the  $x$  positions for which the parsimonious choice is not unique.

## 5.2 Generalizing the FPT and the Approximation Algorithms to the JC model

Our FPT is based on Lemma 3.1, i.e., if two positions  $i$  and  $j$  describe the same bipartition of the sequences then mutations in these positions occur on the same edges. In the JC model, as in the CFN model, every state transition in the Markov model is equally likely. Therefore, the equal evolution lemma can be generalized by simply noting that each position partitions the sequences into  $c$  parts. However, since the FPT proceeds by for each occurring partition checking all possible assignments to the internal vertices the complexity of the FPT increases. Thus the run time complexity of the FPT in the Jukes-Cantor model is  $O(n \cdot c^n)$ . The case of more than four characters is similar.

Our approximation algorithms depend on two properties: the super additivity, Lemma 4.2, and the FPT. We have just argued that both Lemma 4.2 and the FPT can be generalized.

## 6 Experiments on Simulated and Real Biological Data

In this section, we investigate the performances of our algorithms on simulated and real biological data. We start by comparing the different methods on simulated data. Thereafter, we compare our algorithms to the results of (Barry and Hartigan (1987)) by reconstructing the ancestral sequences of a small part of the mitochondrial DNA of Hominidae. Then we use our FPT to reconstruct the ancestral sequences of the complete mitochondrial DNA of Hominidae. We finish by reconstructing the ancestral genomes of a set of HIV related viruses.

### 6.1 Performance on Simulated Data

Although our algorithms have a guaranteed approximation ratio it is interesting to know how well they perform in practice. However, since Big AML is NP-hard it is not possible to compare our solutions with the optimal AML solution. Instead we tested the algorithms on simulated data. That is, we generated data with a probabilistic approach, ran the algorithms on these data, and compared the likelihood of the output to that of the ancestral sequences in the simulated data.

The data was generated as follows:

1. 20 random birth-death trees were generated (Sanderson) of sizes: 5, 10, 15, 20, 30, 40, 50, 60, 70, 80, 90, 100.
2. For each edge the probability of mutation was chosen uniformly at random from the interval  $[0, m]$ . Both a low mutation probability,  $m = 0.1$ , and a high mutation probability was tested,  $m = 0.25$ .
3. Sequences of length 10, 50, 100, 500, 1000, 6000 were then evolved (Rambaut and Grassly (1997)) according to the Jukes-Cantor model from the root down toward the leaves.

The algorithms we tested were:

1. MLST algorithm with NJ resolution, called *BigAML*.
2. The leaf lifting algorithm, called *SmallAML*.
3. A parsimonious solution, called *Parsimony*.

All solutions, including the original ancestral sequences, were improved using the heuristic for  $k = 3$ . Thereafter, the three methods were compared with the original sequences by taking the average normalized difference over the 20 trees of each data size. The results are shown in the plots below.

There are some things to notice. For low mutation probability, i.e. 0-0.1, the parsimonious solution and the original ancestral sequences have almost identical likelihood. Further, the solution provided by the approximation algorithms is close to 2% worse than the parsimonious solution. For high mutation probability, i.e. 0-0.25, the parsimonious solution is 0.5% better than the original ancestral sequences! Moreover, both approximation algorithms are 1-1.5% worse than the original ancestral sequences.

The simulated data indicates that the best algorithm is the parsimony approach together with the local improvement heuristic. In fact, this algorithm outperforms even the original ancestral sequences for  $m = 0.25$ . However, the parsimony approach does not always produce better solutions than the approximation algorithms. For example, in our study of Lentiviruses below the solutions provided by the

approximation algorithms are better than the parsimonious solution. This is probably due to the very high mutation probability in viruses. The Lentiviruses data set provides a biological motivation for the usefulness of the approximation algorithms.

