An Efficient Algorithm for Statistical Multiple Alignment on Arbitrary Phylogenetic Trees

G.A. Lunter, I. Miklós, Y.S. Song, J. Hein

April 30, 2003

Department of Statistics, University of Oxford,
1 South Parks Road, Oxford, OX1 3TG, UK

Abstract

We present an efficient algorithm for statistical multiple alignment based on the TKF91 model on an arbitrary k-leaved phylogenetic tree. The existing algorithms use a hidden Markov model approach, which requires at least $O(\sqrt{5}^k)$ states and leads to a time complexity of $O(5^k L^k)$, where L is the geometric mean sequence length. Using a combinatorial technique reminiscent of inclusion/exclusion, we are able to sum away the states, thus improving the time complexity to $O(2^k L^k)$ and considerably reducing memory requirements. This makes statistical multiple alignment under the TKF91 model a definite practical possibility in the case of a phylogenetic tree with a modest number of leaves.

1 Introduction

A major way by which biological sequences evolve is mutation. Three main types of mutation events are substitutions, insertions and deletions of amino acids or nucleotides. The latter two events, insertions and deletions, introduce the problem of *aligning* sequences, so that homologous positions appear in the same column of an alignment. When the aim is to find sequences from a database which are homologous to a query sequence, current alignment techniques perform quite well. The alignment problem in phylogenetics, however, is still a serious challenge [Lee01, Gol98]. For example, the phylogenies inferred from different, but equally good, alignments can be quite different [Gol98]. Some researchers have suggested that the regions in the sequences which are responsible for this variance in the inferred phylogeny (usually referred to as *unalignable regions* [Lee01]) should be omitted. It is unclear, however, how much information is lost by ignoring such regions. Moreover, several studies have suggested that accurate evolutionary parameters cannot be obtained using only a single "best" alignment [TKF91, HWK$^+$00], even if this alignment is seemingly reliable [HWK$^+$00]. A more robust approach is to take into account all possible alignments, or a large subset of those, in a statistical framework.

Since insertions and deletions, like substitutions, are rare random events, it seems natural to model them by a continuous time stochastic process, with rates for the three types of mutations. Stochastic models for the substitution process are well known and widely used to obtain maximum-likelihood evolutionary parameters [Fel81, WLG01]. The TKF91 [TKF91] model of sequence evolution incorporates such a mutation model, and moreover allows a nucleotide to spawn new nucleotides adjacent to itself, and to die. To model evolution on a phylogenetic tree, the TKF91 model is applied to each branch of the tree, and likelihood calculation of multiple alignments then becomes possible.

Since the TKF91 model is reversible (and the sequence distribution at the root is assumed to be at equilibrium), the root placement is immaterial. In the case of two sequences, one of the sequences may therefore be considered the ancestral sequence, and the other its descendant. In that framework, the number of possible alignments is finite. When the TKF91 model is applied to a tree, however, an infinite number of sequences may appear at internal nodes, for any given set of sequences which appear at the leaf nodes. Thus, such an extension is not straightforward, and several methods have been proposed. For star-shaped trees, Steel and Hein [SH01] have constructed an algorithm with $O(L^{2k})$ running time for k sequences with geometric mean length L, and it has subsequently been extended, with similar time complexity, to binary trees [Hei01]. Thenceforth, the time complexity has been reduced to $O(4^k L^k)$ for star-shaped trees [Mik02], and to $O(5^k L^k)$ for arbitrary binary trees [HJP02] (see appendix B). In the latter paper, the TKF91 model has been reformulated as a multiple-Hidden Markov Model, in which likelihood calculations can be performed using forward and backward algorithms known in the HMM literature [DEKM98]. Similar ideas have appeared for pairwise and triplewise statistical alignment as well, *e.g.* Holmes and Bruno [HB01] give HMM formulations for alignment on two- and three-leaved trees, and use it for sampling larger trees. In the Markov models there are $O(\sqrt{5}^k)$ states, and therefore the running time of these algorithms is indeed $O(5^k L^k)$, in accordance with the general theory of HMMs [DEKM98]. The memory usage is of the order $O(\sqrt{5}^k L^k)$ if the entire dynamic programming table is retained, and $O(\sqrt{5}^k L^{k-1})$ if not [Hir77].

Because of the large number of states, the aforementioned algorithms are quite slow, even for moderate number of sequences. Hein *et al.* [HWK$^+$00], however, have developed an algorithm for statistical alignment of two sequences which needs neither different states of a HMM nor partition of probabilities into different types of alignments [TKF91]. This is in contrast to the original formulation of the TKF91 model, which uses 3 states. The algorithm of [HWK$^+$00] is, both in time and space complexity, as simple as the traditional distance- or similarity-based dynamic programming algorithms [NW70, SK83]. Henceforward, we refer to it as the *one-state recursion*.

In this paper, we present a one-state recursion for statistical alignment on arbitrary phylogenetic trees. Essentially, it combines the idea in [HWK$^+$00] generalised to trees and Felsenstein's reverse traversal algorithm [Fel81]. The first idea allows us to reduce the number of states to one, and the second to compute in linear time the exponential number

of terms that occur in the transition factors. The resulting algorithm has time complexity $O(2^k L^k)$ and space complexity $O(L^k)$ if the entire dynamic programming table is stored in memory, $O(L^{k-1})$ if not. This represents a great saving in both space and time compared to the hidden Markov recursion, and makes statistical alignment on a phylogenetic tree of modest size a definite practical possibility. Indeed, we have implemented both methods for 3 and 4 sequences, and achieved a considerable acceleration. One of us (IM) was able to perform a likelihood ratio test [Fel81, WLG01] for more than 250 triplets of yeast protein sequences in a day, using triplewise statistical alignments. Moreover, this one-state recursion can be coupled with corner-cutting techniques [HWK$^+$00], which provides further reduction both in time and space complexity.

The organisation of this paper is as follows. We briefly describe the TKF model in §2 and discuss in the following section the one-state recursion in the case of two sequences. The two-sequence example is simple, but well illustrates our general approach. We discuss in §4 our algorithm for the one-state recursion and consider a specific application in §5. The main ideas underlying our proof are sketched in §6. The general one-state recursion is described in §7 and the computation of the transition factors which occur in the recursion is discussed in §8. In §C we draw a connection with a hidden Markov model and describe how optimal alignments can be obtained. We conclude with some remarks and discussion in §9. In appendix, we show that the number of Markov states in the HJP recursion is $O(\sqrt{5}^k)$ and give a proof of Lemma 1 from §7.

2 The TKF model

The TKF91 model is a continuous time reversible Markov model for the evolution of nucleotide (or amino acid) sequences. It models three of the main processes in sequence evolution, namely *substitutions*, *insertions* and *deletions* of nucleotides, approximating these as single-nucleotide processes. A nucleotide sequence is represented by an alternating string of *nucleotides* and *links*, connecting the nucleotides, and this string both starts and terminates with a link. We adopt the view that insertions originate from links, and add a nucleotide-link pair to the *right* of the original link; deletions originate from nucleotides and have the effect of removing the nucleotide and its right link. (This view is slightly different but equivalent to the original description, see [TKF91].) In this way, nucleotide subsequences evolve independently of each other, and the evolution of a nucleotide sequence is the sum of the evolutions of individual nucleotide-link pairs. The leftmost link of the sequence has no corresponding nucleotide to its left, hence it is never deleted, and for this reason it is called the *immortal link*.

Since subsequences evolve independently, it suffices to describe the evolution of a single nucleotide-link pair. In a given time-span τ, this evolves into a sequence of nucleotides of finite length. The first nucleotide of this sequence may be homologous to the original one, while subsequent ones will be non-homologous. Table 1 summarises the corresponding

Fate:	Probability:	Label:
$C \to C\#^{n-1}$	$e^{-\mu\tau}(1-\lambda\beta(\tau))(\lambda\beta(\tau))^{n-1}$	$H_\tau B_\tau^{n-1}$
$C \to \#^n$	$(1-e^{-\mu\tau}-\mu\beta(\tau))(1-\lambda\beta(\tau))(\lambda\beta(\tau))^{n-1}$	$N_\tau B_\tau^{n-1}$
$C \to -$	$\mu\beta(\tau)$	E_τ
$\star \to \star\#^n$	$(1-\lambda\beta(\tau))(\lambda\beta(\tau))^n$	$I_\tau B_\tau^n$

Table 1: Probabilities of the TKF91 model (see text).

probabilities. The parameters λ and μ are the birth rate per link and the death rate per nucleotide, respectively, and in order to have a finite equilibrium sequence length, we require $\lambda < \mu$. For brevity we write

$$\beta(\tau) = \frac{1-e^{(\lambda-\mu)\tau}}{\mu - \lambda e^{(\lambda-\mu)\tau}}.$$

On the right-hand side of the arrow in the column labeled "Fate", C denotes a nucleotide homologous to the original nucleotide, whereas #'s denote non-homologous nucleotides. The immortal link is denoted by $\star$ and other links are suppressed. All final arrangements can be thought of as being built from five basic "processes" which we call Birth, Extinction, Homologous, New (or Non-homologous) and Initial (or Immortal). These processes are labeled by their initials, and each corresponds to a specific probability factor as follows:

$$\begin{aligned} B_\tau &= \lambda\beta(\tau) & E_\tau &= \mu\beta(\tau) & I_\tau &= 1-\lambda\beta(\tau) \\ H_\tau &= e^{-\mu\tau}(1-\lambda\beta(\tau)) & N_\tau &= (1-e^{-\mu\tau}-\mu\beta(\tau))(1-\lambda\beta(\tau)) \end{aligned} \quad (1)$$

In a tree, time flows forward from the root to the leaves, and to each node of the tree we associate a time parameter τ which is set equal to the length of the incoming branch. For the root, $\tau = \infty$ by assumption of stationarity at the root, and the resulting equilibrium length distribution of the immortal link sequence is geometric with parameter $B_\infty = \lambda/\mu$ (where length 0 is possible); other links will have left no descendants since $H_\infty = N_\infty = 0$.

In the original TKF91 model, a simple substitution process known as the "Felsenstein81 model" [Fel81] has been used. It is straightforward to replace this by more general models for substitutions of nucleotides or amino acids [HWK$^+$00]. In the present paper, when a new non-homologous character appears at a node (as the result of a B or N process), it is always drawn from the equilibrium distribution; if a character at a node is homologous to the character at its immediate ancestral node, then the probability of this event is given by the chosen substitution model.

