Multiple Sequence Alignment (II)

(Lecture for CS498-CXZ Algorithms in Bioinformatics)

Oct. 6, 2005

ChengXiang Zhai

Department of Computer Science

University of Illinois, Urbana-Champaign
Dynamic programming for multi-sequence alignment gives an exact solution, but it’s computationally expensive.

How can we help biologists do multi-sequence alignment?
When finding an exact solution is computationally too expensive, we explore how to find an approximate solution...

So, how do we find a good approximation of the optimal multi-sequence alignment?
Inferring Multiple Alignment from Pairwise Alignments

• From an optimal multiple alignment, we can infer pairwise alignments between all sequences, but they are not necessarily optimal

• It is difficult to infer a "good" multiple alignment from optimal pairwise alignments between all sequences
Combining Optimal Pairwise Alignments into Multiple Alignment

Can combine pairwise alignments into multiple alignment

Can not combine pairwise alignments into multiple alignment
Inferring Pairwise Alignments

3 sequences, 3 comparisons

4 sequences, 6 comparisons

5 sequences, 10 comparisons
Multiple Alignment: Greedy Approach

• Choose most similar pair of strings and combine into a consensus, reducing alignment of \( k \) sequences to an alignment of \( k-1 \) “sequences”. Repeat

• This is a heuristic greedy method

\[
\begin{align*}
&k \left\{ \begin{array}{l}
\quad u_1 = \text{ACGTACGTACGT}\ldots \\
\quad u_2 = \text{TTAATTAATTAA}\ldots \\
\quad u_3 = \text{ACTACTACTACT}\ldots \\
\quad \vdots \\
\quad u_k = \text{CCGGCCGGCCGG}\ldots \\
\end{array} \right. \\
&\quad u_1 = \text{AC-TAC-TAC-T}\ldots \\
&\quad u_2 = \text{TTAATTAATTAA}\ldots \\
&\quad \vdots \\
&\quad u_k = \text{CCGGCCGGCCGG}\ldots \\
&\{ k-1 \}
\end{align*}
\]
Greedy Approach: Example

• Consider these 4 sequences

\[ s_1 \quad \text{GATTCA} \]
\[ s_2 \quad \text{GTCTGA} \]
\[ s_3 \quad \text{GATATT} \]
\[ s_4 \quad \text{GTCAGC} \]
Greedy Approach: Example (cont’d)

• There are $\binom{4}{2} = 6$ possible alignments

<p>| | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$s_2$</td>
<td>GTCCTGA</td>
<td>$s_1$</td>
<td>GATTCA--</td>
<td>$s_4$</td>
<td>G–T–CAGC(score = 0)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$s_4$</td>
<td>GTCAGC (score = 2)</td>
<td>$s_1$</td>
<td>GATTCA--</td>
<td>$s_4$</td>
<td>G–T–CAGC(score = 0)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$s_1$</td>
<td>G–T–TCA</td>
<td>$s_2$</td>
<td>G–TCTGA</td>
<td>$s_3$</td>
<td>GATAT–T (score = -1)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$s_2$</td>
<td>G–TCTGA (score = 1)</td>
<td>$s_3$</td>
<td>GATAT–T (score = -1)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$s_3$</td>
<td>GATAT–T (score = 1)</td>
<td>$s_4$</td>
<td>G–TCAAGC (score = -1)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Greedy Approach: Example (cont’d)

s2 and s4 are closest; combine:

\[
\begin{align*}
  s2 & \quad \text{GTCTGA} \\
  s4 & \quad \text{GTCAGC} \\
\end{align*}
\]

new set becomes:

\[
\begin{align*}
  s1 & \quad \text{GATTCA} \\
  s3 & \quad \text{GATATT} \\
  s2,4 & \quad \text{GTCTGA} \\
\end{align*}
\]

There are many (4) alternative choices for the consensus, let’s assume we randomly choose one.
Greedy Approach: Example (cont’d)

set is:

\[
\begin{align*}
&s1 \quad \text{GATTCA} \\
&s3 \quad \text{GATATT} \\
&s2,4 \quad \text{GTCTGA}
\end{align*}
\]

scores are:

\[
\begin{align*}
&s1 \quad \text{GAT-TCA} \\
&s3 \quad \text{GATAT-T} \quad \text{(score} \quad = \quad 1) \\
&s1 \quad \text{GATTC--A} \\
&s2,4 \quad \text{G-T-CTGA} \quad \text{(score} \quad = \quad 0) \\
&s3 \quad \text{GATATT} \\
&s2,4 \quad \text{G-TCTGA} \quad \text{(score} = \quad -1)
\end{align*}
\]