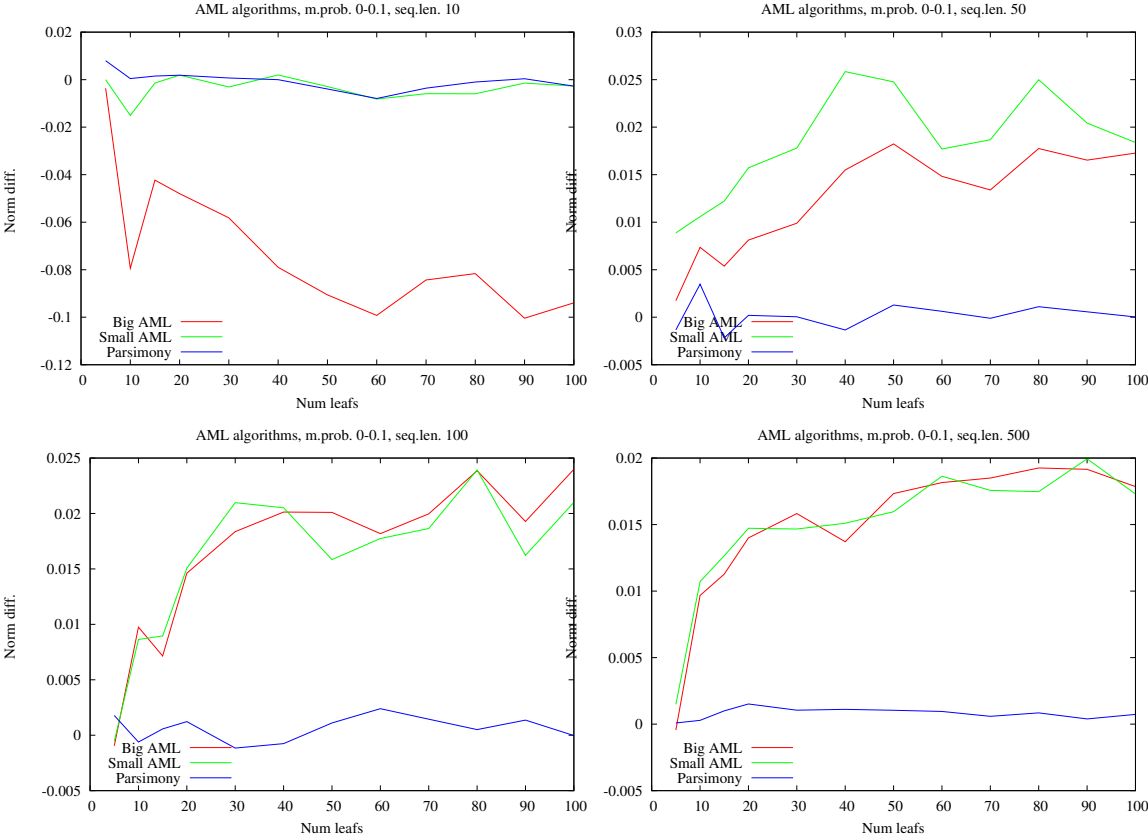


Figure 4: The plots show for different sequence lengths the normalized difference between likelihood of the original ancestral sequences and the solutions provided by the three algorithms. Thus, 0 on the y-axis represent the likelihood of the original sequences. In these plots the mutation probability of each edge was chosen uniformly at random from the interval  $[0 - 0.1]$ .

### 6.2 FPT for the Mitochondrial DNA of Hominidae

Method	no.impr.	w.impr.
FPT	-1704.28	-1689.79
Parsimony	-1704.28	-1699.33
Leaf Lifting	-1882.98	-1729.82

Table 1: The result of applying our methods on *hominidae* subsequence of the Mitochondrial DNA that appear in (Brown *et al.* (1982); Barry and Hartigan (1987)). The FPT algorithm was used to find the optimal solution in the CFN model and thereafter the solution was improved in the four-state JC model. We also checked the parsimony and the leaf lifting methods, with and without the local improvement heuristic (*w.impr.* and *no.impr.* respectively).

Our first dataset includes mitochondrial DNA-sequences of length 896 between two HindIII cleavage sites of five Hominidae: Human, Chimp (*Pan troglodytes*), Gorilla (*Gorilla gorilla*), Orangutan (*Pongo pygmaeus*), and Gibbon (*Hylobates lar*). We used exactly the same sequences that appear in (Brown *et al.* (1982)) and that also was used by (Barry and Hartigan (1987)) (the sequences are available for download

