New Uses for Uniform Lifted Alignments [*]

Dan Gusfield and Lusheng Wang
Department of Computer Science
University of California, Davis, CA, 95616
e-mail: gusfield@cs.ucdavis.edu, lwang@cs.cityu.edu.hk

April 18, 1997

Abstract

The *phylogenetic alignment problem*, a.k.a. the *tree alignment problem*, arises in efforts to deduce histories of molecular evolution, and in certain methods to multiply align more than two sequences. The problem is known to be NP-hard, but several bounded-error approximation methods and polynomial time approximation schemes, have been developed for the problem [11, 8, 3]. The first of these approximation methods is based on what are called *lifted alignments*, and the second method is based on simpler *uniform lifted alignments*. The simplicity of uniform lifted alignments, compared to lifted alignments, allows a deeper study of their properties, and yet also gives a way to derive or compute results about lifted and optimal phylogenetic alignments. In this paper, we first prove the factor- of-two error bound on the optimal uniform lifted alignment differently than was previously done in [8]. Next we use uniform lifted alignments to establish error bounds on *random* lifted alignments. Finally, we use results about uniform lifted alignments to create an efficient algorithm to compute a non-trivial *lower bound* on the cost of the optimal solution to the phylogenetic alignment problem, given any problem instance. We use that lower bound to gauge the accuracy of a phylogenetic alignment computed by Sankoff et al. [6].

AMS Subject Classification: Primary 68Q25. Secondary 05C05, 68R15, 90C27, 90C39, 92B99, 92D15

1 Phylogenetic (tree) Alignment

Evolutionary history is frequently represented by an *evolutionary tree* where known (extant) organisms are represented at the leaves of the tree, and their unknown (but perhaps deduced) ancestors are represented at internal nodes of the tree. It is common now to deduce such evolutionary trees from molecular sequence data obtained from the organisms under study. However, the opposite direction of study is also possible. When the evolutionary tree is already known (from previous data and deductions), it can be used to deduce possible ancestral molecular sequences that gave rise to the extant sequences through a series of mutational events. This general problem has been called the *phylogenetic alignment problem* or the *tree alignment* problem, and has been formalized as the problem of deducing sequences at the internal nodes to minimize the cost given by an objective function defined below.

[*]Partially supported by Dept. of Energy grant DE-FG03-90ER60999.

In the above description, the tree with its deduced internal node labels is the desired output of the problem. However, once the labeled tree is in hand, one can also use it to find a multiple alignment of the extant sequences which is "influenced" by the hypothesized evolutionary history (see [7] or [2]. The details are a bit involved, and we only mention this application as additional motivation for the phylogenetic alignment problem. We will not discuss it further in this paper.

1.1 Formal definitions

Given a rooted tree T with a distinct string (from a set of strings $\mathcal{S}$) labeling each leaf, a *phylogenetic alignment* for T assigns one string to each *internal* node of T.

The rooted phylogenetic tree, T, is meant to represent the "established" evolutionary history of a set of taxa (read "objects"), with the convention that each extant taxon (object) is represented at a unique leaf of T. Each edge (u, v) represents some mutational history that transforms the string at u (assuming u is the parent of v) to the string at v. The cost of that transformation is a function of the two strings, and the cost of the phylogenetic alignment is the sum of all those edge costs. Note that the strings assigned to internal nodes need not be distinct and need not be from the input strings $\mathcal{S}$.

We let $D(S, S')$ denote the cost of transforming string S to string S'. All that is assumed about the function D is that it is a metric: $D(S, S') = 0$ if and only if $S = S'$, and D obeys the triangle inequality conditions, that if S, S' and S'' are any three strings, then $D(S, S') \leq D(S, S'') + D(S'', S')$. This is a natural and common assumption, requiring that the cost to directly transform S to S' be no greater than the cost of the indirect approach of transforming S to an intermediate string, and then transforming the intermediate to S'.

If strings S and S' are assigned to the endpoints of an edge (i, j), then (i, j) has *edge cost* $D(S, S')$. The cost of a path is the sum of the costs on the edges in the path. The cost of a phylogenetic alignment is the total of all the *edge* costs in the tree.

> **The phylogenetic alignment problem for a rooted tree T:** Find an assignment of strings to internal nodes of T (one string to each node) that *minimizes* the cost of the alignment.

The phylogenetic alignment problem was developed principally by Sankoff [4, 5], and was shown to be NP-hard by Wang and Jiang [9]. A factor-of-two approximation algorithm was developed by Jiang, Wang and Lawler [11] along with a polynomial time approximation scheme (PTAS). Both of these were based on *lifted alignments* which will be defined below. Later, using *uniform lifted alignments*, a faster factor-of-two approximation method was developed, and then exploited to obtain vastly faster PTASs with better error bounds [8, 10]. The simplicity (conceptual and computational) of uniform lifted alignments, compared to lifted alignments, was critical in developing these improved PTASs.

1.2 Lifted and Uniform lifted alignments

A phylogenetic alignment is called a *lifted alignment* if for every internal node v, the string assigned to v is also assigned to one of v's children (see Figure 1). It is immediate that each node v in a lifted alignment is assigned a string that labels a leaf in the subtree rooted at v, and hence is a string from the input set $\mathcal{S}$.

In order to specify particular lifted alignments, each node of T is given a unique name. Then any particular lifted alignment is determined by specifying, for each node v, the name of the child of v from whom v's assigned string is "lifted" or "inherited" ("The child is the father of the man").

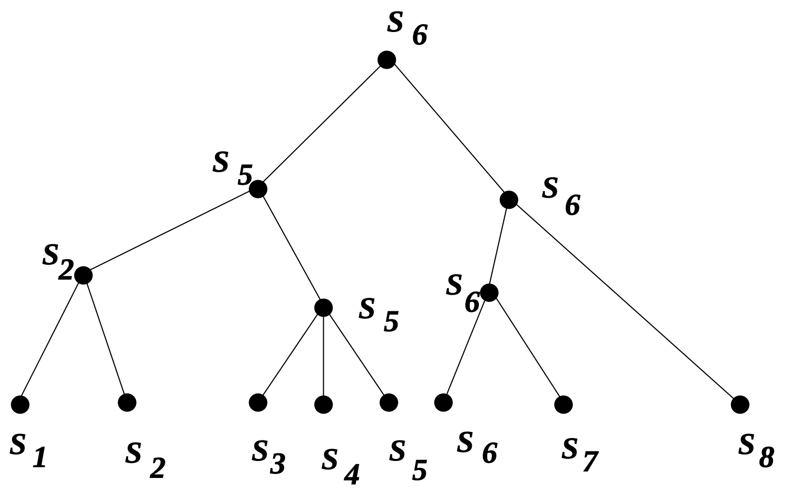

Figure 1: An abstract lifted alignment.

