Multiple Alignment

$$\begin{array}{cccccccc} M & Q & P & I & L & LL & L \\ M & L & R & - & L & L- & - \\ M & K & - & I & L & LL & L \\ M & P & P & V & L & IL & L \end{array}$$

A multi-alignment of a finite set of sequences is obtained by inserting spaces in them such that all of them have equal size. A column in the alignment can contain more than one space provided it contains at least one character.

The **sum-of-pairs** (SP) score function

$$SP(column_j) = \sum_{a,b \in column_j} p(a,b)$$

$$SP(alignment) = \sum_j SP(column_j)$$

Lemma 0.1 *If* $p(-,-) = 0$ *then for any multi-alignment* α *of* $s_1, ..., s_k$

$$SP(\alpha) = \sum_{i<j} score(\alpha_{i,j})$$

where $\alpha_{i,j}$ *is the pairwise alignment induced by* α *on* s_i *and* s_j*.*

Dynamic programming for optimal multi-alignment

Fact *The problems of multi-similarity and optimal multi-alignment (i.e., when k is not fixed) are NP-hard.*

A k-dimensional array A of length $n+1$ in each dimension where n is the common length of the input sequences $s_1, ..., s_k$.

$$A[\mathbf{i}] \leftarrow \max_{b \neq 0}\{A[\mathbf{i}-\mathbf{b}] + SP(Column(\mathbf{s}, \mathbf{i}, \mathbf{b}))\}$$

where $\mathbf{b} \in \{0,\ 1\}^k$,

$$Column(\mathbf{s}, \mathbf{i}, \mathbf{b}) = (c_j)_{1 \leq j \leq k}$$

and

$$c_j = \begin{cases} s_j[i_j] & if\ b_j = 1 \\ - & if\ b_j = 0 \end{cases}$$

Theorem 0.1 *An optimal multi-alignment between k sequences of length n under the SP score function can be computed in time $O(k^2 2^k n^k)$ (in time $O(k 2^k n^k)$ if column scores can be computed in time $O(k)$).*

Trying to save time

For each pair of input sequences $s,\ t$, where $|s| = m$, $|t| = n$, $0 \le i \le m$, $0 \le j \le n$, set

$$cut_{s,t}[i,j] = sim(s[1..i], t[1..j]) + sim(s[i+1..n], t[j+1..m])$$

A cell with index $\mathbf{i} = (i_1, ..., i_k)$ in the multi-alignment matrix A for $s_1, ..., s_k$ is **relevant** with respect to lower bounds L_{s_q,s_r}, $1 \le r < q \le k$, if

$$cut_{s_x,s_y}[i_x, i_y] \ge L_{s_x,s_y}$$

for all $x,\ y$, $1 \le x < y \le k$.

The general idea is to keep a dynamic pool of relevant cells to evaluate.

Theorem 0.2 *Let α be an optimal multi-alignment between $s_1, ..., s_k$. If $SP(\alpha) \geq L$, then for $1 \leq i < j \leq k$, and the alignment $\alpha_{i,j}$ between s_i and s_j induced by α*

$$score(\alpha_{i,j}) \geq L_{s_i,s_j}$$

where

$$L_{s_i,s_j} = L - \sum_{x<y,(x,y)\neq(i,j)} sim(s_x, s_y)$$

Proof.

$$\sum_{x<y} score(\alpha_{x,y}) \geq SP(\alpha) \geq L$$

$$\sum_{x<y \& (x,y)\neq(i,j)} score(\alpha_{x,y}) \geq L - score(\alpha_{i,j})$$

$$\sum_{x<y \& (x,y)\neq(i,j)} sim(s_x, s_y) \geq L - score(\alpha_{i,j})$$

Some implementation details

$pool \leftarrow \{\mathbf{0}\}$

while *pool* not empty **do**

  $\mathbf{i} \leftarrow$ the lexicographically smallest cell in *pool*

  $pool \leftarrow pool \setminus \{\mathbf{i}\}$

  if $\mathbf{i}$ is relevant **then**

    for all $\mathbf{j}$ dependent on $\mathbf{i}$ **do if** $\mathbf{j} \notin pool$ **then**