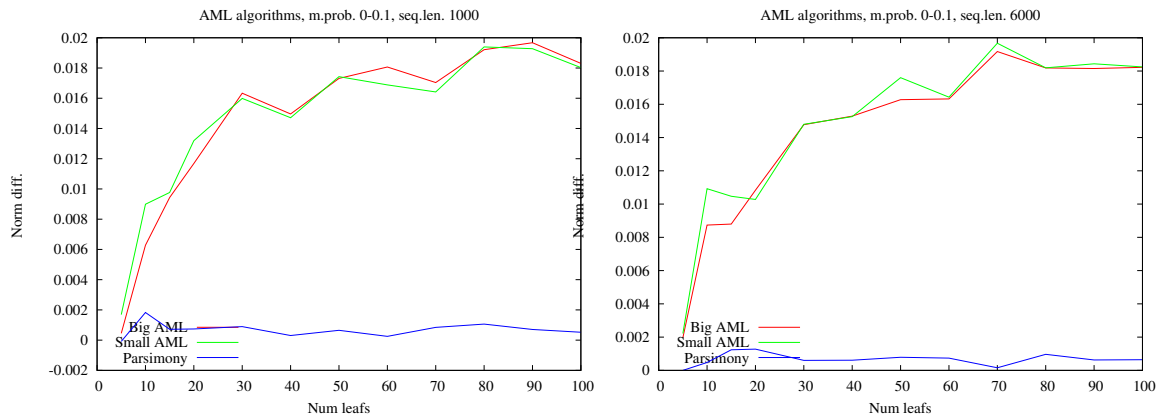


Figure 5: The plots show for different sequence lengths the normalized difference between likelihood of the original ancestral sequences and the solutions provided by the three algorithms. Thus, 0 on the y-axis represent the likelihood of the original sequences. In these plots the mutation probability of each edge was chosen uniformly at random from the interval  $[0 - 0.1]$ .

Method	no.impr.	w.impr.
FPT	-28367.4	-28097.8
Parsimony	-28301.4	-28204.4
Leaf Lifting	-31907.7	-29018.8

Table 2: The result of applying our methods on *hominidae* complete Mitochondrial DNA. The FPT algorithm was run on the two-state model and thereafter the solution was improved in the four-state JC model. We also checked the parsimony and the leaf lifting methods, with and without the local improvement heuristic (*w.impr.* and *no.impr.* respectively).

from our website). Further, we used the accepted tree topology in Figure 8 and reconstructed the ancestral sequences for this input, the resulting likelihoods are listed in Table 1.

In comparison to the work by Barry and Hartigan (Barry and Hartigan (1987)), the sequences reconstructed by our methods have significantly higher likelihood. As mentioned in their paper, Barry and Hartigan used a more general model than the JC model, where each pair of nucleotides may have different transition probabilities. Thus, for a given input the likelihood of the optimal solution in their model should be at least as high as the likelihood of the optimal solution in our model. However, the total log-likelihood of their solution, for 896 long mitochondrial sequences, was  $-2597.5$  while our solutions had a log-likelihood between  $-1882.98$  and  $-1689.79$  (38% - 54% better).

Our second dataset includes the *current* version of the *complete* mitochondrial DNA of the five Hominidae that was mentioned above (the data was downloaded from NCBI). We used CLUSTAL-W for aligning these genomes, and considered only sites without gaps. Out of the 15,834 sites only 18 (about 0.1%) included more than 2 characters. Thus, we believe that the appropriate model of sequence evolution is the CFN two state model. This enable us to apply the FPT for the binary alphabet and get the exact solution. The ancestral symbols in the 18 additional sites were inferred using the edge lengths provided by the FPT solution. The resulting ancestral sequences are available for download from the website. The log-likelihoods of the solutions are listed in Table 2.

We underline, that these datasets demonstrate the importance of using the FPT for inputs with a small number of leafs. Heuristics may very well be far from the optimal solution while the FPT is guaranteed to find the optimal solution.