Take best pair and form another consensus:

\[
s1,3 \quad = \quad \text{GATATT} \quad \text{(arbitrarily break ties)}
\]
new set is:

\[
\begin{align*}
\{s_{1,3} & : \text{GATATT} \\
\{s_{2,4} & : \text{GTCTGA} \}
\end{align*}
\]

scores is:

\[
\begin{align*}
\{s_{1,3} & : \text{GATATT} \\
\{s_{2,4} & : \text{G-TCTGA} \ (\text{score}=-1) \}
\end{align*}
\]

Form consensus:

\[
\{s_{1,3,2,4} = \text{GATCTG} \}
\]

(arbitrarily break ties)
**Progressive Alignment**

- *Progressive alignment* is a variation of greedy algorithms with a somewhat more intelligent strategy for scheduling the merges.
- Progressive alignment works well for close sequences, but deteriorates for distant sequences.
  - Gaps in consensus string are permanent.
  - Simplified representation of the alignments.
- Better solution? Use a profile to represent consensus.

### ATG–CAA

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>T</th>
<th>G</th>
<th>C</th>
<th>A</th>
<th>T</th>
<th>G</th>
<th>C</th>
<th>A</th>
<th>A</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>G</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### AT–CCA–

### ACG–CTG

Hidden Markov Models (HMMs) capture such a pattern...
Feng-Doolittle Progressive Alignment

• Step 1: Compute all possible pairwise alignments
• Step 2: Convert alignment scores to distances
• Step 3: Construct a “guide tree” by clustering
• Step 4: Progressive alignment based on the guide tree (bottom up)

Note that variations are possible at each step!
Feng-Doolittle: Clustering Example

Similarity matrix (from pairwise alignment)

<table>
<thead>
<tr>
<th></th>
<th>$X_1$</th>
<th>$X_2$</th>
<th>$X_3$</th>
<th>$X_4$</th>
<th>$X_5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X_1$</td>
<td>20</td>
<td>15</td>
<td>11</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>$X_2$</td>
<td>16</td>
<td>30</td>
<td>5</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>$X_3$</td>
<td>13</td>
<td>5</td>
<td>25</td>
<td>12</td>
<td>11</td>
</tr>
<tr>
<td>$X_4$</td>
<td>3</td>
<td>4</td>
<td>12</td>
<td>40</td>
<td>9</td>
</tr>
<tr>
<td>$X_5$</td>
<td>4</td>
<td>1</td>
<td>11</td>
<td>9</td>
<td>30</td>
</tr>
</tbody>
</table>

Convert score to distance

$$D = - \log S_{eff} = - \log \frac{S_{obs}}{S_{max} - S_{rand}}$$

$$S_{max} = \frac{S_{max}(x) + S_{max}(y)}{2}$$

Guide tree

Length normalization
Feng-Doolittle: How to generate a multiple alignment?

• At each step, follow the guide tree and consider all possible pairwise alignments of sequences in the two candidate groups (3 cases):
  – Sequence vs. sequence
  – Sequence vs. group (the best matching sequence in the group determines the alignment)
  – group vs. group (the best matching pair of sequences determines the alignment)

• “Once a gap, always a gap”
  – gap is replaced by a neutral symbol X
  – X can be matched with any symbol, including a gap without penalty
Generating a Multi-Sequence Alignment

• Align the two most similar sequences

• Following the guide tree, add in the next sequences, aligning to the existing alignment

• Insert gaps as necessary

Sample output:

<table>
<thead>
<tr>
<th></th>
<th>FOS_RAT</th>
<th>PEEMSVTS-LDLTGGGLPEATTPSEEEAFTLPLNDPEPK-PSLEPVKNISNMELKAEPFD</th>
</tr>
</thead>
<tbody>
<tr>
<td>FOS_MOUSE</td>
<td>PEEMSVAS-LDLTGGGLPEASTPEEEAFTLPLNDPEPK-PSLEPVKISNVELKAEPFD</td>
<td></td>
</tr>
<tr>
<td>FOS_CHICK</td>
<td>SEELAAATAALDLG----APSPAAAEAAFALPLMTEAPPAVPPKEPSG--SGLELKAEPFD</td>
<td></td>
</tr>
<tr>
<td>FOSB_MOUSE</td>
<td>PGPGLAEVRDLPG-----STSAKEDGFGWLLPPPPPPP------------------------LPFQ</td>
<td></td>
</tr>
<tr>
<td>FOSB_HUMAN</td>
<td>PGPGLAEVRDLPG-----SAPAKEDGFSWLLPPPPPPP------------------------LPFQ</td>
<td></td>
</tr>
</tbody>
</table>

Dots and stars show how well-conserved a column is.
Problems with Feng-Doolittle

• All alignments are completely determined by pairwise sequence alignment (restricted search space)

