

## Multiple String Comparison – The Holy Grail

In this chapter we begin the discussion of *multiple string comparison*, one of the most important methodological issues and most active research areas in current biological sequence analysis. We first discuss some of the reasons for the importance of multiple string comparison in molecular biology. Then we will examine multiple string *alignment*, one common way that multiple string comparison has been formalized. We will precisely define three variants of the multiple alignment problem and consider in depth algorithms for attacking those problems. Other variants will be sketched in this chapter, and additional multiple alignment issues are discussed in Part IV.

# 1 Why multiple string comparison?

For a computer scientist, the multiple string comparison problem may at first seem like a generalization for generalization’s sake – “two strings good, four strings better.” But in the context of molecular biology, multiple string comparison (of DNA, RNA or protein strings) is much more than a technical exercise. It is the most critical cutting-edge tool for *extracting and representing* biologically important, yet faint or widely dispersed, commonalities from a *set* of strings. These (faint) commonalities may reveal evolutionary history, or critical conserved motifs or conserved characters in DNA or protein, or common two and three dimensional molecular structure, or clues about the common biological function of the strings. Such commonalities are also used to characterize families or super-families of proteins. These characterizations are then used in database searches to identify other potential members of a family. Because many important commonalities are faint or widely dispersed, they might not be apparent when comparing two strings alone, but may become clear, or even obvious, when comparing a set of related strings. As an example, see Figure 1.

One central technique for multiple string comparison involves *multiple alignment*. While the main algorithmic issues involved in multiple alignment will be discussed later, we need here a clear definition in order to introduce some of its uses.

**Definition** A global multiple alignment of  $k > 2$  strings  $\mathcal{S} = \{S_1, S_2, \dots, S_k\}$  is a natural generalization of alignment for two strings. Chosen spaces are inserted into (or at either end of) each of the  $k$  strings so that the resulting strings have the same length, defined to be  $l$ . Then the strings are arrayed in  $k$  rows of  $l$  columns each, so that each character and space of each string is in a unique column.

For example, see Figure 1 and ??.

## Multiple comparison can be inverse to two-string comparison

In one sense, multiple string comparison is a natural extension of two-string comparison (via alignment), when one is looking for *weakly* conserved patterns rather than

```

*vvvvv*
HUMA      VLSPADKTNVKAAGKVGGAHAGEYGAEALERMFLSFPTTKTYFPHF DLSH      GS
HAOR      MLTDAEKKEVTALWGKAAGHGEEYGAEALERLFQAFPPTTKTYFSHF DLSH      GS
HADK      VLSAADKTNVKGVFSKI GGHAEEYGAEGLERMFIAYPQTKTYFPHF DLSH      GS
HBHU      VHLTPEEKSAVTALWGKV NVDEVGGEALGRLLVVYPWTQRFESFGDLSTPDAVMGN
HBOR      VHLSGGEKSAVTNLWGKV NINELGGEALGRLLVVYPWTQRFEEAFGDLSSAGAVMGN
HBDK      VHWTAEKQLITGLWGKV NVADCGAEALARLLIVYPWTQRFASFGNLSSPTAILGN
MYHU      GLSDGEWQLVLNVWGKVEADIPGHGQEV LIRLFKHPETLEKFDKFKHLKSEDEMKAS
MYOR      GLSDGEWQLVLKVGKVEGDLPGHGQEV LIRLFKTHPETLEKFDKFKGLKTEDEMKAS
IGLOB     SPLTAEASLVQSSWK AVSHNEVEILAAVFAAYPDIQNKFSQFA1GKDLASIKDT
GPUGNI    ALTEKQEALLKQSWEVLKQNI PAHSLRFLALIEAAPESKYVFSFLKDSNEIPE NN
GPYL      GVLTDVQVALVKSSFEEFNANIPKNTHRFFTLVLEIAPGAKDLFSFLKGSSEVPQ NN
GGZLB     MLDQQTINI I KATVPVLKEHGV TITTTFYKNLFAKHPEVRPLF DMGRQE SL

vvvvv          vvvv*
HUMA      AQVKGHGKKVADAL TNAV      AHVDDM      PNALSALSDLHAHKLRVDPVNFKLLS
HAOR      AQIKAHGKKVADAL STAA      GHFDDM      DSALSALSDLHAHKLRVDPVNFKLLA
HADK      AQIKAHGKKVAAAL VEA      NHVDDI      AGALSKLSDLHAQKLRVDPVNFKFLG
HBHU      PKVKAHGKKVLGAFSDGL      AHLDNL      KGT FATLSELHCDKHLHVDPENFRLLG
HBOR      PKVKAHGAKVLT SFGDAL      KNLDDL      KGT FAKLSELHCDKHLHVDPENFNRLG
HBDK      PMVRAHGKKVLT SFGDAV      KNLDNI      KNTFAQLSELHCDKHLHVDPENFRLLG
MYHU      EDLKKHGATVLTALGGIL      KKKGHH      EAEIKPLAQSHATKHKIPVKYLEFIS
MYOR      ADLKKHGGTVLTALGNIL      KKKGQH      EAELKPLAQSHATKHKISIKFLEYIS
IGLOB     GAFATHATRIVSFLSEVIAL1SGNTSNAAAV NSLVSKLGDDHKARGVSAAQ1FGEFR
GPUGNI    PKLKAHAAVIFK TICESA      TELRQKGHAVWDNNTLKR LGSIH LKNKITDPHFVEMK
GPYL      PDLQAHAGKVFKLTYEAA      IQLEVNGAVASDATL KSLGSHVSVKGVVDA HFPVVK
GGZLB     EQPKALAMTVLAAAQNI      ENLPAI      LPAVKKIAVKHC QAGVAAAHYPIVG

vvvvv          vvv
HUMA      HCLLVTLAAHLPAEFTPAVHASL DFKFLASVSTVLT SKYR
HAOR      HCILVVLARHCPGEFTPSAHAAMD KFLSKVATVLT SKYR
HADK      HCFLVVVAIHHPAALTPEVHASL DFKFMCAVGAVLT AKYR
HBHU      NVLVCVLAHHFGKEFTPPVQAAYQKV VAGVANALAHKYH
HBOR      NVLIVVLARHFSKDFSP EVQA AWQKLVSGVAHALGHKYH
HBDK      DILIIVLA AHFTKDF TPECQA AWQKLV RVVAHALARKYH
MYHU      ECIIQVLQSKHPGDFGADAQ GAMNKALELFRKDMASNYKELGFQG
MYOR      EAIHVLQSKHSADFGADAQA AMGKALELFRNDMAAKYKEFGFQG
IGLOB     TALVAYLQANVS WGDNVAAAWNKAL1DNTFAI VVPR L
GPUGNI    GALLGTIKEAIKENWSDEM GQAWTEAYNQLVATIKAEMKE
GPYL      EAILKTIKEVVGDKWSEELNTAWTIAYDELAI I IKKEMKDA A
GGZLB     QELLGAIKEVLGDAATDDILD AWGKAYGVIADVFIQVEADLYAQAVE

```

Figure 1: A multiple alignment of several amino acid sequences of globin proteins from the paper of McLure, Vasi and Fitch [14]. The abbreviations on the left indicate the organisms that the globin sequences are from. Because of the length of the sequences, the multiple alignment is displayed using three lines for each sequence. Columns in the alignment containing a high concentration of similar residues in regions of known

*strongly* conserved patterns. Quoting from Arthur Lesk [12]: “One or two homologous sequences whisper ... a full multiple alignment shouts out loud”.