### 6.3 Reconstructing Ancestral Sequences for Genomes of *Lentiviruses*

Our third dataset includes the genomes of ten viruses (all of them from the *lentiviruses* family). The genomes were downloaded from NCBI, aligned with CLUSTAL-W (Thompson *et al.* (1994)), and further all positions with gaps were removed. We compared the solutions of the parsimony method to that of the

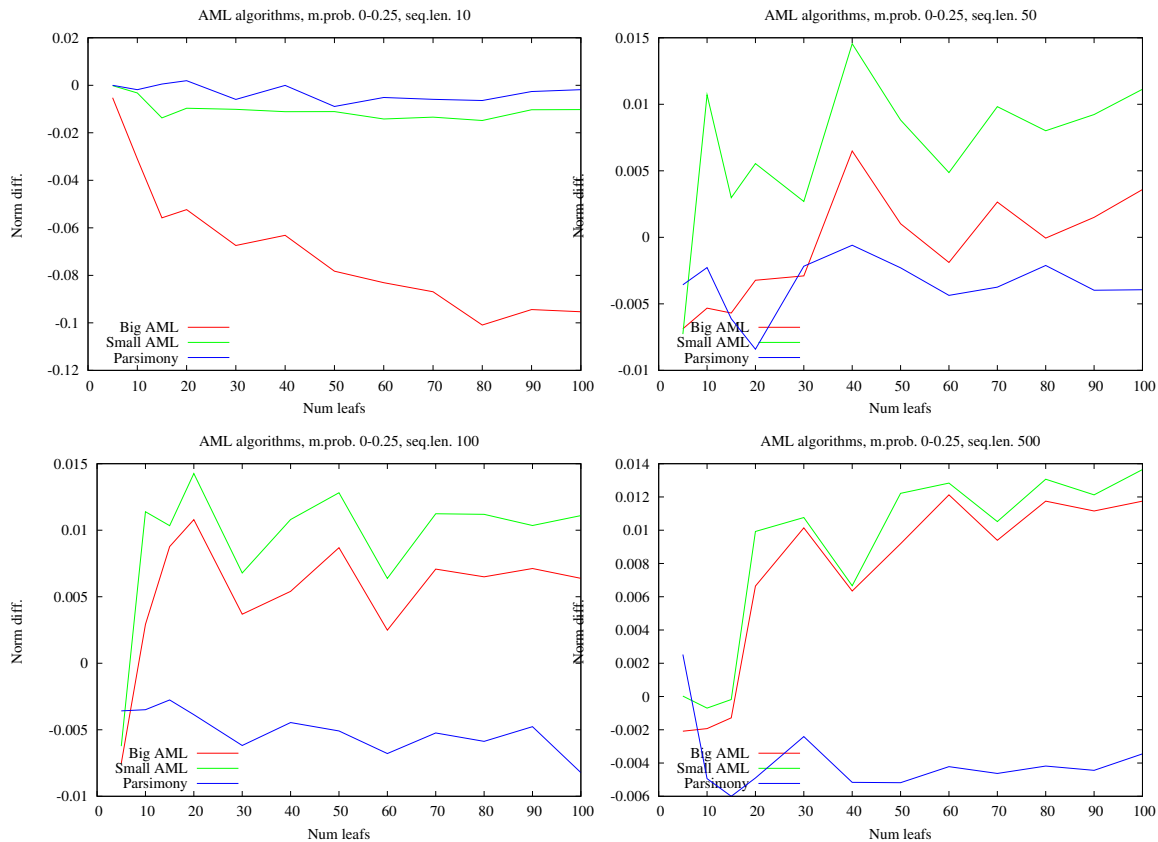


Figure 6: The plots show for different sequence lengths the normalized difference between likelihood of the original ancestral sequences and the solutions provided by the three algorithms. Thus, 0 on the y-axis represent the likelihood of the original sequences. In these plots the mutation probability of each edge was chosen uniformly at random from the interval  $[0 - 0.25]$ . Notice that the solutions provided by bigAML and smallAML are more likely than the original sequences.

leaf lifting method, both with and without the local improvement heuristic. The trees were reconstructed by three methods ACS (Burstein *et al.* (2005)), Neighbor Joining (NJ) and the spanning tree algorithm for Big AML. The NJ tree was created with Kimura’s two-parameter model and the AML algorithms used the Jukes-Cantor model. The results are summarized in Table 3. The genomes, resulting trees, and ancestral sequences can be found on our website.

From the results, it is clear that the approximate leaf lifting method is better than the heuristic parsimony method. The three methods for tree construction gave similar results for leaf lifting. These results can be explained by the very high mutation rate in viruses. Further, since the leaf lifting method constructs a solution in which many labels at internal vertices are the same, the likelihood is the same for many different trees.

