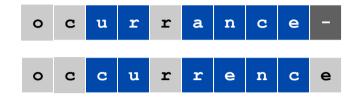
6.6 Sequence Alignment



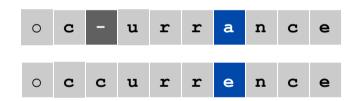
String Similarity

How similar are two strings?

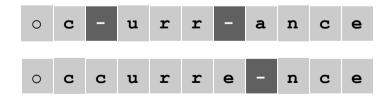
- ocurrance
- occurrence



6 mismatches, 1 gap



1 mismatch, 1 gap



0 mismatches, 3 gaps



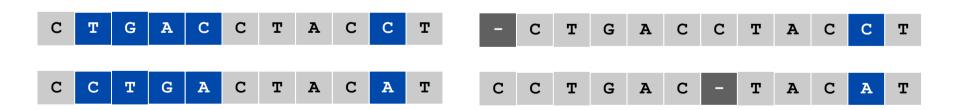
Edit Distance

Applications.

- . Basis for Unix diff.
- . Speech recognition.
- . Computational biology.

Edit distance. [Levenshtein 1966, Needleman-Wunsch 1970]

- . Input Parameters:
 - . Gap penalty δ ;
 - . Mismatch penalty α_{pq} .
- . Cost = sum of gap and mismatch penalties.



 $2\delta + \alpha_{CA}$

$$\alpha_{TC} + \alpha_{GT} + \alpha_{AG} + 2\alpha_{CA}$$

Feasible Solution: Sequence Alignment

INPUT: parameter δ ; matrix $\alpha_{i,j}$;

two strings $X = x_1 x_2 \dots x_m$ and $Y = y_1 y_2 \dots Y_n$

GOAL: find an alignment of minimum cost.

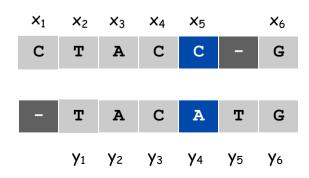
Def. An alignment M is a set of ordered pairs x_i-y_j such that each item occurs in at most one pair and no **crossings**.

Def. The pair x_i-y_j and $x_{i'}-y_{j'}$ cross if i < i', but j > j'.

$$Cost(M) = \sum_{xi-yj \in M} \alpha_{i,j} + \sum_{xi \notin M} \delta + \sum_{yi \notin M} \delta$$
Mismatches
$$Gaps$$

Ex: CTACCG VS. TACATG.

Sol: $M = x_2 - y_1, x_3 - y_2, x_4 - y_3, x_5 - y_4, x_6 - y_6.$





Designing the Dynamic Programming

FACT. Let M be any Alignment of X and Y.

IF the pair (Xm,Yn) is not in M THEN

either x_m is not matched in M or y_n is not matched in M.

Proof.

Otherwise, a cross would occur!!!!

Sequence Alignment: Problem Structure

```
Def. OPT(i, j) = min cost of aligning strings x_1 x_2 ... x_i and y_1 y_2 ... y_j.
 . Case 1: OPTIMAL SOL matches x<sub>i</sub>-y<sub>j</sub>. THEN:
     - pay for x_i-y_j + min cost of aligning the two strings
                        mismatch x_1 x_2 \dots x_{i-1} and y_1 y_2 \dots y_{j-1}
 . Case 2a: OPTIMAL SOL leaves xi unmatched. THEN
      - pay gap for x_i + min cost of aligning x_1 x_2 \dots x_{i-1} and y_1 y_2 \dots y_j
 . Case 2b: OPTIMAL SOL leaves y<sub>j</sub> unmatched.
      - pay gap for y_j + min cost of aligning x_1 x_2 ... x_i and y_1 y_2 ... y_{j-1}
OPT(i,j) = \begin{cases} \alpha_{i,j} + OPT(i-1,j-1) \\ \delta + OPT(i-1,j) \\ \delta + OPT(i,j-1) \end{cases}
i * \delta
                                                                                           if i = 0
                                                                                         otherwise
                                                                                          if j = 0
```

Sequence Alignment: Algorithm

Analysis. $\Theta(mn)$ time and space.

English words or sentences: $m, n \le 10$.

Computational biology: m = n = 100,000. 10 billions ops OK, but 10GB array?



HOMEWORKS

- Give the formal definition of the SEQUENCE ALIGNMENT PROBLEM
- Improve the **space** complexity of the Dynamic Programming Algorithm (Hint: Do you need to store **all** matrix elements, for **all** the steps?