To define a *uniform lifted alignment* we assume (for simplicity of the discussion) that tree T is *binary* so that each internal node has two children, although T need not be balanced. For each internal node v, arbitrarily choose one of its children as the left child $l(v)$ and the other as the right child $r(v)$. Such a choice is called a *layout* of T. The nodes of T are partitioned into levels, numbering from the bottom of the tree upwards, starting from level zero. That is, the leaves farthest from the root are at level zero, and the root is at the largest level. See Figure 2.

Given a fixed layout of T, a lifted alignment is called a *uniform lifted* alignment if, at each level, either *every* internal node at that level receives its assigned string from its left child, or *every* internal node at that level receives its assigned string from its right child. See Figure 2. Note that a uniform lifted alignment is only defined with respect to a specific layout of T.

Clearly, any lifted alignment is a uniform lifted alignment for some layout(s) of T, and a lifted alignment remains a phylogenetic alignment in every layout of T. However, when T has even a single level with more than one internal node, then for every lifted alignment of T, there is a layout of T for which the lifted alignment is *not* a uniform lifted alignment. Thus (with one exception), for a given layout of T the set of uniform lifted alignments is a proper subset of the set of lifted alignments. This subset can be substantially smaller, as established in the next lemma. Define the depth of tree T to be the level of the root node.

Lemma 1.1 *Let T be a binary tree with depth d and n leaves. Then T has 2^{n-1} lifted alignments, but for any fixed layout of T, it has only 2^d uniform lifted alignments.*

Proof A lifted alignment is determined by specifying for each internal node v, which of v's two children will have the same assigned string as v. These choices are independent for each internal node, and since there are $n-1$ internal nodes in a binary tree with n leaves, there are 2^{n-1} lifted alignments of T.

Once a layout of T is fixed, a uniform lifted alignment of T is determined by specifying for each level, whether the internal nodes of that level get assigned the strings from their left children

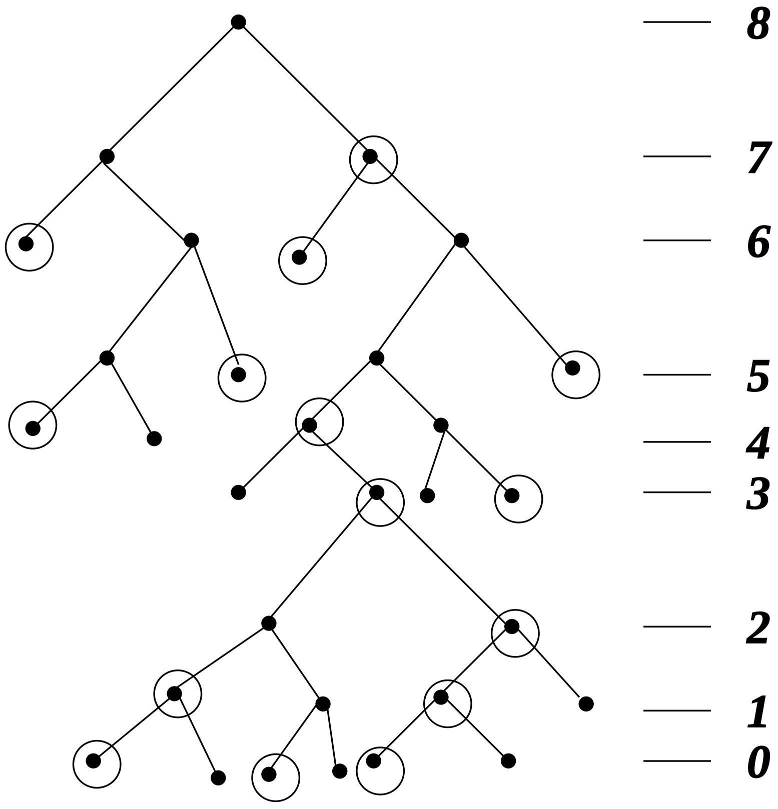

Figure 2: An abstract uniform lifted alignment. The lines and numbers on the right indicate the levels of the tree. Each circle represents a node whose assigned string is lifted to its parent node. At each level, all the lifted strings come from the left children, or all come from the right children.

or from their right children. The choice is independent at each level, hence there are exactly 2^d uniform lifted alignments for that layout. □

For example, when T is balanced and full, there are only n uniform lifted alignments for a fixed layout, while there are 2^{n-1} lifted alignments for T. This is the extreme case, where (for any fixed layout) the number of uniform lifted alignments is much smaller than the number of lifted alignments.

For any problem instance, let T^* denote the optimal phylogenetic alignment, and let $C(T^*)$ denote its cost. In [11] it was shown, for any problem instance, that the optimal lifted alignment (the lifted alignment with lowest cost) has cost less than $2C(T^*)$. Somewhat surprisingly, it was subsequently shown [8] that for *any fixed layout* of T, the optimal *uniform* lifted alignment (the uniform lifted alignment with lowest cost) also has cost less than $2C(T^*)$. This is surprising, because in general (for a fixed layout) the set of uniform lifted alignments is much smaller, and more highly constrained, than the set of lifted alignments. Both the optimal lifted alignment, and the optimal uniform lifted alignment (for a fixed layout) can be computed in polynomial time. The results for the optimal lifted alignment do not depend on the tree being binary. The results for optimal uniform lifted alignment can be generalized in different ways to non-binary trees, depending on the specific definition of a uniform lifted alignment for non-binary trees. Such extensions should be clear after discussing the error and time bounds for binary trees.

1.3 The basic error bound

We now prove that for a fixed layout of T, the cost of the optimal uniform lifted alignment is less than $2C(T^*)$. This proof is different from the original one in [8], and is based on comments by Mike Paterson about optimal lifted alignments. We will assume, for exposition purposes, that T is binary.

We prove the claimed error bound by exhibiting a particular uniform lifted alignment T^U whose cost is within the claimed bound. For any node v, let S^*_v denote the string assigned to v in T^*. We will transform T^* into a uniform lifted alignment T^U (for the fixed layout) by a series of string replacements at internal nodes of T. This transformation is only conceptual, since we do not know T^*.