Note that we do not sum over possible *alignments*, but over what we call evolutionary *histories* (see section 7.1), which is a discrete summary of the actual evolutionary events in the model, specifying the fate of all ancestral nucleotides. As was pointed out in [TKF91], for two sequences, there is almost a correspondence between alignment and evolutionary history (except that gaps in two sequences that immediately follow each other may be interchanged without changing the interpretation as alignments whereas the evolutionary

interpretation in the TKF model is different.) This almost-correspondence breaks down for trees with larger number of leaves, since information on what occurs at internal nodes is lost in an alignment, which only records the homology structure of observed nucleotides.

3 One-state recursion: A graphical proof for two sequences

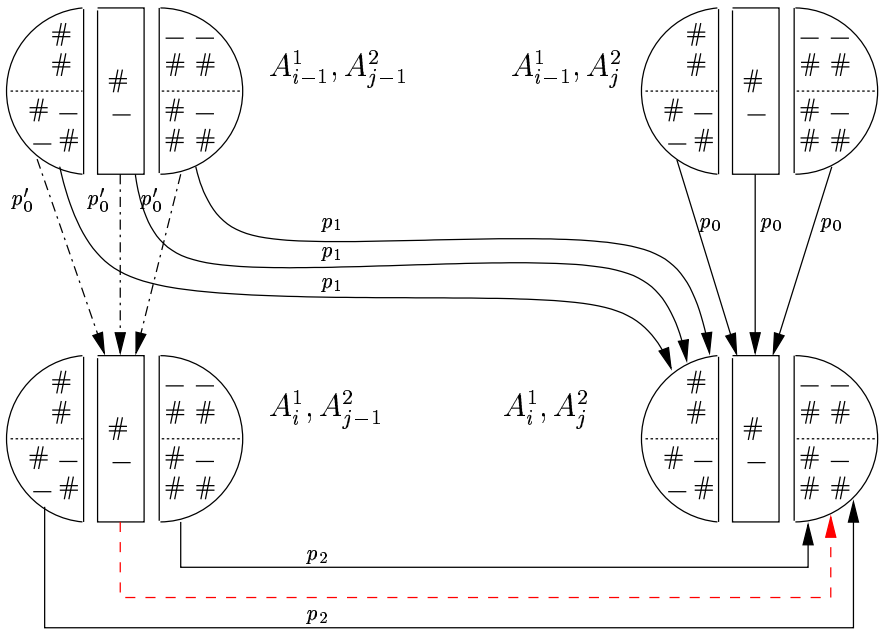


Figure 1: A graphical representation of the three-state recursion for two-sequence alignment. The p_k are transition probabilities associated to S^k (see text): $p_0 = B_\infty E_\tau$, $p_1 = B_\infty H_\tau + B_\infty N_\tau$ and $p_2 = B_\tau$. Circles represent positions in the dynamic programming table, circle segments correspond to the three states. The # and $-$ symbols represent residues and gaps in the last (two) column(s) of the partial alignment.

In [HWK+00] a proof has been given for the one-state recursion in the case of two-sequence alignment. Here we provide a different proof for this particular case. Our proof for the general case follows a similar line of ideas.

Suppose A^1 is the ancestral sequence observed at the root and A^2 is the descendant sequence which results from the TKF91 process after time τ. Let A^k_i (resp. a^k_i) denote the i-long prefix (resp. the ith character) of sequence A^k. Let $P(i,j)$ be the joint probability of observing A^1_i and A^2_j. Given that $P(i-1,j)$, $P(i,j-1)$ and $P(i-1,j-1)$ are known, we want to compute $P(i,j)$.

The possible evolutionary histories (or alignments) are customarily [TKF91] classified into three groups, S^0, S^1 and S^2, according to whether the rightmost root link has 0, 1 or at least 2 descendants at the leaf. Another way to define these sets is by the last columns in the alignment; symbolically, $S^0 = \{\binom{\#}{-}\}$, $S^1 = \{\binom{\#}{\#}, \binom{\#}{-}\binom{-}{\#}\}$ and $S^2 = \{\binom{\#}{\#}\binom{-}{\#}, \binom{-}{\#}\binom{-}{\#}\}$, where the upper symbols represent nucleotides at the root. These sets are associated to the evolutionary

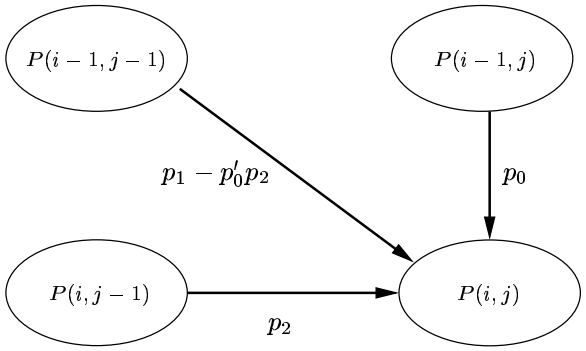


Figure 2: The one-state recursion for two sequences. The factors next to the arrows are "transition factors," not probabilities. Due to combining states, negative terms appear to compensate for illegal transitions implicit in other transition factors.

processes as follows. After the $B_\infty E_\tau$ process, the rightmost root link has 0 descendants, so the history is in $\{\begin{smallmatrix}\#\\ \_\end{smallmatrix}\}$. After $B_\infty H_\tau$ or $B_\infty N_\tau$ it is in $\{\begin{smallmatrix}\#\\ \#\end{smallmatrix}, \begin{smallmatrix}\#\\ \_\ \#\end{smallmatrix}\}$, while after a birth at the leaf node (B_τ), the alignment ends in either of $\{\begin{smallmatrix}\#\ \_\\ \#\ \#\end{smallmatrix}, \begin{smallmatrix}\_\ \_\\ \#\ \#\end{smallmatrix}\}$. These processes may happen in any order, except that a transition from $\{\begin{smallmatrix}\#\\ \_\end{smallmatrix}\}$ to $\{\begin{smallmatrix}\#\ \_\\ \#\ \#\end{smallmatrix}, \begin{smallmatrix}\_\ \_\\ \#\ \#\end{smallmatrix}\}$ is not allowed: new births at the leaf (B_τ processes) are allowed except after an extinction (E_τ) process, see table 1. A graphical depiction of the three-state recursion that implements this restriction is provided in Figure 1. The labeled circle segments represents probabilities of observing the associated sequence prefixes, provided the alignment ends according to the labeling.

If we combine the three states at each position, by summing their probabilities, we can still compute the contribution of the vertical and diagonal transitions to the probabilities at (i,j), since the transition probabilities (p_0 and p_1) do not depend on the state. But, for the horizontal transition we would include a contribution from the illegal transition $\{\begin{smallmatrix}\#\\ \_\end{smallmatrix}\} \to \{\begin{smallmatrix}\#\ \_\\ \#\ \#\end{smallmatrix}, \begin{smallmatrix}\_\ \_\\ \#\ \#\end{smallmatrix}\}$ (denoted by a dashed arrow), leading to an overestimation of the probability at (i,j). This overestimation can be exactly corrected for, however, since the segment labeled $\begin{smallmatrix}\#\\ \_\end{smallmatrix}$ at $(i,j-1)$ is the result of a $B_\infty E_\tau$ process from position $(i-1,j-1)$ (the dot-dashed arrows in Figure 1). In summary, if we let $P(i,j)$ be the sum of the three conditional probabilities at (i,j), we obtain the following one-state recursion:

$$\begin{aligned} P(i,j) \;=\; & P(i-1,j) \times B_\infty \pi(a_i^1) E_\tau + \\ & P(i-1,j-1) \times B_\infty \pi(a_i^1) \left[N_\tau \pi(a_j^2) + H_\tau p(a_i^1 \to a_j^2) \right] + \\ & \left[P(i,j-1) - P(i-1,j-1) B_\infty \pi(a_i^1) E_\tau \right] \times B_\tau \pi(a_j^2) \end{aligned}$$

Here $\pi(\cdot)$ denotes the equilibrium nucleotide distribution, and $p(\alpha \to \gamma)$ is the probability that nucleotide α evolves into γ in time τ. After rearranging terms, we arrive at the recursion depicted in Figure 2.

Although it is possible, in principle, to manually derive analogous recursions for arbitrary number of sequences following the same line of reasoning, such an approach rapidly gets very tedious, as the number of terms which one needs to consider grows exponentially

with the number of sequences. The goal of the present paper is to provide a more constructive method that allows us to prove that a one-state recursion indeed exists for arbitrary trees. Of more practical importance, the proof immediately suggests an efficient method for calculating the transition factors, through a post-order tree traversal algorithm, thus allowing computational derivation of the desired one-state recursion for arbitrary trees.

4 Algorithm

Using the TKF model, it is easy to compute the likelihood of a specific evolutionary history, given as a sequence of events. In effect, this integrates out all possible evolutionary histories on the edges of the phylogenetic tree, while fixing the nucleotide sequences and their mutual alignment at internal nodes. The likelihood of observing the given nucleotide sequences at the leaves is obtained by summing this likelihood over all possible sequences of events. Since we usually do not have information about nucleotide sequences at the internal nodes, such a computation should sum out the internal nucleotide sequences as well. This is what our algorithm computes.

In order to write down the main theorem, we need some notation. T is the set of nodes of the phylogenetic tree relating the sequences, and r is its root. If $n \in T$ is a node, then n_l and n_r denote the left and right immediate descendants of n. For a node n, $\tau(n)$ denotes the length of the incoming branch, and we abbreviate $B_n := B_{\tau(n)}$ etc.. For the root, $\tau(r) = \infty$ by assumption. Finally, $p_n(\alpha \to \gamma)$ is the probability that nucleotide α evolves into γ in time $\tau(n)$.

