## Global alignment of two strings

gap/insertion                substitution                gap/deletion

$$\mathcal{A} = \begin{bmatrix} A & C & - & - & T & C & A & G \\ A & G & T & A & T & C & - & - \end{bmatrix}$$

## Generel cost:

$$\text{cost}(\mathcal{A}) = \sum_{\text{blocks}} \text{"cost of block"}$$

Substitution cost: $s : \Sigma \times \Sigma \to \mathbb{R}$

gap cost: $g : \mathbb{N}_+ \to \mathbb{R}$

## Problem

$$\text{Sim}(A, B) = \max_{\mathcal{A}} \text{cost}(\mathcal{A})$$

$$\mathcal{A}_{opt} = \operatorname{argmax}_{\mathcal{A}} \text{cost}(\mathcal{A})$$

## Computing optimal cost with general gap cost

### Intuition

$$S(i,j) = \max_{\text{last block}} \left\{ \text{"cost of last block"} + \text{"opt. cost of rest"} \right\}$$

### Formalization

$$S(i,j) = \max \begin{cases} 0 \\ \overbrace{s(A[i],B[j]) + S(i-1,j-1)}^{M(i,j)} & \left[\begin{matrix} \sim\sim\sim & \boxed{A[i]} \\ \sim\sim\sim & \boxed{B[j]} \end{matrix}\right] \\ \overbrace{\max\limits_{0<k\leq i} \left[ S(i-k,j) - g(k) \right]}^{D(i,j)} & \left[\begin{matrix} \sim\sim\sim & \boxed{\begin{matrix} A[i-k+1] \cdots A[i] \\ - \; - \; - \; - \end{matrix}} \end{matrix}\right] \\ \overbrace{\max\limits_{0<k\leq j} \left[ S(i,j-k) - g(k) \right]}^{I(i,j)} & \left[\begin{matrix} \sim\sim\sim & \boxed{\begin{matrix} - \; - \; - \; - \\ B[j-k+1] \cdots B[j] \end{matrix}} \end{matrix}\right] \end{cases}$$

Can be implemented using dynamic programming

Running time: $\left(\frac{n}{2}\right)^2 \cdot \frac{n}{2} \leq T(n) \leq n^2 \cdot n \qquad O(n^3)$

Space consumption: $O(n^2)$

## Computing optimal cost with affine gap cost

$$g(k) = \alpha \cdot k + \beta \quad , \quad \alpha, \beta \geq 0$$

### Intuition

... as general gap cost but reduce time to compute $D(i,j)$ and $I(i,j)$. Let us consider $D(i,j)$ ...

### Trick

Consider the best deletion-block, two possibilities:

① A new block

$$\left[\overbrace{\begin{matrix} \sim A[i-1] \\ \sim B[j] \end{matrix}}^{M(i-1,j)} \boxed{\begin{matrix} A[i] \\ - \end{matrix}}\right] \; - (\alpha+\beta) \qquad \left[\overbrace{\begin{matrix} \sim\sim & - \\ \sim\sim & B[j] \end{matrix}}^{I(i-1,j)} \boxed{\begin{matrix} A[i] \\ - \end{matrix}}\right] \; - (\alpha+\beta)$$

② Continuation of existing block

$$\left[\overbrace{\begin{matrix} \sim A[i-k] \\ \sim B[j] \end{matrix} \boxed{\begin{matrix} \cdots A[i-1] \\ - \; - \; - \end{matrix}}}^{D(i-1,j)} \boxed{\begin{matrix} A[i] \\ - \end{matrix}}\right] \; - \alpha$$

Hence,

$$D(i,j) = \max \{ M(i-1,j) - (\alpha+\beta), \; I(i-1,j) - (\alpha+\beta), \; D(i-1,j) - \alpha \}$$

$$= \max \{ S(i-1,j) - (\alpha+\beta), \; D(i-1,j) - \alpha \}$$

## Computing optimal cost with affine gap cost

$$S(i,j) = \max \begin{cases} 0 & i=0,\ j=0 \\ S(i-1,j-1) + s(A[i],B[j]) & i>0,\ j>0 \\ D(i,j) & i>0,\ j\geq 0 \\ I(i,j) & i\geq 0,\ j>0 \end{cases}$$

$$D(i,j) = \max \begin{cases} S(i-1,j) - (\alpha+\beta) & i>0,\ j\geq 0 \\ D(i-1,j) - \alpha & i>0,\ j\geq 0 \end{cases}$$

$$I(i,j) = \max \begin{cases} S(i,j-1) - (\alpha+\beta) & i\geq 0,\ j>0 \\ I(i,j-1) - \alpha & i\geq 0,\ j>0 \end{cases}$$

Can be implemented using dyn.prog/memorization using 3 tables of size $\Theta(n \cdot m)$.

Time:

Space:

What about retrieving an optimal alignment?