The transformation of T^* to T^U is done one level at a time, bottom up. Let $V(k)$ be the internal nodes at level k. To assign strings at level k (after the level $k-1$ beneath it has been transformed) consider two sums: $\sum_{v \in V(k)} D(S^*_v, S_{l(v)})$ and $\sum_{v \in V(k)} D(S^*_v, S_{r(v)})$, where $S_{l(v)}$ and $S_{r(v)}$ are respectively the strings that have been assigned in the transformation to the left and right children of v (in the fixed layout). If the first sum is smaller or equal to the second sum, then assign each internal node v in level k the string $S_{l(v)}$; otherwise assign each internal node in level k the string $S_{r(v)}$. See Figure 3. When all levels have been transformed, the resulting lifted alignment T^U is a uniform lifted alignment. For any internal node v, let S_v denote the string assigned to node v in alignment T^U.

The error analysis

Theorem 1.1 *The uniform lifted alignment T^U has total cost less than $2C(T^*)$.*

Proof With respect to T^U, a node w is called an *orphan* if the string, S_w (assigned to w in T^U) is different from the string assigned to w's parent node. By definition of a lifted alignment,

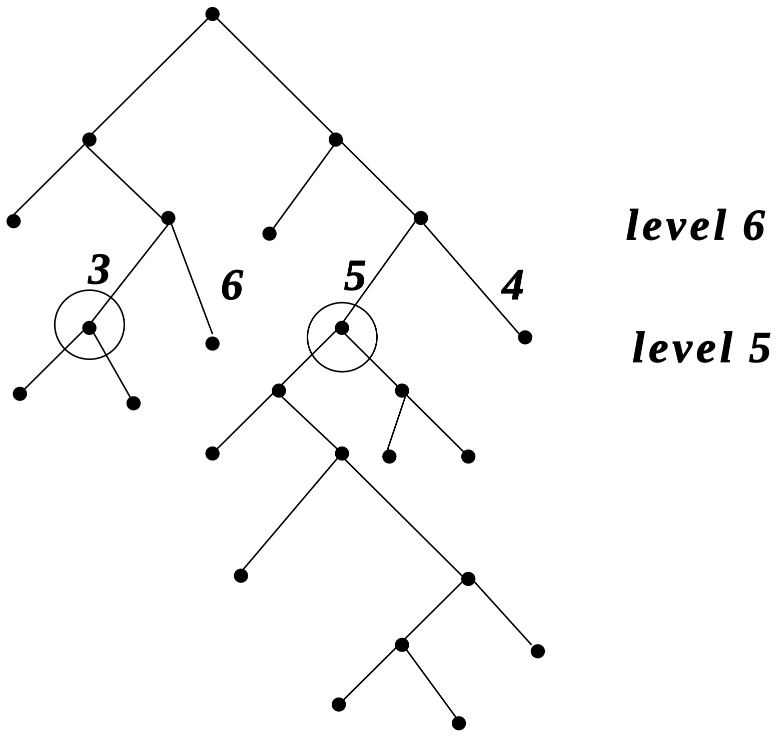

Figure 3: The lifting operation from level 5 to level 6. The numbers on the edges are the distances used to decide whether to lift from the left children or to lift from the right children. The sum for the left children is 8, and the sum for the right children is 10, so the lift is from the left children.

there is a unique leaf z of T labeled with string S_w, and every node on the path from w to z is assigned string S_w in T^U. Moreover, if w is an orphan, then no node off the path from w to z can be assigned string S_w. Let P_w denote that path from orphan w to leaf z, and call it an *orphan path*. It follows that if w and w' are two orphans in T^U, then the orphan paths P_w and $P_{w'}$ have no nodes in common. So, over the set of all orphan paths in T^U, no edge of T appears in more than one orphan path.

For any internal node v, let $o(v)$ denote the unique orphan child of v in T^U. Consider any internal node v, and define P^v as the path consisting of the edge $(v, o(v))$ followed by the orphan path $P_{o(v)}$. Call this path the *semi-orphan* path of v. For distinct v and v', P^v and $P^{v'}$ have no *edges* in common, although they may share one node. Let z_r be the leaf whose label is lifted and assigned to the root node r in T^U. No edge on the path from r to z_r is on any semi-orphan path. Therefore, over all internal nodes, the set of semi-orphan paths are *edge disjoint* and omit some edge of T. See Figure 4. Note that by triangle inequality, the total cost in T^* of the edges on the semi-orphan path P^v is at least $D(S_v^*, S_{o(v)})$, since the edges of that path describe one (rather long and roundabout) way to transform S_v^* to $S_{o(v)}$.

For any internal node v, let $h(v)$ (for "heir of v") be the unique sibling of $o(v)$. By definition, $S_v = S_{h(v)}$, so the edge between v and $h(v)$ has cost zero in T^U. Therefore the cost of T^U is exactly $\sum_v D(S_v, S_{o(v)})$. By triangle inequality, and the definition of an heir, $D(S_v, S_{o(v)}) \leq D(S_v, S_v^*) + D(S_v^*, S_{o(v)}) = D(S_{h(v)}, S_v^*) + D(S_v^*, S_{o(v)})$.

Now consider a level k in T. The cost in T^U of all the edges between level k and level $k-1$ is exactly $\sum_{v \in V(k)} D(S_v, S_{o(v)}) \leq \sum_{v \in V(k)} D(S_{h(v)}, S_v^*) + D(S_v^*, S_{o(v)})$. It then follows, by the selection rule used to transform T^* to T^U, that the cost of the edges between level k and level $k-1$ is at most $2\sum_{v \in V(k)} D(S_v^*, S_{o(v)})$. But, since $D(S_v^*, S_{o(v)})$ is at most the total cost of all the edges on P^v, the cost of all edges between levels k and $k-1$ is at most twice the cost of all the edges on semi-orphan paths originating from nodes at level k. Since this holds for every level, and all semi-orphan paths (from anywhere in T) are pairwise edge disjoint, the cost of all edges in T^U is at most twice the cost of all edges in T^*. More precisely, since no edge on the path from r to z_r is on any semi-orphan path, the above argument establishes that the cost of T^U is less than or equal to $2C(T^*) - 2D(S_r, S_r^*)$.

Now we prove that the cost of T^U is *strictly* less than $2C(T^*)$. This would be immediate if $S_r \neq S_r^*$, for then $D(S_r, S_r^*) > 0$. If $S_r = S_r^*$, then $D(S_r, S_{o(r)}) = D(S_r^*, S_{o(r)}) > 0$ and the cost of T^U is less than or equal to $2C(T^*) - D(S_r^*, S_{o(r)})$. To see this, note that in establishing the bound of $2C(T^*)$, we bounded the cost in T^U of the edges touching the root by $2D(S_r^*, S_{o(r)})$, when in fact the cost is only $D(S_r^*, S_{o(r)})$. Hence the cost of T^U is strictly less than $2C(T^*)$. $\Box$

Since T^U is a uniform lifted alignment, the main result below now follows.