Theorem 1 *Suppose $A^1, \ldots, A^k$ are sequences related by a phylogenetic tree T with k leaves and root r. Let $\mathbf{K} = (\mathrm{K}_1, \ldots, \mathrm{K}_k)$ and let $P(\mathbf{K})$ be the probability of producing, under the TKF91 model, the prefixes of $A^1, \ldots, A^k$ up to position $\mathrm{K}_1, \ldots, \mathrm{K}_k$, respectively, where the numbering starts at 0 with the immortal link. The following algorithm computes $P(\mathbf{K})$:*

$$\begin{aligned} P(\mathbf{0}) &= \frac{\prod_{n \in T}(1 - B_n)}{G^{\mathbf{0}}_{\mathbf{0}}(r, -)}\,, \\ P(\mathbf{K}) &= \frac{1}{G^{\mathbf{0}}_{\mathbf{K}}(r, -)} \sum_{\mathbf{v} \in \{0,1\}^k \setminus \mathbf{0}} (-G^{\mathbf{v}}_{\mathbf{K}}(r, -)) P(\mathbf{K} - \mathbf{v})\,. \end{aligned}$$

If n is an internal node, then

$$\begin{aligned} G^{\mathbf{v}}_{\mathbf{K}}(n, \alpha) &= \left[E_{n_l} G^{\mathbf{v}}_{\mathbf{K}}(n_l, -) + \sum_{\gamma} \Big(H_{n_l} p_{n_l}(\alpha \to \gamma) + N_{n_l} \pi(\gamma) \Big) G^{\mathbf{v}}_{\mathbf{K}}(n_l, \gamma) \right] \\ &\quad \times \left[E_{n_r} G^{\mathbf{v}}_{\mathbf{K}}(n_r, -) + \sum_{\gamma} \Big(H_{n_r} p_{n_r}(\alpha \to \gamma) + N_{n_r} \pi(\gamma) \Big) G^{\mathbf{v}}_{\mathbf{K}}(n_r, \gamma) \right]\,, \\ G^{\mathbf{v}}_{\mathbf{K}}(n, -) &= G^{\mathbf{v}}_{\mathbf{K}}(n_l, -) G^{\mathbf{v}}_{\mathbf{K}}(n_r, -) - B_n \sum_{\gamma} \pi(\gamma) G^{\mathbf{v}}_{\mathbf{K}}(n, \gamma)\,. \end{aligned}$$

If n is a leaf node, then

$$G^{\mathbf{v}}_{\mathbf{K}}(n,\alpha) = \begin{cases} 1, & \text{if } \mathrm{v}_n = 1 \text{ and } a^n_{\mathrm{K}_n-1} = \alpha, \\ 0, & \text{otherwise.} \end{cases}$$

$$G^{\mathbf{v}}_{\mathbf{K}}(n,-) = \begin{cases} 1, & \text{if } \mathrm{v}_n = 0, \\ -B_n\pi(a^n_{\mathrm{K}_n-1}), & \text{if } \mathrm{v}_n = 1. \end{cases}$$

By K_n we mean the component of the vector $\mathbf{K}$ corresponding to leaf n*, and similarly for v_n. By $a^n_{\mathrm{K}_n-1}$ we denote the character at position $\mathrm{K}_n - 1$ in the sequence at leaf n.*

The numbers $G^{\mathbf{v}}_{\mathbf{K}}(r,-)$ are the *transition factors* and play the same role as the transition factors in Figure 2. They are sums and differences of probabilities, designed to exactly cancel all contributions of illegal transitions due to the merging of states into a single probability, analogous to the two-sequence case described in §3. Since we now have k leaves, the dynamic programming recursion for P refers back to $2^k - 1$ entries, instead of just 3.

Note that $G^{\mathbf{0}}_{\mathbf{K}}$ does not depend on $\mathbf{K}$, so a small acceleration can be achieved by computing the pre-factor $1/G^{\mathbf{0}}_{\mathbf{K}}$ only once. A more substantial acceleration is achieved by noting that $G^{\mathbf{v}}_{\mathbf{K}}(n,\alpha)$ and $G^{\mathbf{v}}_{\mathbf{K}}(n,-)$ depend only on the components of $\mathbf{v}$ which correspond to the leaves contained in the subtree of n. Therefore, if the tree with root n has s leaves, it is enough to compute $G^{\mathrm{v}}_{\mathbf{K}}(n,\alpha)$ for 2^s values of $\mathbf{v}$. Once these are computed, any entry for the ancestral node can be computed in constant time. Therefore the overall running time of the algorithm is $2^k L^k$, where k is the number of sequences and L is the geometric mean of the sequence lengths.

5 Example

As a concrete example, we have used our algorithm to find the most likely evolutionary tree which relates the following four globins: human $\alpha 1$ and β hemoglobins, human myoglobin, and leghemoglobin from the bean *Canavalia lineata*. There are three topologically different unrooted trees with four leaves, and we have estimated maximum likelihood parameters for each of them. The most likely tree is, as expected, the one that groups human $\alpha 1$ and β hemoglobin together; the likelihoods of the other two trees are about a factor 400 smaller (see Figure 3). After obtaining the maximum likelihood tree, the maximum likelihood alignment can be obtained using the Viterbi algorithm. This alignment is shown in Table 2. For a more detailed discussion on the Viterbi algorithm, see §C.

We have optimised the likelihood as a function of the five branch length parameters and λ; the deletion rate parameter μ has been adjusted to make the expected sequence length $\frac{\lambda}{\mu-\lambda}$ equal to the average sequence length. By using corner cutting methods we could confine the calculation to the parts of the dynamic programming table which contribute non-infinitesimally to the final probability, resulting in a speedup of about a factor of 500.

At the extremal point we estimated the second derivative to get a covariance matrix; see Figure 3 for the standard deviations in the directions of the eigenvectors of this matrix.

6 Idea of proof

Our aim is to give a "one-state" dynamic programming algorithm for calculating the joint likelihood of observing a set of sequences at the leaves of an evolutionary tree. This likelihood is the sum of the probabilities of all evolutionary histories which produce the observed sequences at the leaves.

We introduce the concept of an "event" which corresponds to a nucleotide birth at a node of the phylogenetic tree and its subsequent fate down its descendant subtree. More precisely, in the process of going from one node to a descendant node, one of the following three things may happen: The nucleotide may survive as a homologous nucleotide, it may die leaving at least one surviving new nucleotide, or it may go extinct altogether. These three possibilities are labeled H, N and E respectively, as discussed in §2. In the first two cases, other non-homologous descendants may have been born, but these possibilities are dealt with in subsequent events. Hence, in this model, evolutionary histories of nucleotide sequences correspond to sequences of events. However this correspondence is not one-to-one for two reasons. One reason is that we need to honour the TKF model, which does not allow extinct nucleotides to spawn new ones. (In Table 1, this is reflected by the third entry, $C \to -$, which is the only one that may not be followed by a string of #'s.) The other reason is that different sequences of events may represent the same evolutionary history, since the relative ordering of events on disjoint subtrees has no meaning. In a Markov chain approach, these two requirements are met by defining states and by ruling out certain events (*i.e.* transitions, in the Mealy machine view [DEKM98]) in certain states. Essentially that is how we also approach the problem. We define a "state" and use it to rule out events which violate the TKF model. Our choice of "state" (see Fig. 4) still allows over-counting of histories. To overcome this, we require the events to be *ordered* in a specific way depending on the state.

```
Hba1: MV--LSPADKTNVKAAWGKVGAHAGEYGAEALERMFLSFPTTKTYFPHF--DLS-H-----GSAQVKGHGKKVAD-AL-TNA-
Hbb:  MV-HLTPEEKSAVTALWGKV--NVDEVGGEALGRLLVVYPWTQRFFESF-GDLSTPDAVM-GNPKVKAHGKKVLG-AF-SDG-
Myo:  MG--LSDGEWQLVLNVWGKVEADIPGHGQEVLIRLFKGHPETLEKFDKFK-HLKSEDE-MKASEDLKKHGATVLT-AL-GGI-
Legh: MGA-FSEKQESLVKSSWEAFKQNVPHHSAVFYTLILEKAPAAQNMFS-F---LSNGVD-P-NNPKLKAHAEKVFKMTVDSAVQ

      VAHVDDMPNALSALSDLHAHKLRVDPVNFK-LLSHCLLVTLAAHLPAEFTPAVHASLDKFLASVSTVL-TS-K---YR-
      LAHLDNLKGTFATLSELHCDKLHVDPENFR-LLGNVLVCVLAHHFGKEFTPPVQAAYQKVVAGVANAL-AH-K---YH-
      LKKKGHHEAEIKPLAQSHATKHKI-PVKYLEFISECIIQVLQSKHPGDFGADAQGAMNKALELFRKDMASNYKELGFQG
      LRAKGEVVLADPTLGSVHVQKGVLDP-HFL-VVKEALLKTFKEAVGDKWNDELGNAWEVAYDELAAAI-KK-A-MGSA-
```

Table 2: The maximum likelihood alignment for the first pedigree in Figure 3. The log-likelihood of this alignment is -1593.223.

The ordering we use has the property that in properly ordered sequences of events, an event which violates the TKF model can be recognised by comparing it to its immediate predecessor. This means that we do not need the state to decide which events are allowed, but the state is still used to define the proper ordering. We can now use a recursion which, as a first approximation, allows all events. This includes all *legal* sequences of events, as well as some sequences of events which end in an illegal pair, either because it is not properly ordered, or because it violates the TKF model. As a second approximation, we subtract all sequences of events which end in particular illegal pairs. In the same way as before, this includes some illegal triplets which should not have been subtracted, and these are added in again, etc., in a procedure reminiscent of inclusion-exclusion. We have dubbed the pairs, triplets etc. involved in this procedure the *disallowed sequences*.

It turns out that there exist only finitely many disallowed sequences, so that the inclusion-exclusion procedure stops after finitely many steps (bounded by the number of nodes in the tree). Moreover, disallowed sequences emit at most one nucleotide per leaf, and that leads to a recursion which refers back only to the $2^k - 1$ nearest predecessors in the k-dimensional dynamic programming table, where k is the number of leaves.

The ordering, and hence the disallowed sequences we have to sum over, still depends on the state. However, the state turns out to influence only the *ordering* of the elements of these disallowed sequences, not the set of events themselves. More precisely, for each state, there is a one-to-one correspondence between such sets (that we call *illegal sets*) and disallowed sequences. The terms involved in the summation do not depend on the ordering, and this enables us to sum away the dependence on the state. A small technical problem is that some events do not emit any character (known in the HMM literature as "silent states"). These events lead to self-references in the recursion, turning it into a (trivial, one-dimensional) linear equation. We can sum over infinitely many possible silent events using the standard "wing folding" technique [Edd01], which amounts to solving the linear equation.

It remains to compute the transition factors, which involve sums over illegal sets of events that lead to specific emissions. The number of illegal sets is still exponential in the number of nodes in the phylogenetic tree. However, using a correspondence between illegal sets and labelings of the tree, it is possible to compute this sum in linear time using a reverse-traversal algorithm. This recursion also deals with the nucleotide assignments in the internal nodes of the tree, similar to Felsenstein's postorder tree traversal algorithm [Fel81].

7 The general one-state recursion

In this section we introduce the general one-state recursion. The main result is presented in §7.2; to maintain the flow of discussion, we defer the proof until Appendix A. In §8, to arrive at the algorithm discussed in §4, we combine the result of the present section with a

fast algorithm for computing the transition factors.

7.1 Definitions

Let T denote the set of nodes of a rooted phylogenetic binary tree. Note that we suppose the tree edges to be given and fixed, and that we only include the nodes in our representation T. Although the tree is rooted, the root position is immaterial because of reversibility of the model (Felsenstein's *Pulley principle*, [Fel81]). The root may coincide with an internal node, giving a zero-length edge and somewhat simplifying the recursion, but for clarity we do not consider that case here.

