String Similarity

How similar are two strings:

ocurrance
occurrence

Alignment 1

```
o c u r r a n c e
o c c u r r e n c e
```

6 mismatches
1 gap

Alignment 2

```
o c _ u r r a n c e
o c c u r r e n c e
```

1 mismatch
1 gap

Alignment 3

```
o c _ u r r a _ n c e
o c c u r r _ e n c e
```

0 mismatches
3 gaps

Applications:

- spelling correction
- speech recognition
- computational biology

Edit distance (Levenshtein 1966, Needleman/Wunsch 1970, ...)

- gap penalty $\delta > 0$
- mismatch penalty $\alpha_{p,q}$ (p, q characters)

Levenshtein: $\delta = 1$, $\alpha_{pq} = 1$ if $p \neq q$, 0 otherwise

- cost of alignment: sum of gap and mismatch penalties

Problem: Given $x = [x_1, \dots, x_m]$, $y = [y_1, \dots, y_n]$, find (alignment of) minimal cost

Alignment (Formalization): sequence of ordered pairs (u_k, v_k)

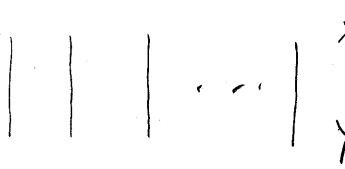
- $u_k \in \{x_1, \dots, x_m, -\}$, $v_k \in \{y_1, \dots, y_n, -\}$
- each x_i and y_j occurs exactly once in a pair
- order of the x_i in alignment is the same as in x (same for y_j and y)

We consider the Levenshtein distance:

Cost of aligning $x[1..i]$ and $y[1..j]$.

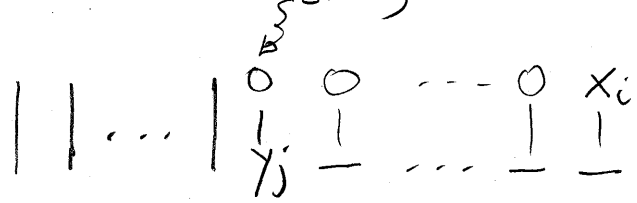
The best alignment can be one of three:

Case 1:



$\Rightarrow$ cost $\geq$ cost$(x[1..i-1], y[1..j-1])$ $+ \alpha_{x_i, y_j}$

Case 2:

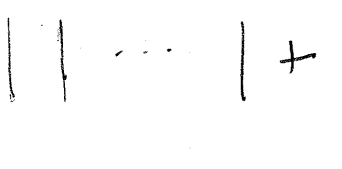


$\Rightarrow$ cost $\geq$ cost$(x[1..i-1], y[1..j])$ $+ \delta$

Case 3: symmetric

cost $\geq$ cost$(x[1..i], y[1..j-1]) + \delta$

Conversely, consider the best alignment for $x[1..i-1]$, $y[1..j-1]$ and add (x_i, y_j):



$\Rightarrow$ cost$(x[1..i-1], y[1..j-1])$ $+ \alpha_{x_i y_j} \geq$ cost

Consider best alignment for $x[1..i-1]$ and $y[1,\dots j]$:

$x[1..i-1]$

$|\,|\,\dots\,|\,+\,\underline{\overset{x_i}{\vdots}}$ $\Rightarrow$ $\text{cost}(x[1..i-1], y[1\dots j] + \delta$

$y[1\dots j]$ $\geq$ cost

Also: $\text{cost}(x[1..i], y[1..j-1]) + \delta \geq \text{cost}.$

We conclude: Let $c[i,j] := \text{cost}(x[1..i], y[1..j])$.

Recurrence for $c[i,j]$:

$$c[0,j] = \delta \cdot j$$

$$c[i,0] = \delta \cdot i$$

$$c[i,j] = \min\big(\, c[i-1, j-1] + \alpha_{x_i, y_j},$$
$$c[i-1, j] + \delta,$$
$$c[i, j-1] + \delta\big)$$

Compute $c[i,j]$ with dynamic programming

Example: HOUSE vs HOST

		H	O	U	S	E
	0	1	2	3	4	5
H	1	0	1	2	3	4
O	2	1	(0)	(1)	2	3
S	3	2	1	1	(1)	2
T	4	3	2	2	2	(2)

Trace: where did the minimum come from?

Alignment

H	O	U	S	E
H	O	–	S	T