Theorem 1.2 *For any layout of T, the cost of the optimal uniform lifted alignment is less than $2C(T^*)$.*

2 Studying lifted alignments via uniform lifted alignments

In this section we show that the use of uniform lifted alignments allow simple derivations of facts about lifted alignments that were not apparent from direct examination of the full set of lifted alignments. The key observation is that (easily derived) facts about the distribution of costs of

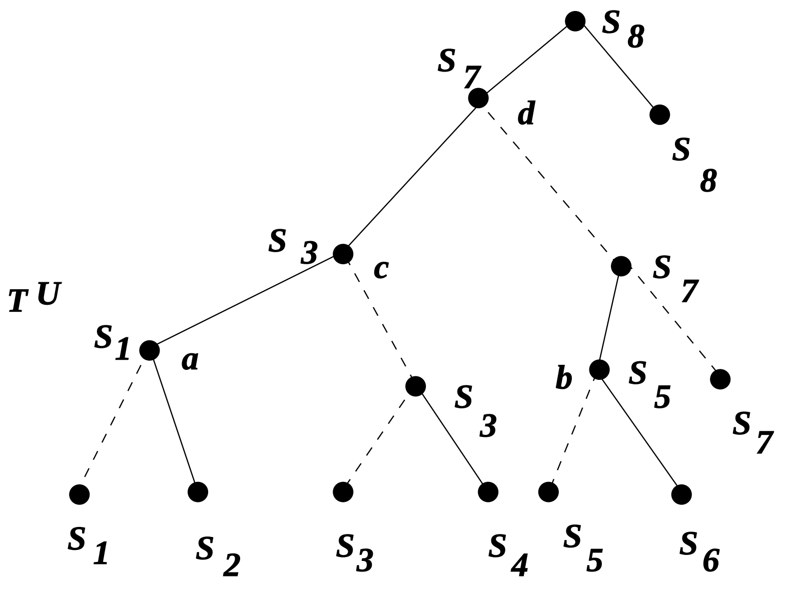

Figure 4: Lifted tree T^U. The internal nodes labeled a, b, c, and d are all orphans. Their respective orphan-paths are shown with dashed lines.

uniform lifted alignments hold also for the distribution of costs of lifted alignments. We start with a result that illustrates the key idea.

Theorem 2.1 *In a binary tree, at least $\frac{1}{2^{d-1}}$ of all lifted alignments have a cost less than $2C(T^*)$.*

Proof Consider counting, over all fixed layouts of T, the number of uniform lifted alignments whose cost is less than $2C(T^*)$. For any fixed layout, T^U is one such uniform lifted alignment. Let $T^{U'}$ be the uniform lifted alignment obtained by relabeling the root of T with the sequence $S_o(r)$ (as defined in the creation of T^U). Clearly, the cost of $T^{U'}$ is the same as the cost of T^U. Since there are 2^{n-1} fixed layouts, there must be at least 2^n lifted alignments whose cost is less than $2C(T^*)$. But each lifted alignment is a uniform lifted alignment for only 2^d layouts, so there must be at least $2^n/2^d = 2^{n-d}$ *distinct* lifted alignments whose cost is less than $2C(T^*)$. Since there are 2^{n-1} distinct lifted alignments, the fraction of all lifted alignments whose cost is less than $2C(T^*)$ is at least $\frac{1}{2^{d-1}}$. $\square$

For a completely unbalanced binary tree with n leaves, Theorem 2.1 only guarantees two out of 2^n lifted alignments have cost less than $2C(T^*)$. But for a *balanced, full* binary tree, at least two out of n of all the lifted alignments have a cost less than $2C(T^*)$. In realistic cases, n can be as small as ten, and 2^d is often close to n, and the theorem establishes that lifted alignments with cost less than $2C(T^*)$ are fairly dense in those cases.

Theorem 2.1 does depend on T being binary (which is the most important and realistic case), but the theorem, and all the subsequent material in this paper, can be easily generalized to cover non-binary trees. For example, if the tree is tertiary, then at least $\frac{1}{3^d}$ of all lifted alignments have a cost no greater than $2C(T^*)$.

The style of analysis contained in the proof of Theorem 2.1 can be used again to establish a bound on the average cost of all lifted alignments. It was shown in [8] that for any fixed layout of T, the *average* cost of the uniform lifted alignments (for that layout) is less than $2C(T^*)$.

Theorem 2.2 *The average cost of all lifted alignments for T is less than $2C(T^*)$.*

Proof Consider summing, over all fixed layouts of T, the costs of all the uniform lifted alignments for that layout. The average cost for any fixed layout is less than $2C(T^*)$ and there are exactly 2^d uniform lifted alignments for each fixed layout, so each fixed layout contributes at most $2^{d+1}C(T^*)$. Since there are exactly 2^{n-1} layouts, the total sum is less than $2^{d+1}C(T^*)2^{n-1}$. Now each lifted alignment contributes its cost to that sum exactly 2^d times, and there are exactly 2^{n-1} distinct lifted alignments, so the average cost of all lifted alignments is less than $2C(T^*)$. $\square$

More generally, the proof of Theorem 2.1 can be easily extended to establish the following

Theorem 2.3 *Fix an arbitrary layout of T. Consider any property that holds for some fraction f of all the uniform lifted alignments of that fixed, but arbitrary, layout. Then that property holds for fraction f of all lifted alignments.*

Using either Theorem 2.3 or 2.2 we can obtain the following bounds on the distribution of costs of random lifted alignments.

Theorem 2.4 *For any $r > 1$, define $e(r)$ to be the expected number of lifted alignments needed to be chosen at random before the smallest cost of all those alignments is within a factor of $2+1/(r-1)$ of the cost of the optimal phylogenetic alignment. Then $e(r) \leq r$.*

For example, $e(r)$ is at most two for an error bound of 3, and $e(r)$ is at most ten for a bound of 2.1112. Note that $e(r)$ is independent of n and of the lengths of the strings. Also note that the above theorem holds when restated to apply only to uniform lifted alignments. Another way to state Theorem 2.4 is: Let $k(\epsilon)$, for any $\epsilon > 0$, be the expected number of lifted alignments to draw at random to find one of cost at most $(2+\epsilon)C(T^*)$. Then $k(\epsilon) \leq 1 + \frac{1}{\epsilon}$.