A *subtree* t of T is a set of nodes with the properties that (i) for any node $n \in t$, its descendants are in t, and (ii) there is a unique node, called the *root* of t and denoted by $r(t)$, without a parent. A *labeling* of a tree t is a function $l : t \to \{B^\alpha, -, H^\alpha, N^\alpha \mid \alpha \in \mathcal{A}\}$, where $\mathcal{A}$ is the set alphabet, which is $\{A, C, G, T\}$ in the case of DNA sequences. B, H and N stand for birth, homologous and new, respectively, as discussed in §2, and α is the character associated to the process. The symbol "$-$" stands for "no character" and means that an E_τ (extinction) process has occurred somewhere higher up in the tree.

An *event* e is a pair (e^t, e^l), where e^t is a subtree of T and e^l a labeling of e^t, with the additional properties that (i) $e^l(n) = B^\alpha$ (any α) if and only if $n = r(e^t)$, and (ii) if $e^l(n) = -$ for any $n \in e^t$, then $e^l(n') = -$ for all descendants n' of n. In this way, an event represents the birth of a nucleotide at $r(e^t)$ and its subsequent fate down the phylogenetic tree. An event carries a definite probability $p(e)$ which can be calculated from the TKF91 model, and we will concern ourselves with that in §8. Note that an event gives rise to at most one new nucleotide at each leaf sequence, and that such nucleotides may or may not be homologous to each other.

We consider *sequences of events*, denoted by $E = (e_1, e_2, \ldots, e_N)$. These give rise to sequences of *states* $(S_1, S_2, \ldots, S_N)$, which are defined recursively as follows: $S_1 := T$ and $S_{i+1} := (S_i \setminus e_i^t) \cup \{n | n \in e_i^t \text{ and } e_i^l(n) \neq -\}$. The state S_i determines at which nodes new nucleotides may be born at event i. Figure 4 illustrates these definitions.

We introduce some notation for convenience: $t_i := e_i^t$, $r_i := r(e_i^t)$, $l_i := e_i^l$. These stand for the *tree*, the *root*, and the *labeling* of event e_i. We define the *context* c_i by $c_i := F$ if $r_i \in S_i$, and $c_i := I$ if $r_i \notin S_i$, meaning Fertile if the birth of event e_i occurred at a live node, Infertile if not. An event e_i is called *legal* if $c_i = F$. A sequence of events is called *legal* if all events are legal. We define those (and only those) states that may result from a sequence of legal events to be *legal states*. Equilvalently, we may define legal states as those which are the result of a single event at the root, or yet differently as T with a number of proper subtrees removed.

Examples of legal and illegal states are provided in Figure 6.

In a sequence E of events, the ordering of the events is important insofar as it determines the ordering of events pertaining to each individual leaf sequence. Permutations do not change the evolutionary history represented by a sequence as long as, for all leaves n, the

ordering in the subsequence of those events which intersect the path from root to n is undisturbed. More precisely, two sequences $E = (e_1, \ldots, e_N), E' = (e'_1, \ldots, e'_N)$ represent the same history iff there exists a permutation σ of $\{1, \ldots, N\}$ such that

$$e_{\sigma(i)} = e'_i \quad \text{and} \quad i < j, \;\; t_i \cap t_j \neq \varnothing \;\Rightarrow\; \sigma(i) < \sigma(j).$$

We call these σ *allowed permutations* (for E). This defines an equivalence relation on the set of sequences of events. We call the equivalence class of any sequence of events the *(evolutionary) history* defined by that sequence.

Although in general the state depends on the ordering of events, the context c_j depends only on the history, since it depends on whether $r_j \in S_j$, which in turn depends on $l_i(r_j)$, for the unique $i < j$ with $r_j \in t_i$ and $r_j \notin t_k$ for all $i < k < j$. The pair (j, i) so defined is equivariant under allowed permutations, *i.e.* the indices of the pair transform according to the permutation. This allows us to define a *legal history* to be a history represented by a legal sequence of events.

7.2 Summing away the states

We denote by $E(\mathbf{K})$ a set of legal sequences of events which emit the prefixes of the given sequences $A^1, \ldots, A^k$ up to position $\mathbf{K} = (\mathrm{K}_1, \ldots, \mathrm{K}_k)$, and which contains precisely one representative of each history compatible with the emissions. The quantity of interest is the likelihood

$$P(\mathbf{K}) := \prod_{n \in T} (1 - B_n) \sum_{(e_1, \ldots, e_N) \in E(\mathbf{K})} p(e_1) \cdots p(e_N)\,.$$

To state the main result of this section, we need to introduce one more definition, which admittedly comes out of the blue at this point. The connection is made at the end of the proof in Appendix A.

Definition 1 (Set of Nested Events) *A set of events $\{e_1, \ldots, e_N\}$ is called a* set of nested events *if $t_i \supseteq t_j \Rightarrow l_i(r_j) = -$.*

Lemma 1 *The likelihood $P(\mathbf{K})$ satisfies the following equation:*

$$\begin{aligned} P(\mathbf{K}) \;=\; & \sum_e P(\mathbf{K} - \mathbf{v}_e) p(e) \\ & - \sum_{\{e_1, e_2\} \text{ nested events}} P(\mathbf{K} - \mathbf{v}_{e_1} - \mathbf{v}_{e_2}) p(e_1) p(e_2) \\ & + \sum_{\{e_1, e_2, e_3\} \text{ nested events}} P(\mathbf{K} - \mathbf{v}_{e_1} - \mathbf{v}_{e_2} - \mathbf{v}_{e_3}) p(e_1) p(e_2) p(e_3) \\ & - \cdots \end{aligned} \tag{2}$$

Here $\mathbf{v}_e$ is the emission vector corresponding to the event e, that is, a k-dimensional vector whose component corresponding to each leaf is 0 if the leaf is labeled $-$ (or if it is outside

$t(e)$ and hence unlabeled), and 1 otherwise.

Proof: See Appendix A.

7.3 Eliminating the silent events

Note that, for a nontrivial tree T, the set $E(\mathbf{K})$ of legal sequences of events emitting a certain prefix is infinite, even for $\mathbf{K} = \mathbf{0}$. This is due to the so-called silent events, *i.e.* events e without emissions at the leaves ($\mathbf{v}_e = \mathbf{0}$). This results in the appearance of the term $P(\mathbf{K})$ on the right-hand side of (2). However, the contribution of these silent events can easily be calculated using a trick analogous to the "wing folding" technique in the HMM literature [Edd01]. A similar trick can be found in [SH01].

Note that, by definition, the set $\{e\}$ is a set of nested events all events e, and therefore (2) can be rewritten as follows:

$$P(\mathbf{K}) = \sum_{M \neq \varnothing \text{ nested events,}} P\left(\mathbf{K} - \textstyle\sum_{e_i \in M} \mathbf{v}_{e_i}\right) (-1)^{|M|+1} \prod_{e_i \in M} p(e_i). \tag{3}$$

We can decompose the sum as

$$\begin{aligned} P(\mathbf{K}) \quad = \quad & \sum_{\substack{M \neq \varnothing \text{ nested events,} \\ \text{not all } \mathbf{v}_{e_i} = \mathbf{0}}} P\left(\mathbf{K} - \textstyle\sum_{e_i \in M} \mathbf{v}_{e_i}\right) (-1)^{|M|+1} \prod_{e_i \in M} p(e_i) \\ & + P(\mathbf{K}) \sum_{\substack{M \neq \varnothing \text{ nested events,} \\ \text{all } \mathbf{v}_{e_i} = \mathbf{0}}} (-1)^{|M|+1} \prod_{e_i \in M} p(e_i). \end{aligned}$$

Solving this equation for $P(\mathbf{K})$ gives the correct answer and includes histories with $0, 1, 2, \ldots$ silent events. A heuristic way to see this is to observe that the equation $X = a + sX$, where s and a are the silent and non-silent contributions, respectively, is solved by $X = a/(1-s) = a(1 + s + s^2 + \ldots)$. This expansion clearly shows the separate contributions of chains of silent events of length $0, 1, 2, \ldots$. The reasoning in the proof of Lemma 1 makes sure that only sequences of legal events are included, and that applies to the silent states as well.

8 Pruning the tree

The recursion (2) enables us to compute $P(\mathbf{K})$, but it requires summing over a number of terms exponential in the number of sequences. A similar problem occurs when calculating the likelihood of nucleotide emissions on a tree under a simple substitution model, in which case one needs to sum over an exponential number of nucleotide assignments to internal nodes. This can be computed in linear time by Felsenstein's linear time post-order tree traversal algorithm, which is again a dynamic programming algorithm, now on a tree instead of the more familiar square lattice. We proceed in a similar way, by finding a correspondence between certain labelings of T and sets of nested events. The summation over all such

labelings, including the $(-1)^{k+1}$ sign arising from the inclusion-exclusion argument, can be performed in linear time by a dynamic programming algorithm similar to Felsenstein's.

The probability factor associated to an event is the product of conditional factors at the nodes of the phylogenetic tree, and the conditional factor at a node only depends on the labeling of the node and its ancestral node. The label determines both the process and the associated nucleotide, except for "$-$", which implies an Extinction event only if its ancestral node is not labeled "$-$". Symbolically,

$$\begin{aligned} & & p(B^\alpha|-) &= B_n\pi(\alpha), \\ p(N^\alpha|X^\gamma) &= N_n\pi(\alpha), & p(N^\alpha|-) &= 0, \\ p(H^\alpha|X^\gamma) &= H_n p_n(\gamma\to\alpha), & p(H^\alpha|-) &= 0, \\ p(-|X^\gamma) &= E_n, & p(-|-) &= 1, \end{aligned}$$

where $\alpha,\gamma\in\mathcal{A}$ (c.f. §7.1). The symbol X, denoting the process in the ancestral node, may be anything except "$-$". The probability factors B_n, N_n, H_n and E_n are subscripted with a node, and are defined (c.f. (1)) in terms of the length of the incoming branch (∞ if $n=r(T)$). Note that $p(B^\alpha|X^\gamma)$ will never occur.

This allows us to calculate probability factors of events, but we want to sum over nested events directly. It follows from Definition 1 that for each node $n\in T$, at most one event in the nested set has n labeled with a symbol other than "$-$". Furthermore, since the roots of events are uniquely identified as the only nodes labeled B^α, there is a one-to-one correspondence between sets of nested events and labelings of T with $B^\alpha, H^\alpha, N^\alpha, -$ which obey the following rule: the root and the immediate descendants of "$-$" are labeled with B^α or "$-$". See Figure 7 for an example. Indeed, such labelings can be decomposed into sets of nested events by recursively removing subtrees containing exactly one B^α label at their roots, and resetting the labelings to "$-$". Note that the all-"$-$" tree has a proper labeling according to these rules, and corresponds to the empty set of nested events.

