### Recall: Multiple Sequence Alignment (MSA)
- Goal is to find a common alignment of several sequences
- Provides more information than pairwise alignment
  - Can match a single protein against entire family
- Useful to...
  - Distinguish evolutionary relationships
  - Discover important parts of a protein family
- **TFAA**--**LSK**
- **ALSA**--**LSD**
- **ALSN**--**LSD**
- **MSSMKDLSG**
- **ELKP**--**LAQ**
- MSA is NP-Complete
  - To find the best alignment we typically measure
    - Sum of pairwise distances within each column
    - Distances are measured using a scoring matrix (e.g., BLOSUM or PAM)
  - Best alignment can be found using Dynamic Programming
    - But requires time $\Theta(nk)$ for $k$ sequences of length $n$
  - MSA using Sum-of-Pairs is known to be NP-complete
    - For an NP-complete problem
      - If a fast (polynomial time) algorithm is ever found then all NP-complete problems have fast algorithms
      - Most researchers believe that no polynomial time algorithm exists
      - Goal becomes: find a reasonable approximation to the exact solution

### Recall: Center Star Method
- Choose one sequence to be the center
- Align each of the other sequences with this center sequence
- Try each sequence as the center to find the one with the least cost $\sum_{i\neq c} d(S_c, S_i)$
- Produces an approximation whose sum-of-pairs cost is $\leq 2$ times optimal
  - $d(.,.)$, the scoring matrix, must satisfy the Triangle Inequality

### Phylogenetic Tree Alignment
- Phylogenetic tree = evolutionary tree
  - If you know the phylogenetic tree for your sequences
    - The cost for an alignment is $\sum_{(i,j) \in E} d(S_i, S_j)$ where $E$ is the set of edges in the tree
  - Note that the Center Star Method is just using a particularly bad phylogenetic tree
    - Unfortunately
      - The multiple alignment is often needed to derive the phylogenetic tree
      - Usually, we don’t know the sequences for the internal nodes
Consensus Representations

- **Goal:** Build a single string that somehow represents an entire set, $S$, of strings
- There are 2 related ideas that are candidates for such a string
  - the **Steiner string**
  - the consensus string of the optimal consensus multiple alignment

- The Steiner string, $S^*$, is the string that minimizes the consensus error
  - The consensus error for string $T$ is $\sum_{i=1}^{m} D(S_i, T)$

- Note that the Steiner string is not necessarily a member of $S$
- Note also that the definition of Steiner string does not depend on a multiple alignment (although a Steiner string induces a multiple alignment)

- The **Steiner string**, $S^*$, is the string that minimizes the consensus error

- The consensus error for string $T$ is $\sum_{i=1}^{m} D(S_i, T)$

- Note that the Steiner string is not necessarily a member of $S$

The Consensus String of a Multiple Alignment

- The consensus string $S_M$ derived from multiple alignment $M$ is the concatenation of the consensus characters for each column of $M$
  - The consensus character for column $i$ is the character that minimizes the summed distance to it from all the characters in column $i$

The Optimal Consensus Multiple Alignment

- The optimal consensus multiple alignment is the alignment that minimizes the sum of the column errors
  - The column error is the sum of distances from the consensus character to each character in that column
  - Unfortunately, we have no way to determine the Steiner string (although we can approximate within a factor of 2 using the center-star string)

How is MSA Actually Done?

- The **Center Star Method**
  - Produces a result with a provable bound
  - But it’s not often used in practice because it doesn’t work as well as other methods

- Technique that is commonly used
  - Iterative pairwise alignment (see below)

- Additional methods
  - Repeated-motif methods
  - Hidden Markov models
  (more on this later in the course)

Iterative Pairwise Alignment

- In simplest form
  - Add strings one at a time to a growing multiple alignment
  - The string chosen is the one closest to some string already in the multiple alignment
  - This is basically
    - a Minimum Spanning Tree (when using edit distance)
    - or a Maximum Spanning Tree (when using similarity scores)

- There are lots of variations on this idea
  - We are using the Minimum Spanning Tree as a way to cluster the strings
  - There are many clustering methods; each one leads to a somewhat different method for multiple alignment
  - For some methods we must compute the distance between a sequence and a set of sequences

Summarizing a Group of Sequences: the Profile

- For a multiple alignment of length $n$, a **profile** is a table of size $|\Sigma \cup \{-\}| \times n$
  - Each entry shows the frequency of a symbol within a column
  - $\Sigma$ is the alphabet (in our case, the 20 amino acids)

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>A</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.00</td>
<td>0.25</td>
<td>0.50</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>-</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.00</td>
<td>0.75</td>
<td>0.00</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>-</th>
<th>B</th>
<th>A</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.25</td>
<td>0.00</td>
<td>0.50</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>C</th>
<th>A</th>
<th>-</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.25</td>
<td>0.50</td>
<td>0.00</td>
</tr>
</tbody>
</table>
Aligning a String to a Profile

- Dynamic Programming can be used just as it is for pairwise comparisons.
- We use a weighted sum of s-values when comparing a letter to a profile-column.
  - s(.,.) is the scoring matrix used for pairwise comparisons.
- Profile to profile comparisons can be done similarly.

Example
- Suppose
  + s(A,A) = 2
  + s(A,B) = s(A,-) = -1
  + s(A,C) = -2
- Then A matched to column 1 scores
  \[ 0.5(2) + 0.25(-1) + 0.25(-2) = 0.25 \]

<table>
<thead>
<tr>
<th></th>
<th>Col 1</th>
<th>Col 2</th>
<th>Col 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.50</td>
<td>0.25</td>
<td>0.50</td>
</tr>
<tr>
<td>B</td>
<td>0.00</td>
<td>0.75</td>
<td>0.00</td>
</tr>
<tr>
<td>C</td>
<td>0.25</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>-</td>
<td>-0.25</td>
<td>0.00</td>
<td>0.50</td>
</tr>
</tbody>
</table>

A Multiple Alignment Package: ClustalW

- Basic outline of algorithm
  - Calculate the C(k,2) \( \text{i.e., choose 2} \) pairwise alignment scores.
  - Use a neighbor-joining algorithm to build a tree based on the distances.
  - Distances are updated using string/string, string/profile, and profile/profile comparisons.
- Actual algorithm includes many ad-hoc rules (e.g., weighting, different scoring matrices, and special gap scores).

http://www.uib.no/aasland/chromo/chromo-tree.gif