Proof For $r = 2$, the theorem says that at least half of all the lifted alignments must have cost less than or equal to $3C(T^*)$. This follows immediately from the fact that the average cost is at most $2C(T^*)$ and the minimum cost is $C(T^*)$. Generalizing, at least $1/r$ of all the lifted alignments must have cost less than or equal to $(2r-1)C(T^*)/(r-1)$, which again follows from the fact that the minimum possible cost is $C(T^*)$ and the mean is at most $2C(T^*)$. $\square$

Stating this result in terms of probabilities rather than expectations, we have the following

Theorem 2.5 *Picking p lifted alignments at random, the minimum cost phylogenetic alignment of those p alignments will have cost within a factor of $2 + 1/(r-1)$ of the optimal phylogenetic alignment, with probability at least $1 - [(r-1)/r]^p$.*

Theorem 2.4 and Theorem 2.5 are analogous (and proven in related ways) to results from [1] about random multiple alignments under a different objective function (the sum-of-pairs objective function).

It is worth noting that the analysis presented above is very loose, and therefore the results are overly pessimistic. For example, in Theorem 2.4 the case of $r = 2$ was proven by observing that the median can be at most $3C(T^*)$. The conclusion was that at least half of all random lifted alignments must have an error bound at most three times the optimal. But if the median were actually $3C(T^*)$, then exactly half of the random lifted alignments must have cost $C(T^*)$, i.e., they must be optimal phylogenetic alignments.

3 Using uniform lifted alignments to compute lower bounds

The optimal (uniform or not) lifted alignment is guaranteed to have cost less than twice that of the optimal phylogenetic alignment on any problem instance, but one expects that these, and other, methods will find closer to optimal solutions for most problem instances. Therefore it is desirable to have efficient methods to compute *lower bounds* on the cost of the optimal phylogenetic alignment, given any problem instance. These lower bounds can be applied not only to gauge the accuracy of the result computed by the approximation method, but by any method. Often one can modify a bounded-error approximation method, or exploit the ideas behind its error analysis, to obtain an efficient method to compute non-trivial lower bounds given a problem instance. When combined with better (but unanalyzed) heuristic methods, this can be a valuable and practical use of bounded-error methodology. In this section we show how to use ideas from lifted and uniform lifted alignments to compute a non-obvious lower bound on the cost of the optimal phylogenetic alignment.

For a fixed layout of T, let A_u denote the average cost of a uniform lifted alignment for that layout. Since $A_u < 2C(T^*)$ (shown in [8]), $A_u/2$ is a *lower bound* on $C(T^*)$. Moreover, it is a particularly appealing lower bound because it holds for *every* layout, and there are 2^{n-1} distinct layouts. Therefore, an attractive strategy is to randomly pick several layouts of T and compute A_u for each one. The maximum A_u obtained this way, divided by two, is then a lower bound for $C(T^*)$. We will later improve this approach, to obtain even higher lower bounds, but first we show that A_u can be computed in $O(n^2)$ time for any layout.

3.1 Computing the cost of the average uniform lifted alignment

We continue to assume that T is binary. We also assume that the distance between each pair of leaf sequences is already known.

Recall that there are 2^d uniform lifted alignments for a fixed layout of T. When T is a full binary tree, then $2^d = n$, the number of leaves, and we can trivialy generate the n uniform lifted alignments and compute the cost for each one. This direct approach takes $O(n^2)$ time. However, when T is very unbalanced, d can be as large as n so a full enumeration would take exponential time as a function of n. None the less, it is possible to compute the average of the 2^d uniform lifted alignments in $O(n^2)$ time.

For a fixed layout of T, we define an ordered pair of leaf sequences (S_i, S_j) to be *legal* for an edge (u, v) (where u is the parent of v), if there is a uniform lifted alignment in which S_i is assigned to u and S_j is assigned to v. Below we will see how to find all legal pairs, and their associated edges, in $O(n^2)$ time. In fact, when a pair (S_i, S_j) is found to be legal for an edge (u, v), the algorithm will determine the exact number of uniform lifted alignments of the fixed layout, in which S_i is assigned to u and S_j is assigned to v.

It is simple to test if an ordered pair (S_i, S_j) is legal for some edge. Consider the two paths up to the root node from the leaves labeled S_i and S_j. Those two paths join at the least common ancestor of S_i and S_j, say at level l. Assume, w.l.o.g. that S_i labels leaf at a level k at or above the leaf labeled by S_j. Then (S_i, S_j) is legal for some edge if and only if the two paths are "parallel" from level k to level l. That is, at each level between k and $l-1$, the nodes on the two paths must both be the left children of their respective parents, or they must both be the right children of their parents. If the paths have the required property, then the ordered pair (S_i, S_j) is legal for one edge out of their least common ancestor (denoted u). Moreover, the ordered pair (S_j, S_i) is legal for the other edge out of u.

Any pair of leaves can be tested in $O(d)$ time, yielding an $O(dn^2)$ time method. The time can be reduced to $O(n^2)$ by reversing the direction of the walks and organizing them with depth-first traversals. In detail, to find all legal pairs, repeat the following algorithm for each edge (u, v) of T, where, w.l.o.g., v is the right child of u, i.e., $v = r(u)$. Execute a *parallel depth-first traversal* from v and $l(u)$ until one of the searches reaches a leaf (see Figure 5). In a parallel depth- first traversal, the two traversals alternate single edge moves, and the first one moves from any level to its right (left) child, if and only if the second one next moves from the same level to its right (left) child. When one of the traversals (the first say) reaches a leaf labeled say with S_i and found on level k, the second traversal continues below level k in a normal depth- first fashion, until it returns to level k. At that point the two traversals begin alternating again in parallel fashion. Let S_j be a leaf sequence that the second traversal encounters before the two traversals return to parallel mode. That is, S_j is a leaf sequence encountered before the second traversal backs up from level k. Then (S_i, S_j) is a legal pair for edge (u, v) because the path from $l(u)$ to S_i parallels the part of the path from v to S_j down to level k.