But in another sense, multiple string comparison is used in a fundamentally different way than two-string comparison. It can be used for biological inference problems that are *inverse* to problems addressed by two-string comparison. In the context of database searching, two-string alignment finds strings that have common subpatterns (substrings or subsequences), but may not have been known to be biologically related. Indeed, the greatest value of database searching comes from using *apparent* string similarities to identify *unsuspected* biological relationships. The inverse problem is to move from *known* biological relationships to *unknown* substring or subsequence similarities. This direction uses multiple string comparison to deduce unknown conserved subpatterns from a set of strings which are already known to be biologically related.

## 1.1 Biological basis for multiple string comparison

Recall the *first fact of biological sequence comparison* from Section ?? on page ?. This first fact underlies the effectiveness of two sequence comparison and of biological database searching. But as powerful as the *first fact* is, it does not capture all the important biology forming the basis for biological sequence comparison. There is an additional fact that compels *multiple* string comparison.

### The Second Fact of biological sequence comparison

Evolutionarily and functionally related molecular strings can *differ significantly* throughout much of the string and yet preserve the same three dimensional structure(s), or the same two dimensional substructure(s) (motifs, domains), or the same active sites, or the same or related dispersed residues (DNA or amino acid).

This *second fact* was implicit in the earlier discussion of *local* alignment (Section ??). What is added here is a matter of degree. Two strings specifying the “same” protein in different species may be so different that the few observed similarities may just be due to chance. Typically, secondary or three-dimensional structure is the most well preserved feature of a large set of evolutionarily related proteins. Conservation of function is less common, and conservation of amino-acid sequence is the least common. “One of the most intriguing observations to arise from recent analyses of protein structures is that similar folds recur, even in the absence of sequence similarity [21]”. (This point was also made by the quote from Cohen on page ??.)

For example, hemoglobin is a nearly universal protein containing four chains of about 140 amino acids each. It is contained in organisms as diverse as mammals and insects, and functions in essentially the same way (it binds and transports oxygen). However, in the time since insects and invertebrates diverged (some 600 million

years ago), multiple amino acid substitutions have occurred in insect and invertebrate hemoglobin sequences, so that on average there have been about 100 amino acid mutations in each chain of the two sequences<sup>1</sup>[1]. A pairwise alignment of hemoglobin chains from two mammals will suggest a functional similarity of the protein (human and chimpanzee sequences are in fact identical), but a two-string alignment from a mammal and an insect may reveal very little. Moreover, many proteins mutate faster than hemoglobin, compounding the problem.

As another, more recent example, the three-dimensional structure of part of a *cell adhesion molecule* called *E-Cadherin* was worked out [16, 22]. The deduced structure was unexpectedly found to show “remarkable similarity to the Ig fold (although there is no sequence homology)”[22]. The Ig fold is a common structure first seen in the immunoglobulin (antibody) superfamily, but later seen in many other proteins as well. The term “homology” here should be interpreted as “similarity”, and “no homology” should be interpreted as “no more similarity than expected between random strings”. So the fold found is a conserved physical structure relating the Ig proteins with the E-Cadherin protein, despite the lack of sequence similarity. A related discussion in [22] about a different protein (the growth hormone receptor) illustrates the same point: “Previously, the growth hormone receptor structure was found to be built of domains related to this (Ig) evolutionarily successful motif; in this case too, no sequence homology exists.”

The ability of many proteins to preserve structure and function in the face of massive amino acid substitution is simply an empirical fact, although it may seem counter-intuitive. One might intuitively expect<sup>2</sup> that every detail of an amino acid string or its encoding DNA is essential, so that even small changes to the string would destroy the structure or function of the protein. And indeed, many gene-related diseases are caused by such small mutations. Often even a single change at a *particular* amino acid position, which might be caused by a single DNA nucleotide change, leads to a serious disease such as sickle-cell anemia, cancer (the RAS oncogene for example is activated by a single nucleotide point mutation [20, 5]), or a prion disease such as *Mad Cow Disease* [17]. But on whole, *many* amino acid positions in a protein string are *non-critical*, and over evolutionary history have mutated to a wide selection of other amino acids without a destructive effect on the protein. So, although the *first fact* of biological sequence analysis, that sequence similarity implies structural or functional similarity, is very powerful and widely exploited, the *converse* of that fact simply is not true.

The extent of permissive mutations in structurally or functionally conserved molecules may be such that comparing two strings at a time reveals little of the critically con-

---

<sup>1</sup>This doesn't translate immediately into the number of positions where amino acid differences occur between mammal and insect hemoglobin because more than one mutation can occur at a given position.

<sup>2</sup>And some early papers on protein evolution even state this expectation as a “fact”. For example see [6].

served patterns or of the critical amino acids. There may be so few conserved amino acids or they may be so widely dispersed that the best alignment between two related protein strings is statistically indistinguishable from the best alignment of two random amino acid strings. The problem occurs at the other extreme as well. Comparison of two proteins from highly related species might not reveal biologically important conserved patterns because the critical similarities are lost in the large number of similarities due only to recent shared evolution. So when doing two-string comparison to find critical common patterns, the challenge is to pick species whose level of divergence is “most informative” (a vague and difficult task).

Multiple string comparison (often via multiple alignment) is a natural response to the problems introduced above. With multiple string comparison, it is not as crucial (although it is still helpful) to pick species whose level of divergence is “most informative”. Often, biologically important patterns that cannot be revealed by comparison of two strings alone become clear when many related strings are simultaneously compared. Moreover, with multiple alignments, it is sometimes possible to arrange a set of related strings (often in a tree) to demonstrate *continuous* changes along a path connecting two extreme strings which by themselves show little pairwise similarity. In particular, multiple alignments of this kind are very useful in deducing evolutionary history.

## 2 Introduction to computing multiple string alignments

Having motivated multiple string comparison with the preceding discussions, we now move to the purely technical issues of how to compute *multiple string alignment*. The definition of *global* multiple alignment was given on page 1, and examples are shown in Figures ?? and 3 b). It is also natural to define *local* multiple alignment. Two-string *local* alignment was defined as a global alignment of *substrings*, and multiple local alignment can be similarly defined.

**Definition** Given a set of  $k > 2$  strings  $\mathcal{S} = \{S_1, S_2, \dots, S_k\}$ , a *local* multiple alignment of  $\mathcal{S}$  is obtained by selecting one substring  $S'_i$  from each string  $S_i \in \mathcal{S}$  and then globally aligning those substrings.

All of the biological justifications for preferring local alignment to global alignment of two strings (see Section ??) also apply to local versus global multiple alignment. These justifications are further strengthened by the *Second Fact* of biological sequence analysis, discussed above. However, the best theoretical results (from a computer science perspective) on multiple alignment have been obtained for global alignment, and so our emphasis will be on global alignment. We will briefly discuss local multiple alignment in Section ??.