      $pool \leftarrow pool \cup \{\mathbf{j}\}$

      $A[\mathbf{j}] \leftarrow A[\mathbf{i}] + SP(Column(\mathbf{s}, \mathbf{i}, \mathbf{j} - \mathbf{i}))$

    else

    $A[\mathbf{j}] \leftarrow \max(A[\mathbf{j}], A[\mathbf{i}] + SP(Column(\mathbf{s}, \mathbf{i}, \mathbf{j} - \mathbf{i})))$

return $A[n_1, ..., n_k]$

Star Alignments

- pick a sequence s_c among $s_1, ..., s_k$ as a center one, e.g., maximizing $\sum_{i \neq c} sim(s_i, s_c)$
- **for** $i = 1, ..., k$ where $i \neq c$ **do**

  find an optimal alignment $\alpha_{i,c}$ between s_i and s_c

  update the multi-alignment between $s_1, ..., s_{i-1}$ and s_c to an alignment between $s_1, ..., s_i$ and s_c by $\alpha_{i,c}$ and the rule "once a gap always a gap"

$$\begin{array}{c} ATTGCCATT \\ ATGGCCATT \end{array}$$

$$\begin{array}{cccc} ATT & G & CCATT & -- \\ ATC & - & CAATT & TT \end{array}$$

$$\begin{array}{ccc} ATTGCC & A & TT \\ ATCTTC & - & TT \end{array}$$

$$\begin{array}{cc} ATTGCCA & TT \\ ACTGACC & -- \end{array}$$

$$\begin{array}{cccccc} ATT & G & CC & A & TT & -- \\ ATG & G & CC & A & TT & -- \\ ATC & - & CA & A & TT & TT \\ ATC & T & TC & - & TT & -- \end{array}$$

Star alignment and edit distance score

For an alignment α between two strings s and t, let

$$eds(\alpha) = h\#spaces + \sum_{(a,b)\ is\ a\ column} cost(a,b)$$

where $h \geq 0$ and the substitution of a for b costs $cost(a,b) \geq 0$ (the cost function is assumed to be symmetric and to satisfy the triangle inequality). $eds(s,t)$ is the minimum of $eds(\alpha)$ over all alignments α between s and t.

The eds function is a distance function, i.e., it is reflexive, symmetric and satisfies the triangle inequality.

For a multi-alignment M between k sequences, let $P(M)$ be the set of all pairwise $\alpha_{i,j}$ alignments induced by M, and

$$edsSP(M) = \sum_{\alpha_{i,j} \in P(M)} eds(\alpha_{i,j})$$

Theorem 0.3 *Let M be the multi-alignment M produced by the star method for strings $s_1, ..., s_k$. $edsSP(M)$ is within $2\frac{k-1}{k}$ of the minimum.*

Proof

Let s_c be the string chosen to be the center, and let $q = \sum_{1 \le j \le k} eds(s_c, s_j)$.

$$edsSP(M) \le \sum_{\alpha_{i,j} \in P(M)} eds(\alpha_{i,j}) \le$$

$$\le \sum_{i<j} eds(s_i, s_c) + eds(s_c, s_j) \le$$

$$\le (k-1) \sum_{j} eds(s_c, s_j) \le (k-1)q$$

Let M^* be a multi-alignment for $s_1, ..., s_k$ minimizing $edsSP(M^*)$.

$$edsSP(M^*) \ge \sum_{\alpha_{i,j} \in P(M^*)} eds(\alpha_{i,j}) \ge \sum_{i<j} eds(s_i, s_j) \ge$$

$$\frac{1}{2} \sum_{i} \sum_{j} eds(s_i, s_j) \ge \frac{1}{2} \sum_{i} q \ge \frac{kq}{2}$$

Homework 2

Design a polynomial-time algorithm for optimal multi-alignment between k n-character sequences under the sum-of-pairs(SP) score function, which uses only $O(n^{k-1})$ space.