Let $N(i, j)$ denote the number of uniform lifted alignments of the layout where S_i is assigned to u and S_j is assigned to v. Then the ordered pair (S_i, S_j) contributes exactly $N(i, j)D(S_i, S_j)$ to the sum of the costs of all the uniform lifted alignments of that layout. (The ordered pair (S_j, S_i) will also contribute the same amount to the sum.) But what is $N(i, j)$? Suppose u is on level l and S_j is on level $l' \leq k$. Then $N(i, j) = 2^{d-l+l'}$. The reason is that for every level from $l - 1$ down to l', the choice of which child (left or right) assigns its sequence to its parent, is fixed by the requirement that S_i be successively assigned (lifted) up to node u and S_j be lifted up to node v. However, the choices at the other levels can be made in every possible way, and are independent at each level. In general, if (S_i, S_j) is legal for (u, v), then the choices are fixed for all levels between node v and the lower of the two leaves labeled by S_i and S_j.

Let $L(T)$ be the set of legal (ordered) pairs for the fixed layout of T. In summary, the total cost of all the uniform lifted alignments of T (for the fixed layout) is $\sum_{(S_i,S_j)\in L(T)} N(i, j)D(S_i, S_j)$. The average cost is that sum divided by 2^d. Assuming the distance between each pair of leaf sequences is known, the average cost of all uniform lifted alignments can be found in $O(n^2)$ time.

We should note that the average cost of all *lifted* alignments can also be computed in $O(n^2)$ time, and half that average is again a lower bound on $C(T^*)$. But it is not desirable to compute only that single lower bound, since one gets a lower bound from the uniform lifted alignments of *each* fixed layout. The average of all those 2^{n-1} lower bounds is the lower bound obtained from the set of all lifted alignments, so some of those lower bounds will be better and some worse than the single one obtained from all lifted alignments. Therefore, one should compute the average cost of the uniform lifted alignments for several randomly selected layouts (and also the average over all lifted alignments), and then take the maximum of those bounds. That approach exploits any *variance* there may be between the average costs for different fixed layouts. We should also note that half the cost of the optimal lifted alignment is a lower bound on $C(T^*)$. Similarly for a fixed layout, half the cost of the optimal uniform lifted alignment is a lower bound on $C(T^*)$. But both of these bounds are inferior to the bounds derived from the respective averages.

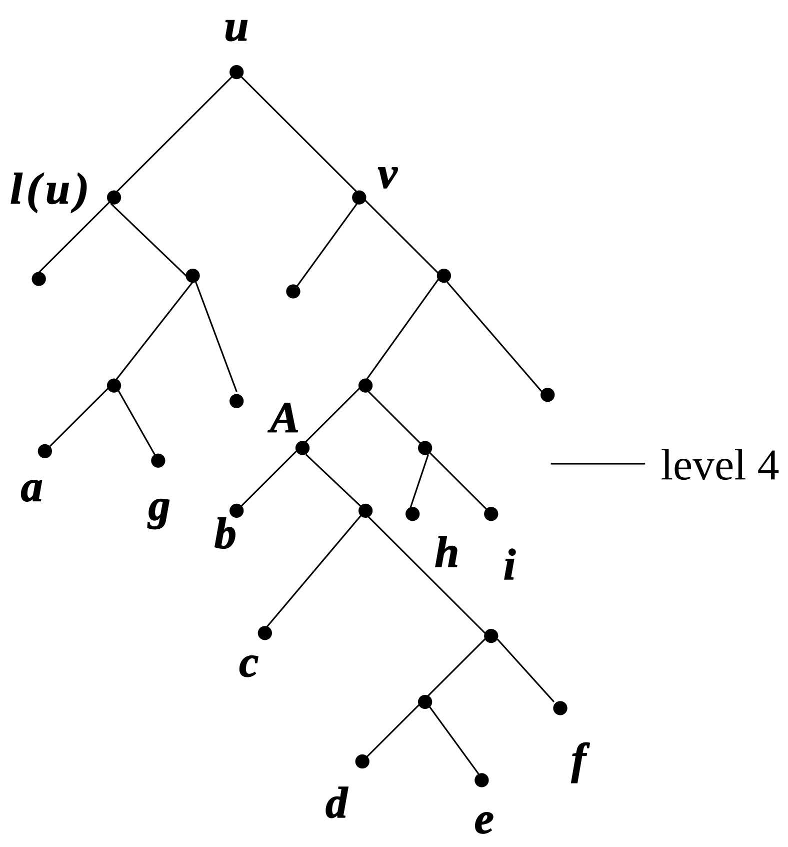

Figure 5: Suppose the parallel depth-first traversal initially walks from $l(u)$ to leaf a, and from v to node A. Then a standard depth-first traversal from A will find that the sequence at leaf a forms a legal pair for (u, v) with each of the sequences at leaves b, c, d, e and f. Then the traversal backs up from a and A, and next determines that (g, h) and (g, i) form legal pairs for edge (u, v).

3.2 Improved lower bounds

The lower bounds discussed above are based on the fact that the average lifted (or uniform lifted) alignment has cost less than $2C(T^*)$. However, as shown in the proof of Theorem 1.1, the cost of T^U is at most $2C(T^*)$ minus twice the cost of some full path to the root of T^*. In the case of the average of all lifted (or uniform lifted) alignments, a similar savings occurs due to legal pairs whose least common ancestor is the root of the tree. This leads to the following improvements in the lower bounds.

Theorem 3.1 *Given a rooted tree T, let L_A be the set of (ordered) legal pairs whose lca is not the root of T, and let L_B be the set of legal pairs whose lca is the root. Let $r(i,j)$ be the number of edges on the path between the lca of leaf i and leaf j, and lowest of the leaves i or j. Then $C(T^*) \geq [\sum_{(i,j)\in L_A} 2^{d-r(i,j)} D(i,j) + \sum_{(i,j)\in L_B} 2^{d-r(i,j)+1} D(i,j)]/(2^{d+1})$.*

We can get a similar improvement based on all the lifted alignments. Let m be the number of non-leaf nodes in T, and for any pair of leaves (i,j), let $m(i,j)$ be the number of non-leaf nodes on the path between i and j.

Theorem 3.2 *Given a rooted tree T, let A be the set of ordered leaf pairs whose least common ancestor (lca) is not the root of T, and let B be the set of ordered leaf pairs whose lca is the root. Then, $C(T^*) \geq [\sum_{(i,j)\in A} 2^{m-m(i,j)} D(i,j) + \sum_{(i,j)\in B} 2^{m-m(i,j)+1} D(i,j)]/(2^{m+1})$.*

Clearly, both of these lower bounds can be again computed in $O(n^2)$ time, assuming that the distances are known between each pair of leaf sequences.