## 7 Concluding Remarks and Future Research

This paper is a first step toward developing approximation algorithms and investigating the properties of the AML criterion. We developed approximation, fixed-parameter tractable (FPT), fast heuristic algorithms, and analyzed the solutions of small stars. Further, we demonstrate how our algorithms can be used for analyzing biological data.

Many Steiner tree problems in metric spaces have approximation algorithms that have better approximation guarantees than the best spanning tree. It seems likely that standard techniques for Steiner tree problems in metric spaces together with our FPT can be used to construct an improved approximation algorithm also for Big AML. Furthermore, it should also be possible to improve the running time of the

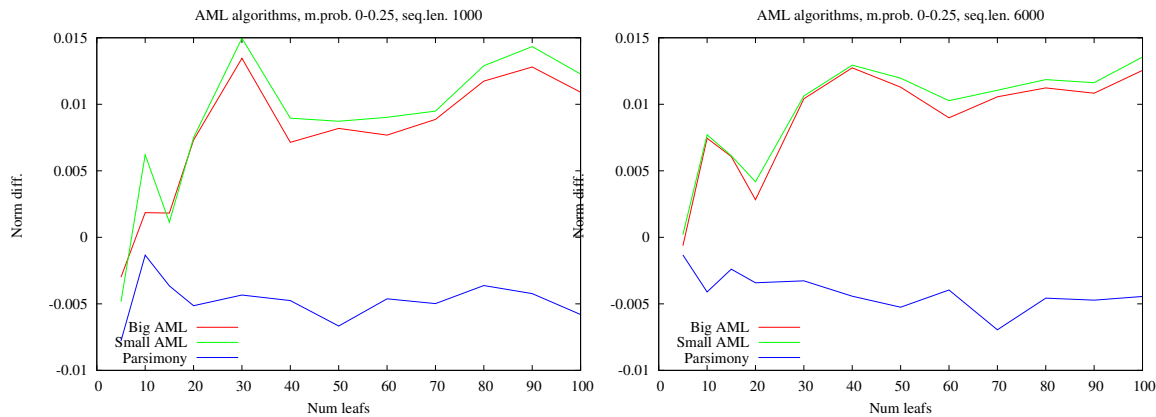


Figure 7: The plots show for different sequence lengths the normalized difference between likelihood of the original ancestral sequences and the solutions provided by the three algorithms. Thus, 0 on the y-axis represent the likelihood of the original sequences. In these plots the mutation probability of each edge was chosen uniformly at random from the interval  $[0 - 0.25]$ . Notice that the solutions provided by the parsimony approach is more likely than the original sequences.

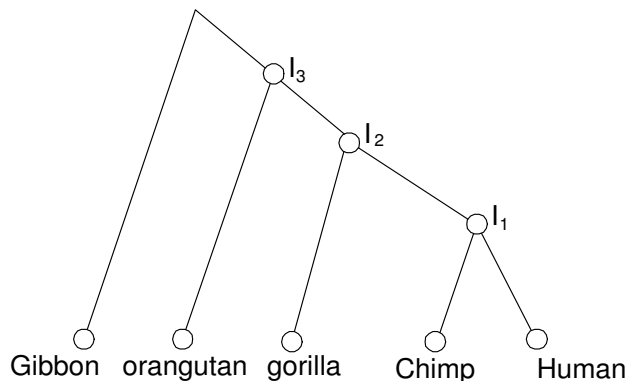


Figure 8: The accepted phylogenetic tree of the Hominidae.

FPT by more extensive use of the 3-star lemma and upper and lower bounds.

It is known that many constant characters implies that the maximum likelihood solution equals the maximum parsimony solution (Tuffley and Steel (1997)). It will be interesting to see if this is the case for AML. We believe so and our simulations support this conjecture. One direction is to use tools similar to those of (Tuffley and Steel (1997)).

ML is a very common criteria for constructing phylogenetic trees. Similar to AML, the complexity of Small ML is still open while Big AML and Big ML are NP-hard (Chor and Tuller (2005); Addario-Berry *et al.* (2004)). Moreover, there are no known approximation algorithms for either Small ML or Big ML. Since AML and ML are similar, we hope that this paper will be a first step toward developing approximation algorithms for ML, and for analyzing the complexity of Small ML. In the future we intend to study these questions for ML.

## Acknowledgments

We would like to thank Jens Lagergren for many valuable comments.