## 2.1 How to score multiple alignments

While the notion of a multiple alignment is easily extended from two strings to many strings, the *score* or goodness of a multiple alignment is not as easily generalized. To date, there is no objective function that has been as well accepted for multiple alignment as (weighted) edit distance or similarity has been for two-string alignment. In fact, as we will see later, some popular methods to find multiple alignments do so without using any explicit objective function. The goodness of those methods is judged by the biological meaning of the alignments that they produce, and so the biological insight of the evaluator is of critical importance.

Not being biologists, we will emphasize multiple alignment methods that *are* guided by explicit objective functions, although we will sketch other methods as well. We will discuss three types of objective functions in this chapter, *sum-of-pairs functions*, *consensus functions* and *tree functions*. Although these objective functions are quite different, we will explore approximation algorithms for these problems that are highly related. In Part IV of the book we will discuss how multiple alignment problems interact with problems of deducing trees representing evolutionary history.

**Definition** Given a multiple alignment  $\mathcal{M}$ , the *induced pairwise* alignment of two strings  $S_i$  and  $S_j$  is obtained from  $\mathcal{M}$  by removing all rows except the two rows for  $S_i$  and  $S_j$ . That is, the induced alignment is the multiple alignment  $\mathcal{M}$  restricted to  $S_i$  and  $S_j$ . Any two opposing spaces in that induced alignment can be removed if desired.

**Definition** The *score* of an induced pairwise alignment is determined using any chosen scoring scheme for two-string alignment in the standard manner.

For an example, see Figure 2.

Notice that the definition of “score” does not specify whether a score is a weighted *distance* or a *similarity*. As before, similarity is more natural for treating local alignment. Most of this chapter concerns global alignment, where the algorithms and consequent theorems all require the score to be a *weighted distance*.

## 3 Multiple alignment with the sum-of-pairs (SP) objective function

### The sum-of-pairs (SP) score

**Definition** The *sum of pairs (SP)* score of a multiple alignment  $\mathcal{M}$  is the *sum* of the scores of pairwise global alignments *induced* by  $\mathcal{M}$ . See Figure 2.

Although one can give “handwaving” arguments for the significance of the *SP* score, it is difficult to give a theoretical justification for it (or any other multiple alignment scoring scheme). However, the *SP* score is easy to work with and has been used by many people studying multiple alignment. The *SP* score was first introduced

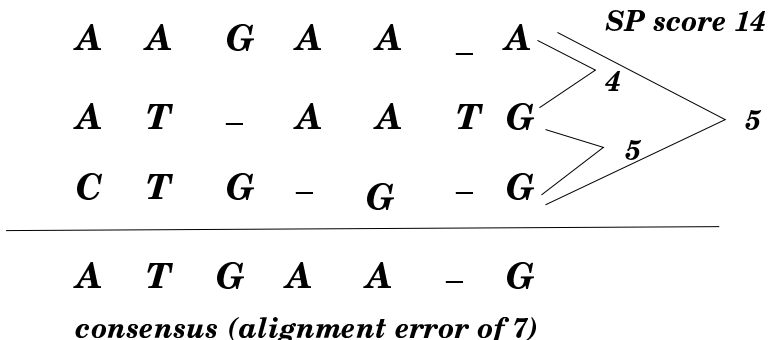


Figure 2: Multiple alignment  $\mathcal{M}$  of three strings shown above the horizontal line. Using the pairwise scoring scheme of  $ms + id$ , i.e.,  $\#(\text{mismatches}) + \#(\text{spaces opposite a non-space})$ , the three induced pairwise alignments have scores of 4, 5, 5 for a total  $SP$  score of 14. Note that a space opposite a space contributes a zero. Using the plurality rule, the consensus string  $S_{\mathcal{M}}$  (defined in Section 4.2) for alignment  $\mathcal{M}$  is shown below the horizontal line. It has an alignment error of seven.

in [4], and is also used in [15, 3, 8]. A similar score is used in [7]. The  $SP$  score is also used in a subtask in the multiple alignment package MACAW [19] developed at the National Institutes of Health, National Center for Biotechnology Information.

**The  $SP$  alignment problem:** Compute a global multiple alignment  $\mathcal{M}$  with *minimum* sum-of-pairs score.

### 3.1 An exact solution to the $SP$ alignment problem

As might be expected, the  $SP$  problem can be solved exactly (optimally) via dynamic programming [15]. Unfortunately, if there are  $k$  strings and each is of length  $n$  say, dynamic programming takes  $\Theta(n^k)$  time and hence is practical for only a small number of strings. Moreover, the exact  $SP$  alignment problem has been proven to be NP-complete [23]. So we will develop the dynamic programming recurrences only for the case of three strings. Extension to any larger number of strings is straightforward, although impractical for even five strings whose lengths are in the low hundreds (typical of proteins). We will also develop an accelerant to the basic dynamic programming solution, somewhat increasing the number of strings that can be optimally aligned.

**Definition** Let  $S_1, S_2, S_3$  denote three strings of lengths  $n_1, n_2$  and  $n_3$  respectively, and let  $D(i, j, k)$  be the optimal  $SP$  score for aligning  $S_1[1..i], S_2[1..j]$  and  $S_3[1..k]$ . The score for a match, mismatch or space is specified by the variables *smatch*, *smis*, *sspace* respectively.

The dynamic programming table  $D$  used to align three strings forms a three dimensional cube. Each cell  $(i, j, k)$  that is not on a boundary of the table (i.e. that has no index equal to zero) has seven neighbors that must be consulted to determine

$D(i, j, k)$ . The general recurrences for computing the cost of a non-boundary cell are similar in spirit to the recurrences for two strings, but are a bit more involved. The recurrences are encoded in the following pseudocode.

### Recurrences for a non-boundary cell $(i, j)$

```

for  $i := 1$  to  $n_1$  do
  for  $j := 1$  to  $n_2$  do
    for  $k := 1$  to  $n_3$  do
      begin
        if  $(S_1(i) = S_2(j))$  then  $c_{ij} := \text{smatch}$ 
        else  $c_{ij} := \text{smis}$ ;
        if  $(S_1(i) = S_3(k))$  then  $c_{ik} := \text{smatch}$ 
        else  $c_{ik} := \text{smis}$ ;
        if  $(S_2(j) = S_3(k))$  then  $c_{jk} := \text{smatch}$ 
        else  $c_{jk} := \text{smis}$ ;

         $d1 := D(i-1, j-1, k-1) + c_{ij} + c_{ik} + c_{jk}$ ;
         $d2 := D(i-1, j-1, k) + c_{ij} + 2 * \text{sspace}$ ;
         $d3 := D(i-1, j, k-1) + c_{ik} + 2 * \text{sspace}$ ;
         $d4 := D(i, j-1, k-1) + c_{jk} + 2 * \text{sspace}$ ;
         $d5 := D(i-1, j, k) + 2 * \text{sspace}$ ;
         $d6 := D(i, j-1, k) + 2 * \text{sspace}$ ;
         $d7 := D(i, j, k-1) + 2 * \text{sspace}$ ;

         $D(i, j, k) := \text{Min}[d1, d2, d3, d4, d5, d6, d7]$ ;
      end;