  ➔ Profile alignment

• No backtracking (subalignment is “frozen”)
  – No way to correct an early mistake
  – Non-optimality: Mismatches and gaps at highly conserved region should be penalized more, but we can’t tell where is a highly conserved region early in the process

  ➔ Iterative refinement
Profile Alignment

• Aligning two alignments/profiles
• Treat each alignment as “frozen”
• Alignment them with a possible “column gap”

\[
\sum_i S(m_i) = \sum_i \sum_{k<l\leq N} s(m^k_i, m^l_i) \\
= \sum_i \sum_{k<l\leq n} s(m^k_i, m^l_i) + \sum_i \sum_{n<k<l\leq N} s(m^k_i, m^l_i) + \sum_i \sum_{k\leq n, n<l\leq N} s(m^k_i, m^l_i)
\]

Fixed for any two given alignments  
Only need to optimize this part
Iterative Refinement

• Re-assigning a sequence to a different cluster/profile
• Repeatedly do this for a fixed number of times or until the score converges
• Essentially to enlarge the search space
ClustalW: A Multiple Alignment Tool

- Essentially following Feng-Doolittle
  - Do pairwise alignment (dynamic programming)
  - Do score conversion/normalization (Kimura’s model, not covered)
  - Construct a guide tree (neighbour-joining clustering, will be covered later)
  - Progressively align all sequences using profile alignment
ClustalW Heuristics

• Avoid penalizing minority sequences
  – Sequence weighting
  – Consider “evolution time” (using different sub. Matrices)

• More reasonable gap penalty, e.g.,
  – Depends on the actual residues at or around the positions (e.g., hydrophobic residues give higher gap penalty)
  – Increase the gap penalty if it’s near a well-conserved region (e.g., perfectly aligned column)

• Postpone low-score alignment until more profile information is available.
Heuristic 1: Sequence Weighting

• Motivation: address sample bias

• Idea:
  – Down weighting sequences that are very similar to other sequences
  – Each sequence gets a weight
  – Scoring based on weights

\[ S(m_i) = M(t,v) + M(t,i) + M(l,v) + M(l,i) \]

\[ S'(m_i) = w_1 w_3 M(t,v) + w_1 w_4 M(t,i) + w_2 w_3 M(l,v) + w_2 w_4 M(l,i) \]
Heuristic 2: Sophisticated Gap Weighting

• Initially,
  – GOP: “gap open penalty”
  – GEP: “gap extension penalty”

• Adjusted gap penalty
  – Dependence on the weight matrix
  – Dependence on the similarity of sequences
  – Dependence on lengths of the sequences
  – Dependence on the difference in the lengths of the sequences
  – Position-specific gap penalties
  – Lowered gap penalties at existing gaps
  – Increased gap penalties near existing gaps
  – Reduced gap penalties in hydrophilic stretches
  – Residue-specific penalties
Gap Adjustment Heuristics

• **Weight matrix:**
  – Gap penalties should be comparable with weights

• **Similarity of sequences**
  – GOP should be larger for closely related sequences

• **Sequence length**
  – Long sequences tend to have higher scores

\[
GOP = \{GOP + \log[\min(N,M)]\} \ast (\text{avg residue mismatch score}) \ast (\text{percent identity scaling factor})
\]

\[N, M = \text{sequence lengths}\]

• **Difference in sequence lengths**
  – Avoid too many gaps in the short sequence

\[
GEP = GEP \ast [1.0 + |\log(N/M)|] \quad N > M
\]
• Position-specific gap penalties
  – Lowered gap penalties at existing gaps
    \[ GOP = GOP \times 0.3 \times \left( \frac{\text{no. of sequences without a gap}}{\text{no. of sequences}} \right) \]
  – Increased gap penalties near existing gaps
    \[ GOP = GOP \times \left\{ 2 + \frac{8(\text{distance from gap}) \times 2}{8} \right\} \]
  – Reduced gap penalties in hydrophilic stretches (5 AAs)
    \[ GOP = GOP \times \frac{1}{3} \text{ If no gaps, and one sequence has a hydrophilic stretch} \]
  – Residue-specific penalties (specified in a table)
    \[ GOP = GOP \times \text{avgFactor} \text{ If no gaps and no hydrophilic stretch.} \]
    
    Average over all the residues at the position
Heuristic 3: Delayed Alignment of ‘Divergent Sequences’

• Divergence measure: Average percentage of identity with any other sequence

• Apply a threshold (e.g., 40% identity) to detect divergent sequences (“outliers”)

• Postpone the alignment of divergent sequences until all of the rest have been aligned
What You Should Know

• Basic steps involved in progressive alignment

• Major heuristics used in progressive alignment

• Why a progressive alignment algorithm is not optimal