## References

Addario-Berry, L., Chor, B., Hallett, M., Lagergren, J., Panconesi, A., and Wareham, T., 2004. Ancestral maximum likelihood of evolutionary trees is hard. *J. of Bioinf. and Comp. Biology* 2, 257-271.

Method	no.impr.	w.impr.
Parsimony ACS	-68006.5	-67081.9
Parsimony BigAML	-68838.8	-66877.7
Parsimony NJ	-67582.6	-66105.5
Leaf Lifting ACS	-64899.8	-64899.8
Leaf Lifting NJ	-64850.2	-64850.2
Leaf Lifting BigAML	-64756.4	-64756.4
BigAML	-64756.4	-64756.4

Table 3: The result of applying our methods on *lentiviruses*. We checked the parsimony vs leaf lifting method, with and without the local improvement heuristic (*w.impr.* and *no.impr.* respectively). The trees were reconstructed by three methods ACS (Burstein *et al.* (2005)), neighbor joining (NJ) and the algorithm for big ancestral likelihood (BigAML).

- Barry, D. and Hartigan, J., 1987. Statistical analysis of humanoid molecular evolution. *Stat. Sci.* 2, 191–210.
- Brown, W. M., Prager, E. M., Wang, A., and Wilson, A. C., 1982. Mitochondrial dna sequences of primates: Tempo and mode of evolution. *J. Mol Evol* 18, 225–239.
- Burstein, D., Ulitsky, I., Tuller, T., and Chor., B., 2005. Information theoretic approaches to whole genome phylogenomics. *RECOMB 2005* 283.
- Cavender, J., 1978. Taxonomy with confidence. *Math. Biosci.* 40.
- Chor, B. and Tuller, T., 2005. Maximum likelihood of evolutionary trees is hard. *RECOMB* 296–310.
- Cover, T. M. and Thomas, J. A., 1991. *Elements of Information Theory*. J. Wiley and sons, New York.
- Day, W., 1983. Computationally difficult parsimony problems in phylogenetic systematics. *JTB* 103, 429–438.
- Farris, J., 1973. A probability model for inferring evolutionary trees. *Syst. Zool.* 22, 250–256.
- Felsenstein, J., 1978. Cases in which parsimony or compatibility methods will be positively misleading. *Syst. Zool.* 22, 240–249.
- Felsenstein, J., 1993. Phylip (phylogeny inference package) version 3.5c. *Technical report, Department of Genetics, University of Washington, Seattle.* .
- Felsenstein, J., 2003. *Inferring Phylogenies*. Sinauer Associates.
- Fitch, W., 1971. Toward defining the course of evolution: minimum change for a specified tree topology. *Syst. Z.* 20, 406–416.
- Foulds, L. and Graham, R., 1982. The steiner problem in phylogeny is NP-Complete. *Adv. Appl. Math.* 3, 43–49.
- Friedman, N., Ninio, M., Peér, I., and Pupko, T., 2002. A structural em algorithm for phylogenetic inference. *JCB* 9, 331–353.
- Gaschen, B., Taylor, J., Yusim, K., Foley, B., Gao, F., Lang, D., Novitsky, V., Haynes, B., Hahn, B., Bhattacharya, T., and Korber, B., 2002. Diversity considerations in hiv-1 vaccine selection. *Science* 296, 2354–2360.
- Hudek, A. K. and Brown, D. G., 2005. Ancestral sequence alignment under optimal conditions. *BMC Bioinformatics* .
- Jukes, T. H. and Cantor, C. R., 1969. Evolution of protein molecules. In *H. N. Munro, editor, Mammalian protein metabolism* 21–132.
- Koshi, M. and Goldstein, R., 1996. Probabilistic reconstruction of ancestral protein seences. *JME* 42, 313–320.
- Krishnan, N. M., Seligmann, H., Stewart, C., Koning, A. P. J., and Pollock, D. D., 2004. Ancestral sequence reconstruction in primate mitochondrial dna: Compositional bias and effect on functional inference. *MBE* 21, 1871–1883.
- Neyman, J., 1971. Molecular studies of evolution: a source of novel statistical problems. *Statistical Decision Theory and Related Topics* 1–27.