3.3 Experimental Work

We experimented with the above ideas on a well known, fifteen node tree, a phylogenetic alignment problem first studied by Sankoff, Cedergren and Lapalme [6]. In that paper, they produced a phylogenetic alignment (we will call Sankoff's alignment) that is not known to be optimal for the data, but is believed to be close to optimal. He wanted to use the above ideas to establish the lower best bounds for the problem instance and for Sankoff's alignment. We also wanted to see how close the best uniform lifted alignment is to the best lifted alignment.

The given evolutionary tree T is shown in Figure 6. Each leaf is assigned an RNA sequence of length around 120 nucleotides. We use the same scoring scheme as in [6], *i.e.*, $D(A,C) = 1.75$, $D(A,G) = 1.0$, $D(A,U) = 1.75$, $D(C,G) = 1.75$, $D(C,U) = 1.0$, $D(G,U) = 1.75$, and 2.25 for insertion/deletion. This scoring scheme clearly satisfies triangle inequality.

Before describing the experimental results, we need to mention one additional modification to the lower bound methods. The tree T used in [6] is unrooted, so in order to apply Theorems 3.1 and 3.2 we select an edge of T and create a new node on it, making that new node the root of the tree. It is easy to see that this does not change the cost of the optimal alignment, but the optimal lifted and optimal uniform lifted alignments do depend on the root choice. Further, since Theorems 3.1 and 3.2 treat leaf pairs whose lca is the root differently than leaf pairs whose lca is not the root, the positioning of the root might change the lower bounds based on those theorems. Every choice leads to a correct lower bound, but some choices might give higher bounds than others. In fact, this happens when using Theorem 3.1, but not when using Theorem 3.2, as we will show next.

Theorem 3.3 *Let T be an unrooted tree. No matter what edge is chosen for the new root, the lower bound on $C(T^*)$ based on Theorem 3.2 is $\sum_{i<j} D(i,j)/2^{m(i,j)}$, where $m(i,j)$ is the number of non-leaf nodes on the path from leaf i to leaf j in T before the addition of the root node.*

Proof The lower bound is obtained by applying Theorem 3.2 to T after the addition of the root node. Let $m' = m + 1$, and let $m'(i,j)$ be the number of non-leaf nodes on the path from i to j after the root is added. Then $m'(i,j) = m(i,j)$ if the lca of i and j is not the root, and $m'(i,j) = m(i,j) + 1$ if the lca is the root. Applying Theorem 3.2 with m' and $m'(i,j)$, and then simplifying the summation completes the proof. □

Hence, in the case of an unrooted tree, computing the lower bound based on Theorem 3.2 is particularly simple, and can easily be done by hand.

3.3.1 Experimental Results

The results from these limited experiments are that the optimal uniform lifted alignments (taken over all choices for root, but only a single layout of T for each choice) have small variance, and have costs fairly close to the optimal lifted alignment. The average costs have somewhat greater variance. The highest lower bound computed in this way establishes that Sankoff's alignment has cost at most 19.85% greater than the optimal phylogenetic alignment.

In more detail, the cost of Sankoff's alignment is 295.5. The lowest cost of an optimal lifted alignment (over varying choices for the root) was 364.0, while the highest cost of an optimal lifted alignment was 393.5. The lowest cost of an optimal uniform lifted alignment (over varying choices for the root, but a fixed layout for each choice) was 371.5, while the highest cost of an optimal uniform lifted alignment was 396.5. The highest average cost of a lifted alignment was 460.3, and the lowest average cost was 396.4. The higest average cost of a uniform lifted alignment was 461.6, and the lowest average cost was 398.2. The highest lower bound based on Theorem 3.1 (over varying choices for the root, but a fixed layout for each choice) was 244.9023, which establishes that Sankoff's alignment deviates from the optimal by no more than 19.85%. The highest lower bound based on Theorem 3.3 was 242.86, which establishes a deviation from optimal of no more than 20.86%. In this small experiment, the results obtained from using only uniform lifted alignments were not much different than the results based on all lifted alignments. We expect there would be a greater distinction in larger trees.

DG the corrected lbu value is 242.191 which establishes a deviation of 21.186%.

3.4 Comparison to other lower bounds

There are two other lower bounds that have been suggested for the phylogenetic alignment problem. One is based on computing a minimum spanning tree, and the other is based on a linear programming relaxation of the phylogenetic alignment problem.

For the minimum spanning tree bound, compute the distance between each pair of leaf sequences; form a complete graph on n nodes (one node v_i for each leaf sequence S_i); and set the cost of the edge between any pair of nodes v_i and v_j as $D(S_i, S_j)$; finally, compute a minimum spanning tree on this graph. Let M_T denote the cost of the minimum spanning tree. Then $M_T/2$ is a lower bound on $C(T^*)$. To see that, consider the optimal phylogenetic alignment T^* on T. A depth-first traversal of T^* specifies a way to connect the leaves which has cost less than $2C(T^*)$. By definition of MST, this spanning connection of the leaves has cost no less than M_T, so $C(T^*) > M_T/2$.

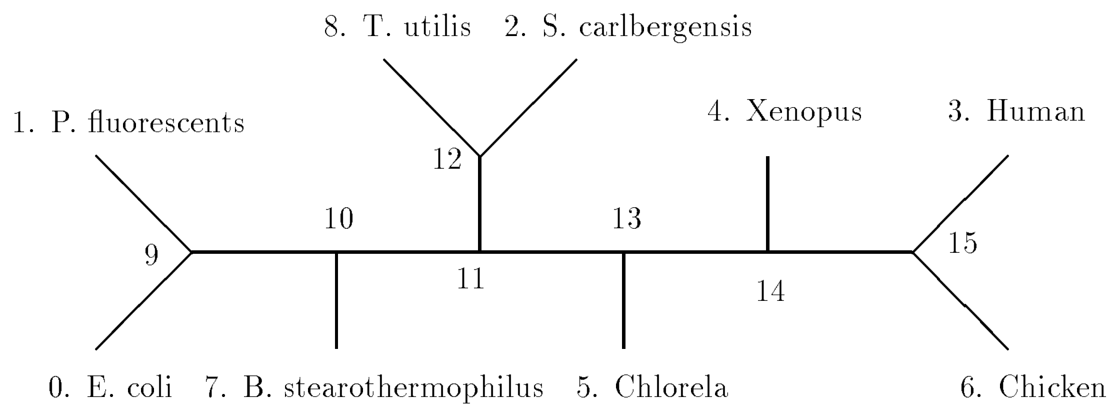

Figure 6: The tree T used in [6].