The factor associated to a nested set S is the product of the separate probability factors of the event in S, with a sign $(-1)^{|S|+1}$. The exponent $|S|+1$ is just one more than the number of B^α's occurring in the associated tree labeling. Hence, analogous to computing the probability factor of an event, the factor associated to a tree labeling may be calculated (except for a single overall minus sign) as the product of several "conditional factors" as follows:

$$\begin{aligned} f(B^\alpha|X^\gamma) &= -E_n B_n\pi(\alpha), & f(B^\alpha|-) &= -B_n\pi(\alpha), \\ f(N^\alpha|X^\gamma) &= N_n\pi(\alpha), & f(N^\alpha|-) &= 0, \\ f(H^\alpha|X^\gamma) &= H_n p_n(\gamma\to\alpha), & f(H^\alpha|-) &= 0, \\ f(-|X^\gamma) &= E_n, & f(-|-) &= 1. \end{aligned}$$

If we group terms in (3) according to their total emission vector $\mathbf{v}=\mathbf{v}_{e_1}+\cdots+\mathbf{v}_{e_m}$, the

sum of contributing factors may be obtained by a Felsenstein-like recursion:

$$\begin{aligned} F(n,\alpha) &= \left[E_{n_l} F^{\mathbf{v}}_{\mathbf{K}}(n_l,-) + \sum_\gamma F(n_l,\gamma)\left(H_{n_l} p_{n_l}(\alpha\to\gamma) + N_{n_l}\pi(\gamma) - E_{n_l} B_{n_l}\pi(\gamma)\right)\right] \\ &\quad \times \left[E_{n_r} F(n_r,-) + \sum_\gamma F(n_r,\gamma)\left(H_{n_r} p_{n_r}(\alpha\to\gamma) + N_{n_r}\pi(\gamma) - E_{n_r} B_{n_r}\pi(\gamma)\right)\right], \\ F(n,-) &= \left[F^{\mathbf{v}}(n_l,-) - \sum_\gamma B_{n_l} F^{\mathbf{v}}(n_l,\gamma)\pi(\gamma)\right] \times \left[F^{\mathbf{v}}(n_r,-) - \sum_\gamma B_{n_l} F^{\mathbf{v}}(n_r,\gamma)\pi(\gamma)\right], \end{aligned}$$

where n_l and n_r denotes the left and right child node of n. Here, $F(n,\alpha)$ (resp. $F(n,-)$) is the sum of all products of conditional factors on the subtree with root n, given that n is labeled X^α (resp. "$-$").

Note that we suppressed the dependence on $\mathbf{v}$. This dependence only surfaces when n is a leaf, in which case the recursion terminates with $F(n,\alpha) = 1$ if $\mathrm{v}_n = 1$ and α is the character in sequence A^n at the current position, and $F(n,-) = 1$ if $\mathrm{v}_n = 0$; $F = 0$ in all other cases. The transition factor associated to $\mathbf{v}$ is finally

$$\sum_{\substack{\{e_1,\ldots,e_k\} \text{ nested set} \\ \mathbf{v}_{e_1}+\cdots+\mathbf{v}_{e_k}=\mathbf{v}}} (-1)^{k+1} \prod_i p(e_i) = -\left(F(r,-) - \sum_\alpha F(r,\alpha) B_r \pi(\alpha)\right),$$

where r is the root of T. A final useful simplification occurs if we set $G(n,\alpha) = F(n,\alpha)$ and $G(n,-) = F(n,-) - \sum_\gamma B_n\pi(\gamma)F(n,\gamma)$, in which case the recursion for G takes the form given in §4 and the transition factor is simply $-G(r,-)$.

To compute $P(\mathbf{K})$ we have to eliminate the silent events by solving a linear equation, as described in §7.3. This amounts to dividing each transition factor by

$$1 - \sum_{\substack{\varnothing\neq\{e_1,\ldots,e_k\} \text{ nested set} \\ \mathbf{v}_{e_1}+\cdots+\mathbf{v}_{e_k}=\mathbf{0}}} (-1)^{k+1} \prod_i p(e_i), \tag{4}$$

one minus the transition factor associated to the null emission. Since the recursion for F also includes the "$-$"-labeled tree which gets assigned a term $+1$ (corresponding to also including the empty set in (4)), in fact $G(r,-)$ for $\mathbf{v} = \mathbf{0}$ is precisely equal to (4).

It remains to compute the initial value $P(\mathbf{0})$, the probability of emitting no nucleotides to any sequence and being left with only the immortal link. According to the TKF91 model, this is the product of the immortal link pre-factors $I_n = 1 - B_n$, where the product extends over all nodes in T, and silent events are included by dividing this by $G(r,-)$. This completes the proof of Theorem 1. ■

9 Discussion

The importance of incorporating statistical analysis into biological studies has become abundantly clear over the years. In particular, many interesting problems which arise in the field

of bioinformatics have been successfully addressed using statistical models. For instance, the evolution of biological sequences, which is the focus of the present paper, has been formulated in a statistical framework in which mutation events are seen as stochastic processes. A closer examination of the past progress reveals, however, that although substitution processes have been modelled as continuous-time evolutionary processes for more than three decades [JC69] and have been widely used in phylogenetics and geneology [Fel01], modelling insertion and deletion processes based on evolution has only recently been generalised to arbitrary number of sequences [SH01, Hei01, Mik02, HJP02]. Furthermore, implementations of such generalisations have hitherto required quite a large running time. Alternative probabilistic approaches to sequence alignment exist (for example, see [Mit99]), but insertion and deletion processes in such models are not explicitly based on evolution.

In the present paper, we have constructed several algorithmic improvements which make multiplthee statistical alignment computationally tractable. More precisely, we have proved that the *one-state recursion* exists for arbitrary number of sequences, thus reducing the space complexity by a factor of $O(\sqrt{5}^k)$, where k is the number of sequences. Furthermore, we have developed a reverse traversal algorithm which calculates, with time complexity linear in the number of sequences, each transition factor appearing in the one-state dynamic programming algorithm. This reduces the time complexity of the entire algorithm to $O(k\,2^k L^k)$, where L is the geometric mean sequence length. In fact, most of the transition factors appearing in the recursion share a common part, which hence needs to be calculated only once. Therefore, with a more careful implementation, the running time of the final algorithm can be reduced to $O(2^k L^k)$.

The achieved improvements are not only theoretical, but make multiple statistical alignment on trees possible in practice, as we have shown in the globin example. Compared to the ordinary hidden Markov model implementation on a four-sequence tree, our method gives rise to a speedup of about a factor 50, as well as reducing the number of states from 45 to 1, with an associated reduction in memory usage.

Our algorithm is useful mainly for trees with small number of leaves, and may be used for hypothesis testing or as a basis for the quartet method. Even for small trees, however, corner cutting techniques are necessary in practical applications. The example of §5 requires computing a table of size 4.8×10^8 in a straightforward implementation, while corner cutting can reduce this by a factor of about 500. Because the probability mass is highly concentrated around the alignment region, we lose almost no contribution to the total probability, even though only 0.2% of the entire table is actually visited. For more than, say, 6 sequences, our method will cease to be practical, and MCMC methods will be more useful. Some promising attempts have already been made in this direction [HB01, JH02]. Our algorithm can be used to compare exact likelihood calculations and approximations, and hence it might still be useful for developing new sampling techniques.

The underlying sequence evolution model has the drawback of treating only single nucleotide insertions and deletions, as was already mentioned in the original paper [TKF91].

A model that can handle insertion and deletion events of whole subsequences is expected to perform better [TKF92], just as affine gaps improve alignments in score-based methods. Similarly, models that incorporate some structure information will probably result in better multiple alignments as well. We hope to incorporate these ideas in the future.

Acknowledgments

The authors wish to thank Rune Lyngsø and Alexei Drummond for helpful discussions, and the referees for their thorough reading and detailed comments. Just before publication, Jens Ledet Jensen sent us an alternative proof of the results presented here. This research is supported by EPSRC under grant HAMJW and by MRC under grant HAMKA. Y.S. Song is supported in part by a grant from the Danish Natural Science Foundation (SNF-5503-13370).

A Proofs

The main goal of this section is to prove Lemma 1 from §7.2. In the first two subsections, we first establish some results which are used in the proof of Lemma 1.

A.1 Unique legal sequences

To define a unique ordering of events for legal histories, we introduce a condition on sequences of events. We suppose the nodes of T to be ordered. For purposes later on, we choose the specific ordering which satisfies, for every subtree t of T,

$$\{\text{left subtree of } t\} < \{\text{right subtree of } t\} < \text{root of } t,$$

where left and right are arbitrary but fixed. An example of our choice of ordering is shown in Figure 8. We also order (context,node) pairs as follows:

$$(c, n) < (c', n') \text{ if } c = I \text{ and } c' = F, \text{ or } c = c' \text{ and } n < n'.$$

To specify a unique ordering of events, we introduce condition Δ for a sequence of events $E = (e_1, \ldots, e_N)$ as follows:

$$\begin{aligned} \Delta(i): &\quad t_i \cap t_{i+1} \neq \varnothing \text{ or } (c_i, r_i) < (c_{i+1}, r_{i+1}), \\ \Delta: &\quad \Delta(i) \text{ holds}, \forall\, 1 \leq i < N. \end{aligned}$$

Lemma 2 *Condition Δ holds for precisely one representative of each legal history.*

Proof: First we prove existence. Let k be the first index for which $\Delta(k)$ is false. Interchanging e_k and e_{k+1} is an allowed permutation, and makes $\Delta(k)$ true, but might render $\Delta(k-1)$ false. In this case, interchanging e_{k-1} and e_k is allowed, and renders $\Delta(k-1)$ and $\Delta(k)$ true, the latter because $\Delta(k-1)$ was true originally. Continuing backwards,

eventually all $\Delta(i)$ for $i \leq k$ are true. By induction on k we find an allowed permutation π such that $\pi(E)$ satisfies Δ.

We now prove uniqueness. Assume there are two representatives for which condition Δ holds, E and E'. Let π be the allowed permutation such that $\pi(E) = E'$. We can find an $i < j$ such that $\pi(j) + 1 = \pi(i)$ [Chr96]. Since π is an allowed permutation, $t_i \cap t_j = \varnothing$. E' satisfies Δ, therefore t_j and t_i are part of the left and right subtree of a node C_0, respectively. Moreover, $i + 1 \neq j$, because this would contradict $\Delta(i)$ for E.