```

What remains is to specify how to compute the  $D$  values for the boundary cells on the three initial faces of the table, i.e., when  $i = 0$ , or  $j = 0$ , or  $k = 0$ . To do this, let  $D_{1,2}(i, j)$  denote the familiar *pairwise* distance between substrings  $S_1[1..i]$  and  $S_2[1..j]$ ; and let  $D_{1,3}(i, k)$  and  $D_{2,3}(j, k)$  denote the analogous pairwise distances involving pairs  $S_1, S_3$  and  $S_2, S_3$ . These distances are computed in the standard way. Then,

$$\begin{aligned}
D(i, j, 0) &= D_{1,2}(i, j) + (i + j) * \text{sspace}, \\
D(i, 0, k) &= D_{1,3}(i, k) + (i + k) * \text{sspace}, \\
D(0, j, k) &= D_{2,3}(j, k) + (j + k) * \text{sspace}, \\
\text{and } D(0, 0, 0) &= 0.
\end{aligned}$$

The correctness of the recurrences and the fact that they can be evaluated in  $O(n_1 n_2 n_3)$  time is left to the reader. The recurrences can be generalized to incorporate alphabet-weighted scores, and this is also left to the reader.

## 3.2 A bounded-error approximation method for $SP$ alignment

Because the exact solution to the  $SP$  alignment problem is feasible for only a small number of strings, most practical (heuristic) multiple alignment methods do not insist on finding the optimal  $SP$  alignment. However, little is usually known about how much the alignments produced by those heuristics can deviate from the optimal  $SP$  alignment. In contrast, in this section we discuss one of the few methods where such error analysis has been possible. We develop here a *bounded-error approximation* method, from [10], for  $SP$  alignment. That is, the method is *provably fast* (runs in polynomial worst-case time) and yet produces alignments whose  $SP$  score is *guaranteed* to be less than *twice* the score of the optimal  $SP$  alignment. This will be the first of several bounded-error approximation methods presented in this book, including additional methods for different multiple alignment problems.

### 3.2.1 An initial key idea: Alignments consistent with a tree

The  $SP$  approximation we present, the improvements of it, and several other methods for other problems all use a key idea that relates multiple alignments to trees<sup>3</sup>. We develop that idea in general before continuing with  $SP$  alignment in particular. Recall that for two strings,  $D(S_i, S_j)$  is the (optimal) weighted edit distance between  $S_i$  and  $S_j$ .

**Definition** Let  $\mathcal{S}$  be a set of strings, and let  $T$  be a tree where each node is labeled with a distinct string from  $\mathcal{S}$ . Then, a multiple alignment  $\mathcal{M}$  of  $\mathcal{S}$  is called *consistent* with  $T$  if the induced pairwise alignment of  $S_i$  and  $S_j$  has score  $D(S_i, S_j)$ , for each pair of strings  $(S_i, S_j)$  that label adjacent nodes in  $T$ . For an example, see Figure 3.

**Theorem 3.1** *For any set of strings  $\mathcal{S}$  and for any tree  $T$  whose nodes are labeled by distinct strings of  $\mathcal{S}$ , we can efficiently find a multiple alignment  $\mathcal{M}(T)$  of  $\mathcal{S}$  that is consistent with  $T$ .*

Note that the role of  $T$  in this theorem is very special and the induced alignment of two strings  $S_i$  and  $S_j$  which do not label adjacent nodes will generally have a score greater than  $D(S_i, S_j)$ .

**Proof of Theorem 3.1** We will construct the multiple alignment  $\mathcal{M}(T)$  one string at a time and show inductively that Theorem 3.1 holds after each new string is added to the alignment. To start, choose any two strings  $S_i$  and  $S_j$  that label adjacent nodes in  $T$  and form a two-string alignment of  $S_i$  and  $S_j$  with distance

---

<sup>3</sup>This connection between trees and multiple alignments should not be confused with the connection between *evolutionary* trees and multiple alignments, to be discussed in Sections ?? and ??.

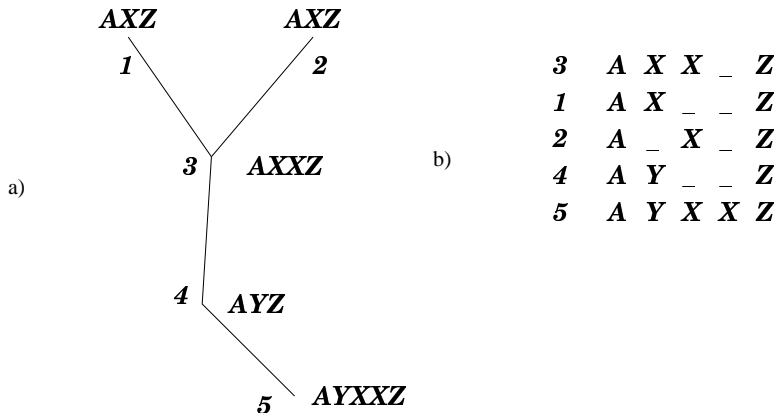


Figure 3: Figure a) shows a tree with its nodes labeled by a (multi-)set of strings. Figure b) is a multiple alignment of those strings that is consistent with the tree. The pairwise scoring scheme scores a zero for each match and a one for each mismatch or space opposite a character. The reader can verify that each of the four induced alignments specified by an edge of the tree has a score equal to its respective optimal distance. However, the induced alignment of two strings which do not label adjacent nodes may have a score greater than their optimal pairwise distance.

$D(S_i, S_j)$ . The theorem trivially holds at this point. Assume that the theorem holds after some arbitrary number of strings have been added to the multiple alignment. To continue, select any string  $S'$  not yet included in the alignment such that  $S'$  labels a node adjacent to a node whose label,  $S_i$  say, is already in the alignment. In that existing multiple alignment, some spaces may have been inserted into  $S_i$ , and we use  $\overline{S}_i$  to denote the string  $S_i$  with those spaces included. Note that for every string in the existing multiple alignment, each character of that string is in a distinct column with exactly one character of  $\overline{S}_i$ .

Next, optimally align string  $S'$  with  $\overline{S}_i$  using a scoring scheme for two-string alignment, *with the added rule* that two opposing spaces have zero cost (they are considered a match). It is immediate that the score of that resulting pairwise alignment is exactly  $D(S_i, S')$ . Now we will add  $S'$  to the existing multiple alignment so that the induced alignment of  $S_i$  and  $S'$  has score  $D(S_i, S')$ , and so that all the induced scores of the strings already in the alignment remain unchanged. Let  $\overline{S}'$  be the string  $S'$  with any spaces inserted by the alignment of  $S'$  and  $\overline{S}_i$ . If the pairwise alignment of  $S'$  and  $\overline{S}_i$  does not insert any new spaces into  $\overline{S}_i$ , then append  $\overline{S}'$  to the existing multiple alignment. The result is a multiple alignment with one more string, where the induced score of  $S_i$  and  $S'$  is  $D(S_i, S')$  and where all the induced scores from the previous multiple alignment remain unchanged.