- Pagel, M., 1999. The maximum likelihood approach to reconstructing ancestral character states of discrete characters on phylogenies. *Systematic Biology* 48, 612–622.
- Pupko, T., Peer, I., Shamir, R., and Graur, D., 2000. A fast algorithm for joint reconstruction of ancestral amino acid sequences. *MBE* 17, 890–896.
- Rambaut, A. and Grassly, N., 1997. Seq-gen: An application for the monte carlo simulation of dna sequence evolution along phylogenetic trees. *Comp. Appl. Biosci.* 13, 235–238.
- Sanderson, M., . r8s. [Http://ginger.ucdavis.edu/r8s/](http://ginger.ucdavis.edu/r8s/).
- Steel, M. and Penny, D., 2000. Parsimony, likelihood, and the role of models in molecular phylogenetics. *MBE* 17, 839–850.
- Thompson, J. D., Higgins, D. G., and Gibson, T. J., 1994. Clustal w: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucleic Acids Res.* 22, 46734680.
- Tuffley, C. and Steel, M., 1997. Link between maximum likelihood and maximum parsimony under a simple model of site substitution. *Bulletin of Mathematical Biology* 59, 581–607.
- Wang, L., Jiang, T., and Lawler, E. L., 1996. Approximation algorithms for tree alignment with a given phylogeny. *Algorithmica* 16.
- Yang, Z., Kumar, S., and Nei, M., 1995. A new method of inference of ancestral nucleotide - and amino acid sequences. *Genetics* 141, 1641–1650.

## A AML in 4-stars and 5-stars

In this section, we provide details about the properties of AML solutions of 4-stars and 5-stars.

**Lemma A.1 (AML for 4-star<sup>2</sup>).** *In a 4-star the most likely ancestral sequence is either a parsimonious sequence<sup>3</sup> or one of the leaf labels.*

*Proof sketch.* The proof follows the same line of reasoning as the 3-star lemma. That is, if the ancestral sequence is non-parsimonious in one type of split then the mutation probability on the associated edge has to be smaller than on the other three. In fact, as in the 3-star case, the mutation probability has to be 0. □

**Lemma A.2 (AML for 5-star).** *In a 5-star the most likely ancestral sequence can be a sequence which is neither parsimonious nor one of the leaf sequences.*

*Proof sketch.* In Table 4 we show an example of leaf sequences for which the most likely ancestral sequence is non-parsimonious and not one of the leaf sequences. It is easy to come to the conclusion that an ancestral sequence in the example either is: (1) the all 0 sequences, (2) the  $l_1$  or  $l_2$  sequence, or (3) the sequence  $c$  in which all but the first position is 0. Checking the three examples it can be seen that case (3) is the optimal choice. Moreover,  $c$  is a sequence which is neither parsimonious nor a leaf sequence. □

$l_1$	110	000	000	000
$l_2$	101	000	000	000
$l_3$	000	111	000	000
$l_4$	000	000	111	000
$l_5$	000	000	000	111
$c$	100	000	000	000

Table 4: Example of a most likely ancestral sequence  $c$  in a 5-star which is non-parsimonious and not one of the leaf sequences.

---

<sup>2</sup>The proof of this result cannot be modified to handle alphabets of more than three symbols.

<sup>3</sup>There may be 8 parsimonious ancestral sequences for which equal evolution holds.

## B AML is an Inconsistent Estimator of Trees

Here we give one simple example for which AML cannot be used to distinguish between two different binary trees. The sequences are sampled from the tree in Figure 9A. As was shown in Section 3.1 there are mutation probabilities  $P_1 > 0$ ,  $P_2 > 0$ , and  $P_3 > 0$  such that the AML estimate of  $P_1$  is zero, no matter how long sequences are sampled. Thus, AML cannot distinguish between the two trees in figures 9B and 9C no matter how long sequences are sampled from the tree.

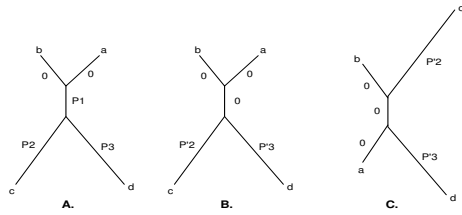


Figure 9: Simple example depicting the inconsistency of AML as an estimator of trees. If the sequences are sampled from the tree in figure A then AML cannot be used to distinguish between the two trees in figures B and C.