We prove by induction that for each $k \geq 0$, a node C_k exists such that t_{j-k} is in the left subtree of C_k, t_i is in the right subtree of C_k, and t_j is contained in either. This means that there is no k such that $i + 1 = j - k$, therefore there is only one representative that satisfies the Δ condition.

We already proved that the induction hypothesis holds for $k = 0$. Assume it holds for certain k. Four possible cases are to be considered.

1. $t_{j-k-1} \cap t_i \neq \varnothing$ and $t_{j-k-1} \cap t_{j-k} \neq \varnothing$. This implies that $C_k \in t_{j-k-1}$ and hence $t_{j-k-1} \cap t_j \neq \varnothing$. Since π is an allowed permutation, $\pi(i) < \pi(j-k-1)$ and $\pi(j-k-1) < \pi(j)$ but this is impossible.

2. $t_{j-k-1} \cap t_i \neq \varnothing$ and $t_{j-k-1} \cap t_{j-k} = \varnothing$. This means that t_{j-k-1} is on the right subtree of C_k, and therefore $r_{j-k-1} > rj - k$, but this contradicts the Δ condition, since all $c_j = F$ because the history is legal.

3. $t_{j-k-1} \cap t_i = \varnothing$ and $t_{j-k-1} \cap t_{j-k} \neq \varnothing$. This means that t_{j-k-1} is on the left part of C_k. For $C_{k+1} = C_k$ the condition of the induction holds.

4. $t_{j-k-1} \cap t_i = \varnothing$ and $t_{j-k-1} \cap t_{j-k} = \varnothing$. In this case $r_{j-k-1} < r_{j-k} < r_i$. The first inequality comes from the Δ condition, while the second one is from the induction hypothesis. However, these inequalities guarantee the existence of a proper C_{k+1}. ■

In properly ordered sequences, legal sequences may be recognised by looking at pairs of events.

Definition 2 *A pair of events* (e_1, e_2) *is called* pairwise legal *if* $t_1 \supseteq t_2 \Rightarrow l_1(r_2) \neq -$.

Lemma 3 *The set*

$$\{(e_1, \ldots, e_N) | \Delta \textit{ holds}, (e_i, e_{i+1}) \textit{ pairwise legal for } i = 1, \ldots, N-1\}$$

consists of precisely one representative of each legal history.

Proof: $\subseteq$: Since Δ holds, at most one of each legal history is included. We prove that illegal histories are not included. Let $e_1, \ldots, e_N$ be an illegal sequence in the set. Let e_i be the first event with $c_i = I$, that is, $r_i \notin S_i$. Since $c_j = F$ for $j < i$, in particular S_{i-1} is legal. Suppose $t_{i-1} \cap t_i \neq \varnothing$. If $t_{i-1} \subset t_i$, then $r_i \notin S_i$ implies $r_i \notin S_{i-1}$ but $r_{i-1} \in S_{i-1}$ and r_{i-1} is a descendant of r_i, in contradiction with S_{i-1} legal. So $t_{i-1} \supseteq t_i$, but then $r_i \notin S_i$ implies (e_{i-1}, e_i) is not pairwise legal. So $t_{i-1} \cap t_i = \varnothing$. Note that $c_{i-1} = F$ and $c_i = I$, so Δ is false, which is the required contradiction.

$\supseteq$: For all legal histories there exists a sequence obeying Δ by Lemma 2, and all neighbouring pairs in legal sequences are pairwise legal. ■

This result enables us to determine the sequences of events we want to sum over, by looking only at neighbouring pairs. To use the inclusion/exclusion trick we next have to characterise sequences of event pairs that do *not* obey our condition.

A.2 Disallowed sequences and illegal sets

Definition 3 *Suppose S_i is a legal state, and suppose that for all $j = i, \ldots, k-1$ we have (e_j, e_{j+1}) not pairwise legal or $\Delta(j)$ false. We call such (sub)sequences* disallowed sequences.

Lemma 4 *Let $(e_i, e_{i+1}, \ldots, e_k)$ be a disallowed sequence starting from state S_i. Then:*

a) The sequence (c_j, r_j), $i \leq j \leq k$, is strictly decreasing.

b) All events e_j with $c_j = F$ are disjoint.

c) For each node n, there is at most one event e_j with $n \in t_j$ and $l_j(n) \neq -$.

Note that statement (c) implies that there is at most one emission per leaf.

Proof: If (e_j, e_{j+1}) is not pairwise legal, then r_{j+1} is a descendant of r_j and $l_j(r_{j+1}) = -$. This means that $r_j > r_{j+1}$ and $c_{j+1} = I$, so $(c_j, r_j) > (c_{j+1}, r_{j+1})$. If $\Delta(j)$ is false, then it directly follows from the definition of condition Δ that $(c_j, r_j) > (c_{j+1}, r_{j+1})$.

Now, that the sequence (c_j, r_j) is strictly decreasing implies that, for some m, we have $c_i = c_{i+1} = \cdots = c_m = F$ and $c_{m+1} = c_{m+2} = \cdots = c_k = I$. For $i \leq j < m$ the pair (e_j, e_{j+1}) is pairwise legal since $c_{j+1} = F$, therefore we must have $\Delta(j)$ false, which implies that $t_j \cap t_{j+1} = \varnothing$. Furthermore, $t_j \cap t_{j+1} = \varnothing$ and $r_j > r_{j+1}$ together imply that there exists a subtree t of T such that t_j is in a right subtree of t, whereas t_{j+1} is in a left subtree of t. Applying this reasoning sequentially, we conclude that for all $i \leq a < b \leq m$ there exists a subtree t such that t_a (t_b) is in a right (left) subtree of t. Hence, $t_a \cap t_b = \varnothing$ and it follows that all of $t_i, t_{i+1}, \ldots, t_m$ are pairwise disjoint.

Note that S_{m+1} is a legal state since S_i is a legal state and all events $e_i, e_{i+1}, \ldots, e_m$ are legal events. For every $m < j \leq k$, that the event e_j is illegal implies that $r_j \notin S_j$. Now, suppose $S_j \cap t_j \neq \varnothing$. Then, S_j must contain at least one descendant of r_j, which is possible only if, since S_{m+1} is a legal state and $r_j \notin S_j$, at least one of $e_{m+1}, e_{m+2}, \ldots, e_{j-1}$ has a birth at a descendant node of r_j. But, this contradicts the fact that $r_{m+1} > r_{m+2} > \cdots > r_k$, and therefore we must have $S_j \cap t_j = \varnothing$ for every $m < j \leq k$.

Finally, the results from the previous two paragraphs together imply (c). ■

Lemma 5 *For a given legal initial state S, there is a one-to-one correspondence between illegal sets and disallowed sequences starting from S.*

Proof: We prove the correspondence by establishing two injections.

From disallowed sequence to illegal set: The events with $c_j = F$ are pairwise disjoint, and their trees not the subtree of any other because the initial state is legal. The trees with $c_j = I$ may be subtree of another t_k, in which case $l_k(r_j) = -$, so the events form an illegal set. There is only one disallowed sequence giving rise to this set, since the sequence (c_j, r_j) corresponding to the disallowed sequence is ordered.

From illegal set to disallowed sequence: Take the events whose trees are not subset of another, and whose root is in S, sorted decreasing by root. Follow them by the other events, sorted decreasing by root. A disallowed sequence results. ■

A.3 Proof of Lemma 1

For the ease of reference, we here restate the lemma. We denote by $E(\mathbf{K})$ a set of legal sequences of events which emit the prefixes of the given sequences $A^1, \ldots, A^k$ up to position $\mathbf{K} = (\mathrm{K}_1, \ldots, \mathrm{K}_k)$, and which contains precisely one representative of each history. The quantity of interest is the likelihood

$$P(\mathbf{K}) := \sum_{(e_1,\ldots,e_N)\in E(\mathbf{K})} p(e_1)\cdots p(e_N)\,.$$

Lemma 1 *The likelihood $P(\mathbf{K})$ satisfies the following equation:*

$$\begin{aligned} P(\mathbf{K}) \;=\; & \sum_e P(\mathbf{K} - \mathbf{v}_e)p(e) \\ & - \sum_{\{e_1,e_2\}\ \mathit{illegal}} P(\mathbf{K} - \mathbf{v}_{e_1} - \mathbf{v}_{e_2})p(e_1)p(e_2) \\ & + \sum_{\{e_1,e_2,e_3\}\ \mathit{illegal}} P(\mathbf{K} - \mathbf{v}_{e_1} - \mathbf{v}_{e_2} - \mathbf{v}_{e_3})p(e_1)p(e_2)p(e_3) \\ & - \cdots \end{aligned} \tag{5}$$

Here $\mathbf{v}_e$ is the emission vector corresponding to the event e, that is, a k-dimensional vector whose component corresponding to each leaf is 0 if the leaf is labeled $-$, or 1 otherwise.

Proof: Let $S(e)$ denote the state after event e in state S. Define $P_S(\mathbf{K})$ to be the likelihood of emitting legal sequences up to position $\mathbf{K}$ and ending up in state S. We allow illegal states S, in which case $P_S(K) = 0$ always. For brevity, we denote by $C(S_0, e_0, S_1, e_1, \ldots, e_{n-1}, S_n)$ the condition $S_0(e_0) = S_1 \wedge \ldots \wedge S_{n-1}(e_{n-1}) = S_n$, and by $\mathrm{Dis}(S, e_0, \ldots, e_n)$ the condition

that $(e_0, \ldots, e_n)$ is a disallowed sequence when starting from state S. Then

$$\begin{aligned} P_S(\mathbf{K}) \;=\; & \sum_{(S',e):C(S',e,S)} P_{S'}(\mathbf{K} - \mathbf{v}_e)p(e) \\ & - \sum_{\substack{(S',e,S'',e'):\\ C(S'',e',S',e,S)\wedge \mathrm{Dis}(S'',e',e)}} P_{S''}(\mathbf{K} - \mathbf{v}_e - \mathbf{v}_{e'})p(e')p(e) \\ & + \sum_{\substack{(S',e,S'',e',S''',e''):\\ C(S''',e'',S'',e',S',e,S)\wedge \\ \wedge \mathrm{Dis}(S''',e'',e',e)}} P_{S'''}(\mathbf{K} - \mathbf{v}_e - \mathbf{v}_{e'} - \mathbf{v}_{e''})p(e'')p(e')p(e) \\ & - \ldots \end{aligned}$$

This recursion sums over all sequences of events which do not have disallowed subsequences, since all terms in the first line that include a disallowed subsequence (of length 2) are subtracted in the second line; this also subtracts terms involving disallowed subsequences of length 3 that were not included in the first place, which are added in again in the third line, and so on. By Lemma 3, this implies that we indeed sum over all sequences of legal events (which end up in state S). Note that for illegal states S, $P_S(\mathbf{K}) = 0$ always, so we may define $\mathrm{Dis}(S, e_0, \ldots)$ arbitrarily for illegal states S.