However, the same argument holds for any phylogenetic alignment, optimal or not. Therefore if T^L is any uniform lifted alignment, then $C(T^L) > M_T/2$, so the average cost of the uniform lifted alignments is always a better (higher) lower bound than $M_T/2$. In the experiment we ran, the cost of the minimum spanning tree for the nine sequences is 366. Hence it gives a lower bound of 183 compared to the (higher) lower bound of 244.9 obtained from Theorem 3.1. The MST lower bound establishes only that Sankoff's alignment deviates from the optimal alignment by at most 62%.

The following linear programming lower bound was suggested by R. Ravi. For each edge e in T, create a variable associated with e. Then for each leaf pair (i, j), create the constraint that the variables associated with edges on the path from i to j must sum to at least $D(i, j)$. Subject to those constraints, the minimum sum of all the variables is a lower bound on $C(T^*)$. This follows from the assumed triangle inequality condition, and the fact that the edge distances from any phylogenetic alignment provide a feasible solution to this LP.

We ran this LP on the example from [6], and obtained a value of 253.5. This establishes that Sankoff's alignment deviates from the optimal by at most 15.78%. We have also been able to prove that the lower bound using Theorem 3.1 is never greater than the LP bound. However, under certain conditions, the bounds are equal, and we don't know how far apart the two bounds will typically be.

3.5 Discussion

The idea of using guaranteed error bounds to compute lower bounds on the cost of an optimal solution has been met with strong disbelief since it was first proposed in an earlier draft of this paper.

One objection is the claim that bounded-error approximation methods rarely perform as badly as their guaranteed error bounds allow, hence dividing the cost of the solution by the guarantee will give a poor lower bound. Without arguing how well bounded approximation methods do generally, our response is twofold. First, poor compared to what else? Alternative lower bounds are not always available or easy to compute. In the case of phylogenetic alignment, the bound based on Theorem 3.3 is easier to compute than the MST bound and was substantially better, while reasonably close to the LP bound, which was much harder to compute. In fact, in the experiment, we initially, and trivially, computed the bound based on Theorem 3.3 by hand. It was much harder to set up, and input the LP, and then check its result. Second, in the case of the phylogenetic alignment problem, we note that the best (highest) lower bound we use is not half of the best (lowest) upper bound obtainable by lifted alignment methods. The best upper bound (364.0 in our experiment) comes from the optimal lifted alignment, which has cost no greater (and generally less) than the optimal *uniform* lifted alignment (cost 371.5), which has cost no greater (and generally less) than the *average uniform* lifted alignment (cost 461.6), which has cost less than the actual *number* (489.8) DG [should be 485.72] we compute (using Theorem 3.1) and divide by two, in order to get the lower bound (244.9). Generally, we expect a sizable gap between the cost of the optimal lifted alignment and that number. Moreover, a different lower bound can be calculated [using Theorem 3.1, and]for every layout of the tree, and there are an exponential number of layouts. Additional variation comes with the choice of root in cases when the tree is unrooted. The final lower bound is the highest one obtained by the various trials. Hence even if the optimal lifted alignment (the bounded-error method with the lowest upper bound) has a cost close to $C(T^*)$, it does not follow that the bounds suggested in this paper must be close to $C(T^*)/2$.

A related objection is that the empirical lower bounds reported here are large compared to empirical bounds for other problems, obtained by effective combinatorial optimization methods. The claim is that a good solution should be within 3% of the optimal, not 20% of the optimal. We agree, but note that efficient methods that reliably obtain provable bounds of 3% were not generally obtained in the first papers proposing those methods. They were obtained after much work by a community of researchers. Lower bounds based on lifted alignments can be improved with additional ideas (we know some now), but probably will require additional computation time. Moreover, bounds in the range of 3% come from improving both *upper* and the lower bound methods. In our experiment, we only compute a lower bound for an existing solution that provides the upper bound. We do not know if the numbers (16% or 20%) are large because the lower bound methods are poor, or because the phylogenetic alignment we examined is far from optimal. So judging the empirical results in this paper by empirical results from much more mature and expensive methods (both upper and lower bounds) is too severe. A more meaningful comparison is to other lower bound methods with comparable speed and simplicity. It is still unresolved how much better the LP method does in general than the methods based on theorems 3.3 and 3.1. But considering the difference in computational effort, we believe the performance of our methods is encouraging enough to continue research on this idea, both for the phylogenetic alignment problem, and for other optimization problems.

4 Acknowledgement

We would like to thank the anonymous referee for suggestions on improving the paper, and on pushing us to improve the initial empirical results.

References

[1] D. Gusfield. Efficient methods for multiple sequence alignment with guaranteed error bounds. *Bull. of Math. Biology*, 55:141–154, 1993.

[2] D. Gusfield. *Algorithms on Strings, Trees and Sequences: Computer Science and Computational Biology.* Cambridge University Press, 1997.

[3] R. Ravi and J. D. Kececioglu. Approximation methods for sequence alignment under a fixed evolutionary tree. *Proc. 6'th Symp. on Combinatorial Pattern Matching. Springer LNCS 937*, pages 330–339, 1995.

[4] D. Sankoff. Minimal mutation trees of sequences. *SIAM J. on Applied Math.*, 28:35–42, 1975.

[5] D. Sankoff and R. Cedergren. Simultaneous comparisons of three or more sequences related by a tree. In D. Sankoff and J. Kruskal, editors, *Time warps, string edits, and macromolecules: The theory and practice of sequence comparion*, pages 253–264. Addison Wesley, 1983.

[6] D. Sankoff, R. J. Cedergren, and G. Lapalme. Frequency of insertion-deletion, transversion, and transition in the evolution of 5s ribosomal RNA. *J. Mol. Evol.*, 7:133–149, 1976.

[7] David Sankoff. Minimal mutation trees of sequences. *SIAM J. on Applied Math.*, 28:35–42, 1975.

[8] L. Wang and D. Gusfield. Improved approximation algorithms for tree alignment. *Proc. 7'th Symp. on Combinatorial Pattern Matching. Springer LNCS 1075,*, pages 220–233, 1996.

[9] L. Wang and T. Jiang. On the complexity of multiple sequence alignment. *J. of Comp. Biology*, 1:337–348, 1994.

[10] L. Wang, T. Jiang, and D. Gusfield. A more efficient approximation scheme for tree alignment. In *Proc. of RECOMB 97: The first international conference on computational molecular biology*, pages 310–319. ACM Press, 1997.

[11] L. Wang, T. Jiang, and E.L. Lawler. Approximation algorithms for tree alignment with a given phylogeny. *Algorithmica*, 16:302–315, 1996.