CISC 636 Computational Biology & Bioinformatics (Fall 2016)

Pairwise sequence alignment

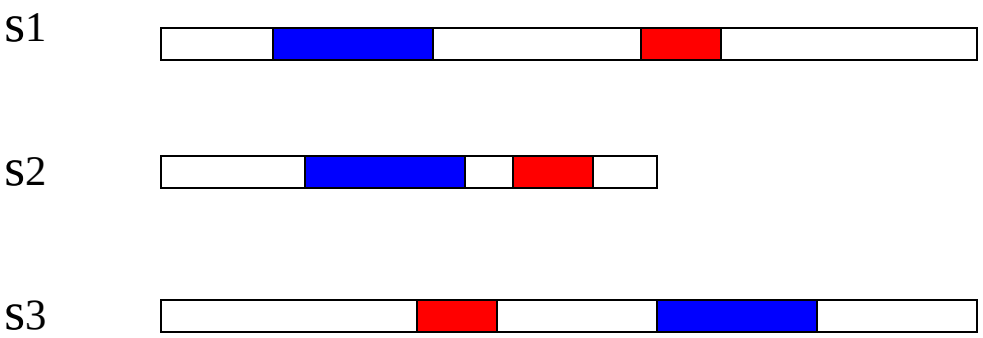
Smith-Waterman (local alignment)

Local pairwise optimal alignment

why need local alignment (vs global)?

- mosaic structure (functioning domains) of proteins, which may be caused by in-frame exchange of whole exons, or alternative splicing)

e.g., are these three sequences similar or not?



Local alignment

- Naive algorithm:
  - there are $\Theta(n^2 m^2)$ pairs of substrings; to align each pair as a global alignment problem will take O(nm); the optimal local alignment will therefore take $O(n^3 m^3)$.

- **Smith-Waterman** algorithm (dynamic programming)
  recurrence relationship

$$F(i,j) = \max \{ 0, F(i-1, j-1) + s(x_i, y_j), F(i-1,j) - d, F(i, j-1) - d \}$$

Notes: 1) For this to work, the random match model must have a negative score. Why?

2) The time complexity of Smith-Waterman is $\Theta(n\ m)$.

Example: Align HEAGAWGHEE and PAWHEAE.

Use BLOSUM 50 for substitution matrix and d=-8 for gap penalty.

		H	E	A	G	A	W	G	H	E	E
	0	0	0	0	0	0	0	0	0	0	0
P	0	0	0	0	**0**	0	0	0	0	0	0
A	0	0	0	5	0	**5**	0	0	0	0	0
W	0	0	0	0	2	0	**20**	**12**	0	0	0
H	0	10	2	0	0	0	12	18	**22**	14	6
E	0	2	16	8	0	0	4	10	18	**28**	20
A	0	0	8	21	13	5	0	4	10	20	27
E	0	0	0	13	18	12	4	0	4	16	26

AWGHE

AW-HE

Gap penalties

- Linear

$$\gamma(g) = - g\, d$$

where g is the gap length and d is the penalty for a gap of one base

- Affine

$$\gamma(g) = - d - (g-1)e$$

where d is gap-open penalty and e, typically smaller than d, is gap-extension penalty. Such a distinction is mainly to simulate the observation in alignments: gaps tend to be in a stretch.

Note: gap penalty is a sort of gray area due to less knowledge about gap distribution.

General algorithm to handle Affine gap penalty

To align two sequences x[1...n] and y[1...m],

i) if x at *i* aligns with y at *j*, a score $s(x_i, y_j)$ is added; if either x_i or y_j is a gap, a score of $\gamma(g)$ is subtracted (penalty).

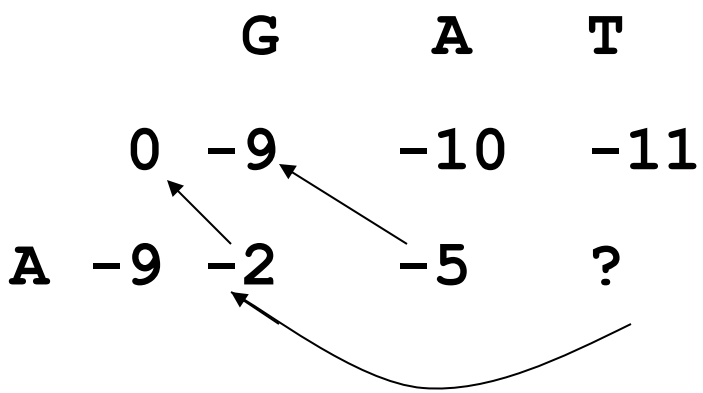
ii) The ***best*** score up to (i,j) will be

$$F(i,j) = \max \{ F(i-1, j-1) + s(x_i, y_j),$$
$$F(k,j) - \gamma(i-k), \quad k = 0, \ldots, i-1$$
$$F(i, k) - \gamma(j-k), \quad k = 0, \ldots, j-1$$
$$\}$$

This algorithm is $O(n^3)$ for n=m.

Example: Align **GAT** and **A** using the following scoring scheme:

identity 4; transition -2; transversion -4; gap penalty: op = -9, ex = -1



GAT **GAT**

-A- **A--**

Gotoh algorithm (1982) [Affine gap $\gamma(g) = -d -(g-1)e$]

$F(0,j) = \gamma(j)$, $F(i,0) = \gamma(i)$

$F(i,j) = \max \{ F(i-1, j-1) + s(x_i, y_j),$
$P(i, j),$ // gap in sequence y; vertical moves
$Q(i, j)$ // gap in sequence x; horizontal moves
$\}$

$P(0,j) = -\infty$ // so as to always take F(0,j)

$P(i,j) = \max \{ F(i-1,j) - d,$ // open a gap
$P(i-1,j) - e$ // extend a gap
$\}$

$Q(i,0) = -\infty$ //so as to always take F(i,0)

$Q(i,j) = \max \{ F(i, j-1) - d,$ //open a gap;
$Q(i, j-1) - e$ // extend a gap
$\}$

This algorithm is $O(n^2)$