Now we sum the recursion equation over all states S. The left-hand side becomes $P(\mathbf{K})$. The first summation on the right-hand side turns into

$$\sum_S \sum_{(S',e):S'(e)=S} P_{S'}(\mathbf{K} - \mathbf{v}_e)p(e) = \sum_{(S',e)} P_{S'}(\mathbf{K} - \mathbf{v}_e)p(e) = \sum_e P(\mathbf{K} - \mathbf{v}_e)p(e)\,,$$

which is the desired term. Similarly, the second summation becomes

$$\sum_{(S'',e',e):\mathrm{Dis}(S'',e',e)} P_{S''}(\mathbf{K} - \mathbf{v}_e - \mathbf{v}_{e'})p(e')p(e)\,.$$

Observe that the summand is independent of the ordering of events e', e. Hence, we can also sum over illegal *sets* of events and over S'', using Lemma 5. This turns the summation into

$$\sum_{S''} \sum_{\{e',e\}\text{ illegal}} P_{S''}(\mathbf{K} - \mathbf{v}_e - \mathbf{v}_{e'})p(e')p(e) = \sum_{\{e',e\}\text{ illegal}} P(\mathbf{K} - \mathbf{v}_e - \mathbf{v}_{e'})p(e')p(e),$$

which is the desired term. All other summations are dealt with analogously. Note that only finitely many summands contribute to the recursion, since illegal sets have a bounded number of elements. ■

B States for the HJP recursion

For a fixed number k of sequences, the total number of Markov states[1] in the HJP recursion depends on the tree topology. Let T be a k-leaved rooted binary tree. Let $w(n)$ denote the

[1] We refer the reader to [HJP02] for their definition of "state."

number of leaves joined to the internal node $n \in T$ and let $q(T)$ be the number of internal nodes $n \in T$ with $w(n) = 2$.

If $q(T) = 2$ and the root[2] r of the tree has $w(r) = 2$, then the number of Markov states in the HJP recursion for such a tree is equal to

$$\left(2^k + \sum_{j=0}^{k-1} 2^j\right) + 2\left[\sum_{n=0}^{k-4}\left(2^{k-2-n} + \sum_{j=0}^{k-2-n-1} 2^j\right)\right] + \sum_{n=1}^{k-4}\sum_{m=0}^{n-1}\left(2^{n-m} + \sum_{j=0}^{n-m-1} 2^j\right).$$

As the tree topology we just considered has the least number of states, the above formula is a lower bound on the number of Markov states for any k-leaved rooted binary tree of arbitrary topology.

For $q(T) \gg 2$, the total number of states chiefly depends on whether or not the nodes n with $w(n) = 2$ are assigned "#", which stands for having a nucleotide. The number of states associated to such choices is

$$\begin{aligned}\sum_{m=0}^{q}\left[\binom{q}{m}\left(2^{k-2m} + \sum_{j=0}^{k-2m-1} 2^j\right)\right] &= 2^k \sum_{m=0}^{q}\binom{q}{m}\left(2^{-2m} + \frac{2^{k-2m}-2}{2^k}\right) \\ &\approx 2 \cdot 2^k \left(\frac{5}{4}\right)^q . \qquad (6)\end{aligned}$$

For $q(T) = \lfloor k/2 \rfloor$, i.e. in the case of a "balanced" tree, the expression in (6) is of order $O(\sqrt{5}^k)$.

C A Hidden Markov model

In many cases it is useful to obtain the maximum likelihood alignment using the Viterbi algorithm. This cannot be done with the one-state recursion, and we need to use the full Markov chain corresponding to the TKF91 model on a tree. We give two versions here. The first is not properly a Markov chain, in that the sum of exit probabilities from states do not sum to 1, but it does assign the right probability to full paths through the chain. This is enough for Viterbi to work. Secondly, we show how to turn this simpler chain into a proper Markov chain, which can be used for sampling alignments.

We have used the word 'state' in a different way than is usual for Markov chains. In this section, we write State with a capital letter if we use it in the Markov chain sense.

We view our chain as a Mealy machine [DEKM98], *i.e.* events and their emissions correspond to transitions, and paths through the chain correspond to histories. Lemma 3 gives a one-to-one correspondence between histories and paths through our chain. Given a sequence of events $e_0, \ldots, e_1$, a new event e_{i+1} is allowed if (e_i, e_{i+1}) is pairwise legal, and if condition $\Delta(i)$ holds. Both can be determined if we know the state S_{i+1} and the root of e_i^t, so as a first approximation to our Markov State we can use the pair (S_{i+1}, e_i^t). We

[2] In [HJP02], the root of a tree T is one of the internal nodes $n \in T$ of degree 3.

say that the nodes $r(t)$ and $r(t')$ are *directly related* if $t \cap t' \neq \varnothing$, *i.e.* one is the ancestor of the other, or they coincide. Then, the transition rule is that from State (S, v), event e is allowed if (1) $r(e^t) \in S$, and (2) $r(e^t)$ is directly related to v or $r(e^t) \geq v$. Now, if we set

$$S'_{i+1} := S_{i+1} \setminus \{n \,|\, n \text{ indirectly related to } r(e_i^t) \text{ and } n < r(e_i^t)\},$$

the transition rule simplifies to: e_{i+1} is allowed if $r(e_{i+1}^t) \in S'_{i+1}$, and it can be shown that the update rule

$$\begin{aligned} S'_{i+1} \quad := \quad & \left(S'_i \setminus (e_i^t \cup \{n \in S'_i \,|\, n \text{ indirectly related to } r(e_i^t) \text{ and } n < r(e_i^t)\}) \right) \\ & \cup \{n \in e_i^t \,|\, e_i^l(n) \neq -\} \end{aligned}$$

correctly updates S'_i to S'_{i+1}.

Let $p(e)$ be the probability factor associated to event e. The States S' and transition and update rules give rise to a graph on S', and we associate the probability factors $p(e)$ to the edges. To complete the graph, we introduce the initial and end States A and Ω. Furthermore, we introduce a single edge (with $p = \prod_{n \in T} I_n$) between A and the State T, and join each State except A to Ω by a single edge (with $p = 1$). In this way, the product of probability factors along any path from A to Ω is just the probability of the associated sequence of events, or history, under the TKF91 model. Since by our choice of States, each permissible history is counted exactly once, the probabilities sum up to 1.

For sampling, it is necessary to have a proper Markov chain, and the graph just defined can be turned into one by a scalar base change. Let P_S be the probability of being in State S, *i.e.* the sum of probabilities of all paths ending in S. Then, the state equation can be written as $P_S = \sum_{S'} P_{S'} t_{S'S}$, where $t_{S'S}$ is the transition matrix. If we introduce new variables $\hat{P}_S = f_S P_S$, then we can write $\hat{P}_S = \sum_{S'} \hat{P}_{S'} \hat{t}_{S'S}$, where $\hat{t}_{S'S} = t_{S'S} f_S / f_{S'}$. The graph becomes a Markov chain if $\sum_S \hat{t}_{S'S} = 1$. A tedious but straightforward calculation shows that $f_S = \left(\prod_{n \in S} I_n\right)^{-1}$ solves this equation, and the transition probabilities become

$$S_i \xrightarrow{e} S_{i+1} : \frac{f_{S_{i+1}}}{f_{S_i}} p(e),$$

where $f_A = f_\Omega = 1$.

References

[Chr96] D. A. Christie. Sorting permutations by block-interchanges. *Information Processing Letters*, 60:165–169, 1996.

[DEKM98] R. Durbin, S. Eddy, A. Krogh, and G. Mitchison. *Biological sequence analysis.* Cambridge University Press, 1998.

[Edd01] S. Eddy. HMMER: Profile hidden markov models for biological sequence analysis (http://hmmer.wustl.edu/), 2001.

[Fel81] J. Felsenstein. Evolutionary trees from dna sequences: a maximum likelihood approach. *J. Mol. Evol.*, 17:368–376, 1981.

[Fel01] J. Felsenstein. Phylip: phylogeny inference package. University of Washington, 2001.

[Gol98] N. Goldman. Effects of sequence alignment procedures on estimates of phylogeny. *BioEssays*, 20:287–290, 1998.

[HB01] I. Holmes and W. J. Bruno. Evolutionary HMMs: a Bayesian approach to multiple alignment. *Bioinformatics*, 17(9):803–820, 2001.

[Hei01] J. Hein. An algorithm for statistical alignment of sequences related by a binary tree. In R. B. Altman, A. K. Dunker, L. Hunter, K. Lauderdale, and T. E. Klein, editors, *Pac. Symp. Biocomp.*, pages 179–190. World Scientific, 2001.

[Hir77] D. S. Hirshberg. Algorithms for the longest common subsequence problem. *J. ACM*, 24:664–675, 1977.

[HJP02] J. Hein, J. L. Jensen, and C. N. S. Pedersen. Recursions for statistical multiple alignment. Technical Report 425, Dept. of Theor. Stat., Univ. of Aarhus, January 2002.

[HWK+00] J. Hein, C. Wiuf, B. Knudsen, M. B. Møller, and G. Wibling. Statistical alignment: Computational properties, homology testing and goodness-of-fit. *J. Mol. Biol.*, 302:265–279, 2000.

[JC69] T.H. Jukes and C.R. Cantor. *Mammalian Protein Metabolism*, chapter Evolution of protein molecules, pages 21–132. Academic Press, 1969.

[JH02] J.L. Jensen and J. Hein. Gibbs sampler for statistical multiple alignment. Technical Report 429, Dept. of Theor. Stat., Univ. of Aarhus, September 2002.

[Lee01] M. S. Y. Lee. Unalignable sequences and molecular evolution. *Trends in Ecology & Evolution*, 16:681–685, 2001.

[Mik02] I. Miklós. An improved algorithm for statistical alignment of sequences related by a star tree. *Bul. Math. Biol.*, 64:771–779, 2002.

[Mit99] G. Mitchison. A probabilistic treatment of phylogeny and sequence alignment. *J. Mol. Evol.*, 49:11–22, 1999.

[NW70] S. B. Needleman and C. D. Wunsch. A general method applicable to the search for similarities in the amino acid sequences in two proteins. *J. Mol. Biol.*, 48:443–453, 1970.