But if the pairwise alignment does insert a new space in  $\overline{S}_i$  between characters  $l$  and  $l+1$ , say, then insert a space between characters  $l$  and  $l+1$  in *every* string in the *existing* multiple alignment. This will create new column(s) in the existing multiple

alignment in which every character is a space, but otherwise retains the columns of the existing multiple alignment. (For example, suppose that the first four strings from Figure 3 have been multiply aligned and that  $\overline{S_4}$  at that point is  $AY_Z$ . The alignment of  $S_5$  to  $\overline{S_4}$  inserts an additional space into  $\overline{S_4}$  at the fourth position. The first four rows of Figure 3 shows the resulting multiple alignment after that space is replicated in strings  $S_1, S_2$  and  $S_3$ .) The result of adding a column containing only spaces is a multiple alignment in which the score of all pairwise induced alignments is the same as before any spaces were inserted. Then appending  $\overline{S'}$  to that multiple alignment creates a multiple alignment of one more string, where the statement of the theorem holds. The existence of  $\mathcal{M}(T)$  then follows by induction.

The time needed to compute  $\mathcal{M}(T)$  is dominated by the time to compute  $k - 1$  pairwise alignments. If each string has length  $n$ , then each pairwise alignment takes time  $O(n^2)$  and the time to construct  $\mathcal{M}(T)$  is  $O(kn^2)$ .  $\square$

Theorem 3.1 seems to be “folklore” but may have been first explicitly stated in [7]. We can now return to the approximation method for the *SP* alignment problem.

### 3.2.2 The center star method for *SP* alignment

We will describe the method in terms of an *alphabet-weighted* scoring scheme for two-string alignment, and let  $s(x, y)$  be the score contributed when a character  $x$  (possibly a space) is aligned opposite a character  $y$  (possibly a space).

**Definition** A scoring scheme satisfies the *triangle inequality* if for any three characters  $x, y$  and  $z$ ,  $s(x, z) \leq s(x, y) + s(y, z)$ .

Triangle inequality makes good intuitive sense in the context of edit distance. It says that the “cost” of transforming  $x$  to  $z$  directly is no more than the cost of first transforming  $x$  to an intermediate character  $y$ , and then transforming  $y$  to  $z$ . It should be noted however that not all scoring matrices in use in computational biology satisfy the triangle inequality.

**Definition** Given a set of  $k$  strings  $\mathcal{S}$ , define a *center* string  $S_c \in \mathcal{S}$  as a string in  $\mathcal{S}$  that *minimizes*  $\sum_{S_j \in \mathcal{S}} D(S_c, S_j)$ , and let  $M$  denote that minimum sum. Define the *center star* to be a star tree of  $k$  nodes, with the center node labeled  $S_c$  and with each of the  $k - 1$  remaining nodes labeled by a distinct string in  $\mathcal{S} - S_c$ . For an example, see Figure 4.

**Definition** Define the multiple alignment  $\mathcal{M}_c$  of the set of strings  $\mathcal{S}$  to be the multiple alignment *consistent* with the center star.

**Definition** Define  $d(S_i, S_j)$  as the score of the pairwise alignment of strings  $S_i$  and  $S_j$  *induced* by  $\mathcal{M}_c$ . Denote the score of an alignment  $\mathcal{M}$  as  $d(\mathcal{M})$ .

Clearly,  $d(S_i, S_j) \geq D(S_i, S_j)$ , and  $d(\mathcal{M}_c) = \sum_{i < j} d(S_i, S_j)$ . Also, since  $\mathcal{M}_c$  is consistent with the star tree, and string  $S_c$  is at the center of the star,  $d(S_i, S_c) = D(S_i, S_c)$  for each string  $S_i$ . We will show that  $d(\mathcal{M}_c)$  is at most *twice* the score of

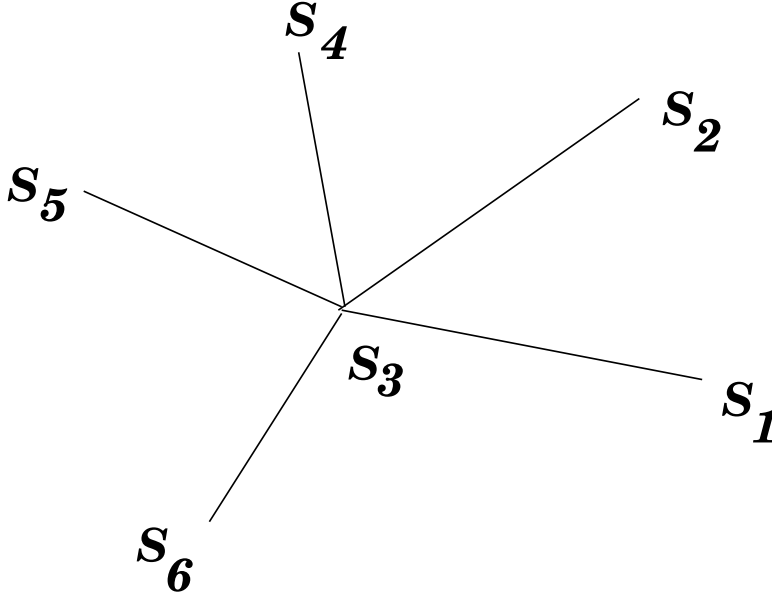


Figure 4: A generic center star for six strings, where the center string  $S_c$  is  $S_3$ .

the optimal  $SP$  multiple alignment of  $\mathcal{S}$ , *provided* that the scoring scheme used for pairwise alignment satisfies the triangle inequality.

**Lemma 3.1** *Assume that the two-string scoring scheme satisfies the triangle inequality. Then, for any strings  $S_i$  and  $S_j$  in  $\mathcal{S}$ ,  $d(S_i, S_j) \leq d(S_i, S_c) + d(S_c, S_j) = D(S_i, S_c) + D(S_c, S_j)$ .*

**Proof** Consider any single column in the multiple alignment  $\mathcal{M}_c$  and let  $x, y$  and  $z$  be the three characters in this column from the three strings  $S_i, S_c, S_j$  respectively. By the triangle inequality,  $s(x, z) \leq s(x, y) + s(y, z)$ , and so the claimed *inequality* follows by the definition of  $d$ . The claimed *equality* follows because the pairwise alignment of  $S_i$  and  $S_c$  induced by  $\mathcal{M}_c$  is an optimal alignment of  $S_i$  and  $S_c$ , and this is true also for the alignment of  $S_c$  and  $S_j$ .  $\square$

**Definition** Let  $\mathcal{M}^*$  be the optimal multiple alignment of the  $k$  strings of  $\mathcal{S}$ . Let  $d^*(S_i, S_j)$  be the score of the pairwise alignment of strings  $S_i$  and  $S_j$  induced by  $\mathcal{M}^*$ . Then  $d(\mathcal{M}^*) = \sum_{i < j} d^*(S_i, S_j)$ .

We can now state and prove the main theorem of this section.