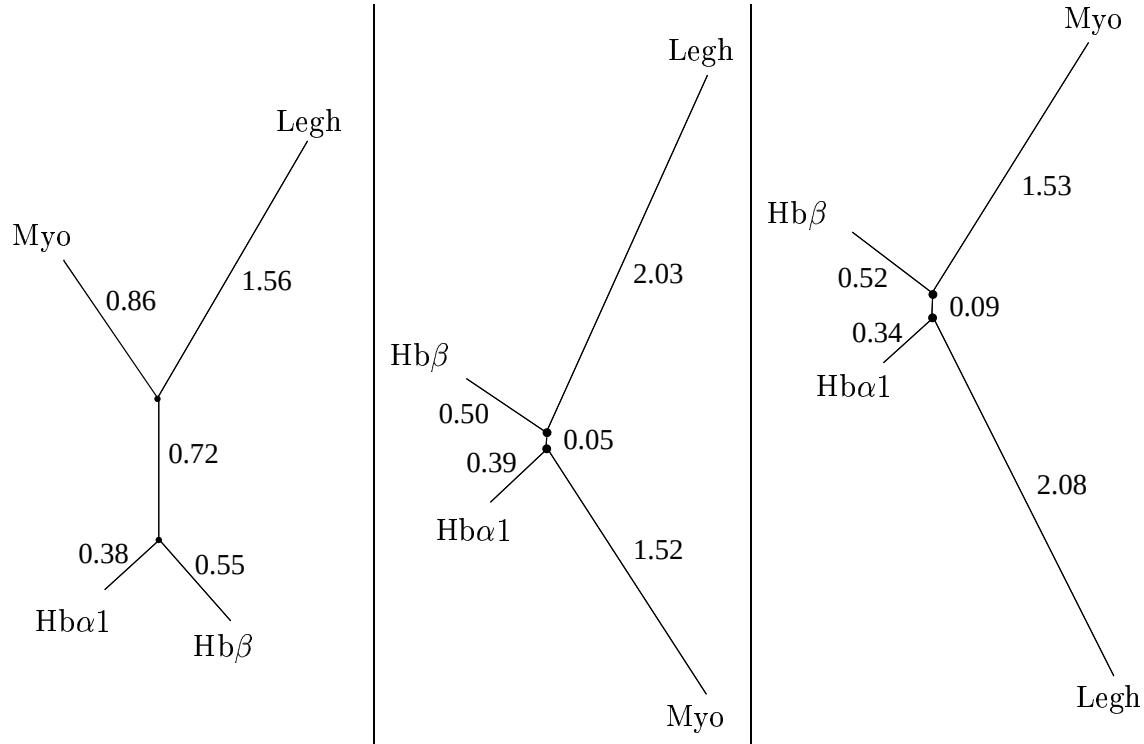
[SH01] M. Steel and J. Hein. Applying the Thorne-Kishino-Felsenstein model to sequence evolution on a star-shaped tree. *Appl. Math. Let.*, 14:679–684, 2001.

[SK83] D. Sankoff and J. B. Kruskall. *Time warps, string edits and macromolecules: the theory and practice of sequence comparison.* Addison-Wesley, 1983.

[TKF91] J. L. Thorne, H. Kishino, and J. Felsenstein. An evolutionary model for maximum likelihood alignment of DNA sequences. *J. Mol. Evol.*, 33:114–124, 1991.

[TKF92] J. L. Thorne, H. Kishino, and J. Felsenstein. Inching toward reality: an improved likelihood model of sequence evolution. *J. Mol Evol.*, 34:3–16, 1992.

[WLG01] S. Whelan, P. Lió, and N. Goldman. Molecular phylogenetics: state-of-the-art methods for looking into the past. *Trends in Genetics*, 17:262–272, 2001.



$-\log L = 1560.138$ $\lambda = 0.0334$	$-\log L = 1566.276$ $\lambda = 0.0339$	$-\log L = 1566.041$ $\lambda = 0.0314$
$\sigma(t_2 + t_3 + \lambda) = 0.12$	$\sigma(t_1 + t_2 + t_4 + \lambda) = 0.12$	$\sigma(t_1 + t_2 + t_4 + \lambda) = 0.12$
$\sigma(t_1 + t_4 + 2\lambda) = 0.22$	$\sigma(t_1 - 2t_3 + t_4) = 0.32$	$\sigma(t_1 + t_2 - 2t_4) = 0.21$
$\sigma(t_1 - t_5 - 7\lambda) = 0.47$	$\sigma(t_3 + 5\lambda) = 0.29$	$\sigma(2t_1 - 2t_2 + 3\lambda) = 0.50$
$\sigma(t_1 - 2t_4 + t_5 - 5\lambda) = 0.78$	$\sigma(t_1 + t_3 - t_4 + \lambda) = 0.38$	$\sigma(t_5 + 3\lambda) = 0.23$
$\sigma(t_2 - t_3 - \lambda) = 0.23$	$\sigma(t_3 - t_5 - 6\lambda) = 0.40$	$\sigma(t_3 - t_5 + 8\lambda) = 0.54$
$\sigma(t_5 - 5\lambda) = 0.54$	$\sigma(3t_5 - 4\lambda) = 1.1$	$\sigma(t_3 - 3\lambda) = 0.46$

Figure 3: Maximum likelihood trees relating human $\alpha 1$ and β hemoglobins, myoglobin, and bean leghemoglobin, for all three topologically distinct trees; total likelihood values and insertion rate; and estimated standard deviations. The numbers refer to the branch length in units of expected number of substitutions per amino acid. As substitution rate matrix we used Dayhoff's PAM matrix, normalized to 1 expected substitution event per unit time.

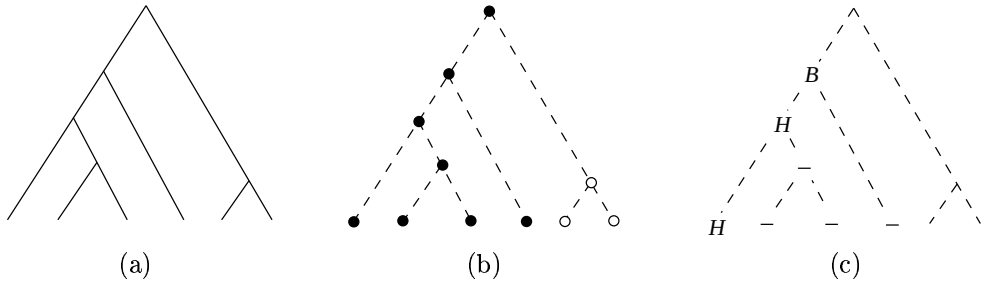


Figure 4: (a) A 6-leaved phylogenetic binary tree. (b) A possible state. Filled circles • (resp. open circles ∘) represent the nodes which are contained (resp. not contained) in the state. (c) A possible event. Any node without a label is not contained in the subtree corresponding to the event.

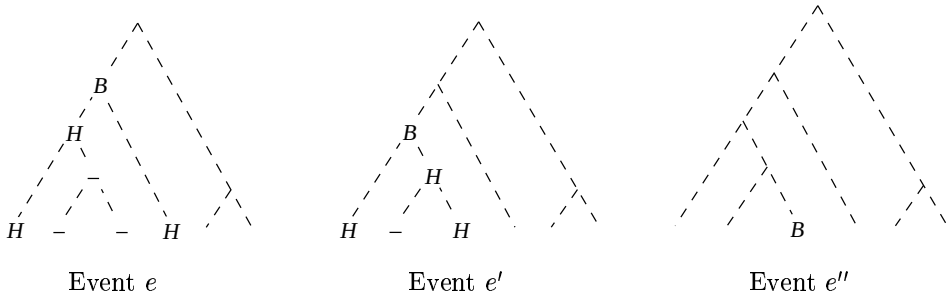


Figure 5: Three possible events.

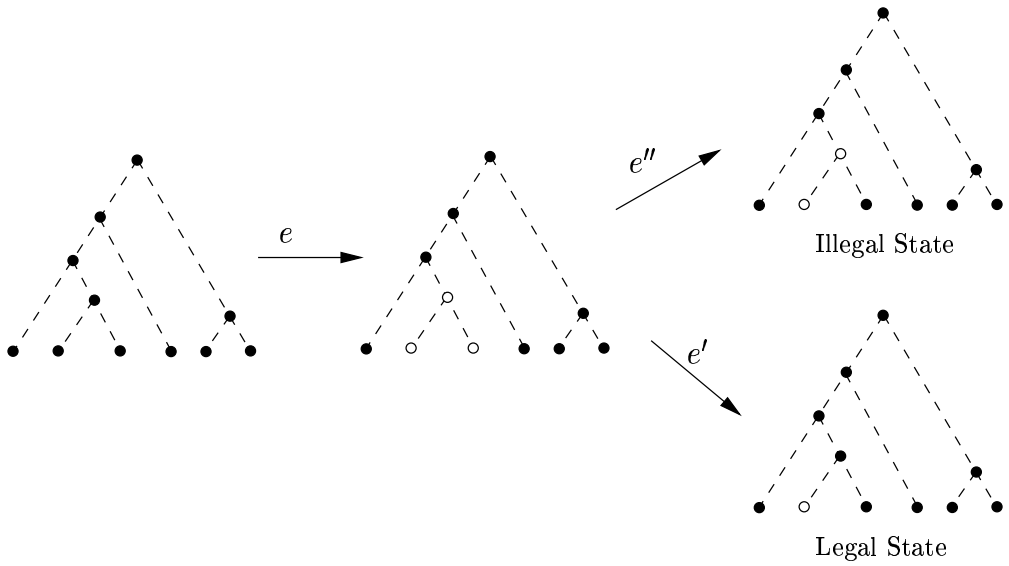


Figure 6: Sequences of states corresponding to the sequences (e, e') and (e, e'') of events, where e, e', e'' are as in Figure 5. The sequence (e, e') is a legal sequence, resulting is a legal state, whereas (e, e'') is an illegal sequence, giving rise to an illegal state. Note that (e, e', e'') is a legal sequence.

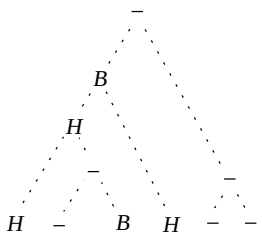


Figure 7: The tree labeling corresponding to the set of nested events $\{e, e''\}$, where the events e and e'' are as in Figure 5.

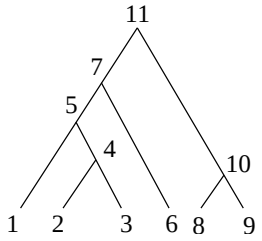


Figure 8: A 6-leaved tree with its nodes labeled according to our convention.