**Theorem 3.2**  $d(\mathcal{M}_c)/d(\mathcal{M}^*) \leq 2(k-1)/k < 2$ .

**Proof** First, define  $v(\mathcal{M}_c) \equiv \sum_{(i,j)} d(S_i, S_j)$  and  $v(\mathcal{M}^*) \equiv \sum_{(i,j)} d^*(S_i, S_j)$ , where the pair  $(i, j)$  is an *ordered* pair in each case. Clearly,  $v(\mathcal{M}_c) = 2d(\mathcal{M}_c)$  and  $v(\mathcal{M}^*) = 2d(\mathcal{M}^*)$ , and so the ratios  $d(\mathcal{M}_c)/d(\mathcal{M}^*)$  and  $v(\mathcal{M}_c)/v(\mathcal{M}^*)$  are equal. It is more convenient to work with the second ratio. Recall that the minimum sum of distances,

$M$ , is defined as  $\sum_j D(S_c, S_j)$ . Now  $v(\mathcal{M}_c) = \sum_{(i,j)} d(S_i, S_j) \leq \sum_{(i,j)} [D(S_i, S_c) + D(S_c, S_j)]$ , by Lemma 3.1. For any fixed  $j$ ,  $D(S_c, S_j)$  (which equals  $D(S_j, S_c)$ ) shows up in this expression exactly  $2(k-1)$  times. So  $v(\mathcal{M}_c) \leq 2(k-1) \times \sum_j D(S_c, S_j) = 2(k-1)M$ .

From the other side,  $v(\mathcal{M}^*) = \sum_{(i,j)} d^*(S_i, S_j) \geq \sum_{(i,j)} D(S_i, S_j) = \sum_i \sum_j D(S_i, S_j) \geq k \times \sum_j D(S_c, S_j) = kM$  (by the choice of  $S_c$ ). So  $d(\mathcal{M}_c)/d(\mathcal{M}^*) = v(\mathcal{M}_c)/v(\mathcal{M}^*) \leq 2(k-1)M/kM = 2(k-1)/k = 2 - 2/k < 2$ .  $\square$

Note that for  $k = 3$  the guaranteed upper bound is  $\frac{4}{3}$ . That is, for three strings, the multiple alignment produced by the center star method will *never* be more than 34% more than the optimal *SP* score. Translated into lower bounds this says that for  $k = 3$ ,  $d(\mathcal{M}^*) \geq .75d(\mathcal{M}_c)$ . For  $k = 4$  the upper bound is only 1.5, and for  $k = 6$  (a problem size considered to be too large for efficient exact solution with strings of length 200) the bound is still only 1.67.

**Corollary 3.1**  $kM \leq \sum_{i < j} D(S_i, S_j) \leq d(\mathcal{M}^*) \leq d(\mathcal{M}_c) \leq [2(k-1)/k] \sum_{i < j} D(S_i, S_j)$ .

In practice one can better measure the goodness of  $\mathcal{M}_c$  by the ratio  $d(\mathcal{M}_c)/\sum_{i < j} D(S_i, S_j)$ . By Corollary 3.1 this ratio is always less than two, but the analysis used there is worst case, so one can expect the ratio to often be considerably less than two. Similarly, one should expect that  $d(\mathcal{M}_c)/d(\mathcal{M}^*)$  will often be considerably less than two, since typically  $\sum_{(i,j)} D(S_i, S_j)$  will be considerably larger than  $kM$ , that  $d(\mathcal{M}^*)$  will not generally be close to  $\sum_{i < j} D(S_i, S_j)$  for any but strings which are very similar, and that  $D(S_i, S_j)$  will be less than  $D(S_i, S_c) + D(S_c, S_j)$  for most typical strings.

Corollary 3.1 is also useful in the MSA speedup discussed in Section ?? for the exact solution to *SP* alignment, since that method requires knowing an efficiently computed upper bound  $z$  on the optimal *SP* alignment score.

### 3.3 Weighted *SP* alignment

A generalization of *SP* alignment was introduced by Altschul and Lipman [2], in which the induced pairwise score of each pair  $(S_i, S_j)$  is multiplied by a weight  $w(i, j)$ . Then the score of a multiple alignment  $\mathcal{M}$  is  $\sum_{i < j} w(i, j)d(S_i, S_j)$ , where  $d(S_i, S_j)$  is the score of the pairwise alignment induced by  $\mathcal{M}$ . The weights change the importance given to different pairs of strings, and are often intended to reflect known evolutionary distance between the organisms from which the strings are obtained. Using weights, then one can try to induce the multiple alignment to more accurately reflect known evolutionary history.

The optimal *weighted SP* alignment can again be computed in exponential time by dynamic programming, but little is known about approximating weighted *SP* alignment. However, the weights are primarily desired as a reflection of evolutionary distance, and there is a different multiple alignment objective function that addresses the same goal of using evolutionary history to influence the resulting multiple alignment. That objective function is contained in the *phylogenetic alignment* problem, for

which there is an approximation method quite related to the center star method. We will discuss those topics in Section ???. But before doing so, we discuss the consensus objective function, and an approximation algorithm even more related to the center star method.

## 4 Multiple alignment with consensus objective functions

Many multiple string alignment or comparison methods used in biology have as their goal to derive a *consensus representation* of the critical common features of a *set* of strings. While there is no universal consensus on how to define “consensus”, we examine three definitions that capture the spirit of most of the methods. We will then show that the three definitions lead to the same string and the same multiple alignment, which can again be approximated by the center star method. For clarity of exposition, we will use some notions of consensus that do not appear in the computational biology literature, and consequently we will define some terminology that is new as well.

### 4.1 Steiner consensus strings

The first notion of a consensus sequence is stated without any explicit connection to multiple alignment. As before, we assume the existence of a two-string scoring scheme that satisfies the triangle inequality and recall that  $D(S_i, S_j)$  denotes the weighted edit distance of strings  $S_i$  and  $S_j$ .

**Definition** Given a set of strings  $\mathcal{S}$ , and given another string  $S$ , the *consensus error* of a string  $S$  relative to  $\mathcal{S}$  is  $E(S) = \sum_{S_i \in \mathcal{S}} D(S, S_i)$ . Note that  $S$  need not be from  $\mathcal{S}$ .

**Definition** Given a set of strings  $\mathcal{S}$ , an optimal *Steiner string*  $S^*$  for  $\mathcal{S}$  is a string that *minimizes* the consensus error  $E(S^*)$  over all possible strings.

Note that  $S^*$  need not be from  $\mathcal{S}$ , and generally will not be. We will usually refer to a “Steiner consensus string” as a “Steiner string”.

The Steiner string  $S^*$  attempts to capture and reflect in a single string, the common characteristics of the set of strings  $\mathcal{S}$ . There is no known efficient method to find  $S^*$ , but there is an approximation method which finds a string  $S$  such that the ratio  $E(S)/E(S^*)$  is never more than two.

**Lemma 4.1** *Let  $\mathcal{S}$  have  $k$  strings, and assume that the two-string scoring scheme satisfies the triangle inequality. Then there exists a string  $\bar{S} \in \mathcal{S}$  such that  $E(\bar{S})/E(S^*) \leq 2 - 2/k < 2$ .*

**Proof** Let  $\bar{S}$  be any string in  $\mathcal{S}$ . By the triangle inequality of the pairwise scoring scheme,  $D(\bar{S}, S_i) \leq D(\bar{S}, S^*) + D(S^*, S_i)$  for any  $S_i$ . So  $E(\bar{S}) = \sum_{S_i \in \mathcal{S}} D(\bar{S}, S_i) \leq \sum_{S_i \neq \bar{S}} (D(\bar{S}, S^*) + D(S^*, S_i)) = (k-2)D(\bar{S}, S^*) + E(S^*)$ .

Now pick  $\bar{S}$  to be a string in  $\mathcal{S}$  which is *closest* to the optimal Steiner string  $S^*$ . That is, choose  $\bar{S}$  so that  $D(\bar{S}, S^*)$  is less than or equal to  $D(S_i, S^*)$ , for any  $S_i \in \mathcal{S}$ . (Of course,  $\bar{S}$  is not known constructively since  $S^*$  is not known, but  $\bar{S}$  does exist.) Then  $E(S^*) = \sum_{S_i \in \mathcal{S}} D(S^*, S_i) \geq kD(\bar{S}, S^*)$ . Therefore  $E(\bar{S})/E(S^*) \leq ((k-2)D(\bar{S}, S^*)/kD(\bar{S}, S^*)) + 1 = (k-2)/k + 1 = 2 - 2/k < 2$ .  $\square$

Recall that the *center string*  $S_c$  is a string in  $\mathcal{S}$  which minimizes  $\sum_{S_i \in \mathcal{S}} (S_c, S_i)$  over all strings in  $\mathcal{S}$ .

**Theorem 4.1** *Assuming that the scoring scheme satisfies the triangle inequality,  $E(S_c)/E(S^*) \leq 2 - 2/k$ .*

**Proof** The theorem follows immediately from Lemma 4.1 and the fact that  $E(S_c) \leq E(\bar{S})$  by definition.  $\square$

So the center string has a consensus error that is at most  $2 - 2/k$  times the error of the optimal Steiner consensus string. This result is a specialization of an approximation method used in the phylogenetic alignment problem, to be discussed in Section ??.

## 4.2 Consensus strings from multiple alignment

Although the definition of the optimal Steiner string makes no mention of multiple alignment, the problem of finding  $S^*$  is equivalent to more traditional consensus problems that *are* defined in terms of multiple alignments. We now make those consensus problems precise.

**Definition** Given a multiple alignment  $\mathcal{M}$  of a set of strings  $\mathcal{S}$ , the *consensus character* of column  $i$  of  $\mathcal{M}$  is the character which minimizes the summed distance to it from all the characters in column  $i$ . Let  $d(i)$  denote that minimum sum in column  $i$ .

Since the alphabet is finite, a consensus character for each column of  $\mathcal{M}$  exists and can be found by enumeration. As one simple special case, if the pairwise scoring scheme scores a match with a zero and a mismatch or a space opposite a character with a one, then the consensus character in column  $i$  is the *plurality* character, i.e., the character occurring the most often in column  $i$ . Note that the plurality character can be a space. See Figure 2 for an example.

**Definition** The *consensus string*  $S_{\mathcal{M}}$  derived from alignment  $\mathcal{M}$  is the concatenation of the consensus characters for each column of  $\mathcal{M}$ .

It is common in the computational biology literature to compute a multiple alignment  $\mathcal{M}$  of a set of strings, and then represent those strings by the consensus string

$S_{\mathcal{M}}$  derived from  $\mathcal{M}$ . It is then natural to use the goodness of string  $S_{\mathcal{M}}$  as a way to evaluate the goodness of the multiple alignment  $\mathcal{M}$ . One way to do this is to define the goodness of  $\mathcal{M}$  as  $\sum_i D(S_{\mathcal{M}}, S_i)$ . That is, the goodness of  $\mathcal{M}$  is reflected by how well  $S_{\mathcal{M}}$  acts as a *Steiner string* for the set  $\mathcal{S}$ . Another (seemingly different) way to evaluate the goodness of multiple alignment  $\mathcal{M}$  through its consensus string  $S_{\mathcal{M}}$  is more closely tied to  $\mathcal{M}$ .

**Definition** Let  $\mathcal{M}$  be a multiple alignment of the set of strings  $\mathcal{S}$ , and let  $S_{\mathcal{M}}$  be its consensus string containing  $q$  characters. Then the *alignment error* of  $S_{\mathcal{M}}$  equals  $\sum_{i=1}^q d(i)$ , and the alignment error of  $\mathcal{M}$  is defined as the alignment error of  $S_{\mathcal{M}}$ .

See Figure 2.

**Definition** The *optimal consensus multiple alignment* is a multiple alignment  $\mathcal{M}$  for input set  $\mathcal{S}$  whose consensus string  $S_{\mathcal{M}}$  has smallest alignment error over all possible multiple alignments of  $\mathcal{S}$ .

So we have three notions of how to define a consensus of a set of strings  $\mathcal{S}$ , and two related notions of how to evaluate a multiple alignment  $\mathcal{M}$ . The first notion of consensus is the *Steiner string*  $S^*$  defined from  $\mathcal{S}$  without any mention of multiple alignment. The second notion is the consensus string  $S_{\mathcal{M}}$  which is derived from a multiple alignment  $\mathcal{M}$ , but whose goodness is evaluated by how well it acts as a Steiner string for  $\mathcal{S}$ . In this case, the goodness of the multiple alignment  $\mathcal{M}$  is defined as the consensus error of its consensus string  $S_{\mathcal{M}}$ . The third notion is the most closely tied to multiple alignment. In that notion, string  $S_{\mathcal{M}}$  is both derived from  $\mathcal{M}$  and evaluated by how well it reflects the *columnwise* properties of  $\mathcal{M}$ . In this case the goodness of  $\mathcal{M}$  is again based on the goodness of  $S_{\mathcal{M}}$ , in particular, on the alignment error of  $S_{\mathcal{M}}$ . Three different notions of consensus, but in fact these three notions are equivalent in that they lead to the same multiple alignment.

**Definition** Given set  $\mathcal{S}$  of  $k$  strings, let  $T$  be the star tree with Steiner string  $S^*$  at the root, and each of the  $k$  strings of  $\mathcal{S}$  at distinct leaves of  $T$ . Then the multiple alignment of  $\mathcal{S} \cup S^*$  *consistent* with  $T$  is said to be consistent with  $S^*$ .

**Theorem 4.2** *Let  $S$  denote the consensus string of the optimal consensus multiple alignment. Then, removal of the spaces from  $S$  creates the optimal Steiner string  $S^*$ . Conversely, removal of the row for  $S^*$  from the multiple alignment consistent with  $S^*$  creates the optimal consensus multiple alignment of  $\mathcal{S}$ .*

**Proof** Let  $\mathcal{S}$  have  $k$  strings, and let  $\mathcal{M}$  be any multiple alignment of  $\mathcal{S}$ . By definition, each character of the consensus string  $S_{\mathcal{M}}$  derived from  $\mathcal{M}$  is associated with a distinct column of  $\mathcal{M}$ , and this association induces a particular pairwise alignment between  $S_{\mathcal{M}}$  and each  $S_i$  in  $\mathcal{S}$ . Clearly, the score of that alignment is at least  $D(S_i, S_{\mathcal{M}})$ . Now the alignment error of  $S_{\mathcal{M}}$  is exactly the sum of the scores of those  $k$  pairwise induced alignments, and so the alignment error of  $S_{\mathcal{M}}$  is at least  $\sum_i D(S_i, S_{\mathcal{M}})$ , which is the consensus error of  $S_{\mathcal{M}}$  for  $\mathcal{S}$ . But by definition,  $S^*$  has the minimum consensus error for  $\mathcal{S}$ , so the alignment error of  $S_{\mathcal{M}}$  is at least the consensus error of  $S^*$ .

Now consider the multiple alignment  $\mathcal{M}^*$  of  $\mathcal{S} \cup S^*$  consistent with  $S^*$ , and for any string  $\alpha$  in  $\mathcal{S} \cup S^*$ , let  $\bar{\alpha}$  denote the string in the row of  $\mathcal{M}^*$  corresponding to  $\alpha$ . By consistency, the score of the induced alignment in  $\mathcal{M}^*$  of  $\bar{S}^*$  and  $\bar{S}_i$  is  $D(S^*, S_i)$  for any  $S_i \in \mathcal{S}$ . Let  $\mathcal{M}'$  be  $\mathcal{M}^*$  after removing the row for  $S^*$ . Then, the alignment error of  $\bar{S}^*$  with  $\mathcal{M}'$  is exactly  $\sum_i D(S^*, S_i)$ , which is the consensus error of  $S^*$  for  $\mathcal{S}$ . Hence, using the conclusion from the first paragraph, the alignment error of any other consensus string  $S_{\mathcal{M}}$  for any other multiple alignment  $\mathcal{M}$  must be at least as large as the alignment error of  $\bar{S}^*$  for  $\mathcal{M}'$ . It follows that  $\mathcal{M}'$  is the optimal consensus multiple alignment for  $\mathcal{S}$ , and that  $\bar{S}^*$  is its consensus string. Then, since  $S^*$  is obtained from  $\bar{S}^*$  by removing the spaces in  $\bar{S}^*$ , the theorem is proved.  $\square$

So the optimal consensus multiple alignment specifies the optimal Steiner string  $S^*$ , and conversely,  $S^*$  can be used to construct the optimal consensus multiple alignment. A Steiner string and the consensus string for the optimal consensus multiple alignment both capture the same commonalities in  $\mathcal{S}$ , and the optimal consensus multiple alignment is a way to explicitly display those commonalities in a columnwise manner.

### 4.3 Approximating the optimal consensus multiple alignment

From the proof of Theorem 4.2 and the statement of Theorem 4.1, it should now be clear how to find a multiple alignment  $\mathcal{M}$  whose *alignment error* is never more than  $2 - 2/k$  times the alignment error of the optimal consensus multiple alignment (assuming the triangle inequality of the two-string scoring scheme). Just find the center string  $S_c$  as before, place it at the center of a  $k$  node star, label each leaf with one of the remaining strings in  $\mathcal{S}$ , and construct the multiple alignment  $\mathcal{M}$  consistent with this tree.

The multiple alignment  $\mathcal{M}$  just described is exactly the alignment  $\mathcal{M}_c$  which was used to approximate the *SP* objective function. So we have established the following, perhaps surprising result.

**Theorem 4.3** *Assuming the triangle inequality, the multiple alignment  $\mathcal{M}_c$  created by the center star method has an *SP* score that is never more than  $2 - 2/k$  times the *SP* score of the optimal *SP* alignment, and it has a (consensus) alignment error that is never more than  $2 - 2/k$  times the alignment error of the optimal consensus multiple alignment.*

## References

- [1] B. Alberts, D. Bray, J. Lewis, M. Raff, K. Roberts, and J. Watson. *Molecular Biology of the Cell (third edition)*. Garland press, New York, NY., 1994.

- [2] S. Altschul and D. Lipman. Trees, stars, and multiple sequence alignment. *SIAM J. on Applied Math*, 49:197–209, 1989.
- [3] D. Bacon and W. Anderson. Multiple sequence alignment. *J. Mol. Biol.*, 191:153–161, 1986.
- [4] H. Carrillo and D. Lipman. The multiple sequence alignment problem in biology. *SIAM J. on Applied Math*, 48:1073–1082, 1988.
- [5] G. Cooper. *Oncogenes*. Jones and Bartlett, Boston, MA., 1990.
- [6] M. O. Dayhoff. Computer analysis of protein evolution. *Scientific American*, pages 87–94, July 1969.
- [7] D. Feng and R. F. Doolittle. Progressive sequence alignment as a prerequisite to correct phylogenetic trees. *J. Mol. Evol.*, 25:351–360, 1987.
- [8] O. Gotoh. Alignment of three biological sequences with an efficient traceback. *J. Theo. Biol.*, 121:327–337, 1986.
- [9] S. Gupta, J. Kececioglu, and A. Schaffer. Making the shortest-paths approach to sum-of-pairs multiple sequence alignment more space efficient in practice. *Proc. 6th Symp. on Combinatorial Pattern Matching. Springer LNCS 937*, pages 128–143, 1995.
- [10] D. Gusfield. Efficient methods for multiple sequence alignment with guaranteed error bounds. *Bull. of Math. Biology*, 55:141–154, 1993.
- [11] P. Horton and E. Lawler. An analysis of the efficiency of the  $a^*$  algorithm for multiple sequence alignment. unpublished note, 1994.
- [12] T. J. P. Hubbard, A. M. Lesk, and A. Tramontano. Gathering in the fold. *Nature Structural Biology*, 4:313–313, April 1996.
- [13] D. Lipman, S. Altshul, and J. Kececioglu. A tool for multiple sequence alignment. *Proc. of the Nat. Academy of Science*, 86:4412–4415, 1989.
- [14] M. McClure, T. Vasi, and W. Fitch. Comparative analysis of multiple protein-sequence alignment methods. *Molecular Biology and Evolution*, 11:571–592, 1994.
- [15] M. Murata, J. Richardson, and J. Sussman. Simultaneous comparison of three protein sequences. *Proc. of the Nat. Academy of Science*, 82:3073–3077, 1985.
- [16] M Overduin, T. Harvey, S. Bagby, K. Tong, P Yau, M. Takeichi, and M. Ikura. Solution structure of the epithelial cadherin domain responsible for selective cell adhesion. *Science*, 267:386–389, 1995.

- [17] S. Prusiner. The prion diseases. *Scientific American*, pages 48–57, January 1995.
- [18] E. Rich and K. Knight. *Artificial Intelligence*. McGraw Hill, New York, 1991.
- [19] G.D. Schuler, S.F. Altschul, and D.J. Lipman. A workbench for multiple alignment construction and analysis. *Proteins: Structure, Function and Genetics*, 9:180–190, 1991.
- [20] C. Shih and R. Weinberg. Isolation of a transforming sequence from a human bladder carcinoma cell line. *Cell*, pages 161–169, 1982.
- [21] M. B. Swindells. Commentary: Finding your fold:. *Protein Engineering*, 7:1–3, 1994.
- [22] G. Wagner. E-cadherin: A distant member of the immunoglobulin superfamily. *Science*, 267:342, 1995.
- [23] L. Wang and T. Jiang. On the complexity of multiple sequence alignment. *J. of Comp. Biology*, 1:337–348, 1994.