Exercise sheet 4: Sequence Alignment - affine gap costs

Exercise 1 - Waterman-Smith-Beyer traceback

Within the algorithm of Waterman, Smith and Beyer arbitrarily large gaps are considered. Thus, also the traceback has to investigate all gap sizes. This can be done following two strategies:

- check in increasing gap length (start with smallest gap)
- check in decreasing gap length (start with largest gap)

Question 1A Given a gap function, are the strategies (measured in absolute runtime or number operations) expected to be equally performant or is one of them better than the other?

Possible Answers

☐ both are equally performant.
☐ checking in increasing direction performs better
☐ checking in decreasing direction performs better

Solution

☒ both are equally performant.
☐ checking in increasing direction performs better
☐ checking in decreasing direction performs better

(You still have to check all possible gap sizes)

Question 1B) Does the order of checking insertion and deletions change the runtime?

Possible Answers

☐ yes, checking insertions first will perform better
☐ yes, checking deletions first will perform better
☐ no, the order doesn't affect performance
☐ checking in an alternating way might improve performance

Solution

☐ yes, checking insertions first will perform better
☐ yes, checking deletions first will perform better
☒ no, the order doesn't affect performance
☐ checking in an alternating way might improve performance

(You still have to check all possible gaps and sizes)

Exercise 2 - Waterman-Smith-Beyer traceback

Given the Waterman-Smith-Beyer algorithm with the following scoring function:

$$s(x,y) = \begin{cases} -1 & if\ x = y \\ +1 & else \end{cases} \tag{1}$$

$$g(k) = 1 + k \tag{2}$$

When aligning sequences of very different lengths the penalizing of leading and trailing gaps, i.e unaligned sequence ends, dominates the alignment score.

We distinguish between leading and trailing gaps (example):

```
A-CC-CC-G
-TCCGCCT-
```

In this case the leading gaps are:

```
A-
-T
```

The trailing gaps are:

```
-G
T-
```

The algorithm by Waterman, Smith and Beyer can be adapted to treat end gaps with a score of 0. The adapted recursion formula is defined here, where n and m are the lengths of the respective sequences:

Let A and B be sequences of length n and m respectively

$$a_i \in A,\ with\ 1 \leq i \leq n b_j \in B,\ with\ 1 \leq j \leq m$$

$$D_{i,0} = D_{0,j} = 0 \qquad (a) \tag{3}$$

$$D_{i,j} = min \begin{cases} D_{i-1,j-1} + s(a_i, b_j) & (b) \\ \min_{i \leq k < j} D_{i,j-k} + g(k) & (c) \\ \min_{i \leq k < i} D_{i-k,j} + g(k) & (d) \\ D_{i,0} + 0 & (e) \\ D_{0,j} + 0 & (f) \\ \min_{1 \leq k \leq j} D_{i,j-k} + 0 & (g) \\ \min_{i \leq k \leq i} D_{i-k,j} + 0 & (h) \end{cases} \tag{4}$$

Question 2A Describe the recursion parts (labels a - h)

(Use the hint to match descriptions to the correct parts)

Hint: Descriptions

- ☐ if (i = n), k trailing gaps in A
- ☐ k normal gaps in B
- ☐ j leading gaps in A (inner block of gaps)
- ☐ match/missmatch case
- ☐ i leading gaps in B (inner block of gaps)
- ☐ outer block of gaps in one sequence
- ☐ if (j = m), k trailing gaps in B
- ☐ k normal gaps in A

Solution

a. outer block of gaps in one sequence
b. match/missmatch case
c. k normal gaps in A
d. k normal gaps in B
e. j leading gaps in A (inner block of gaps)
f. i leading gaps in B (inner block of gaps)
g. if (i = n), k trailing gaps in A
h. if (j = m), k trailing gaps in B

Question 2B Why is the scoring function $s(x, y)$ not a metric.

Hint: Multiple Choice

- ☐ The identity clause is violated
- ☐ The symetry clause is violated
- ☐ The triangle inequality clause is violated
- ☐ It is possible to create a metric scoring function leading to the same optimal alignments where end gaps are free
- ☐ For the given scoring function a match case is as favorable as a the leading end gap case

Solution

- ☒ The identity clause is violated
- ☐ The symetry clause is violated
- ☒ The triangle inequality clause is violated
- ☐ It is possible to create a metric scoring function leading to the same optimal alignments where end gaps are free
- ☐ For the given scoring function a match case is as favorable as a the leading end gap case

The identity clause is violated: for x = y, s(x,y) = -1, which is not 0.

The triangle inequality clause is violated: s(x,x) > s(x,x) + s(x,x), when x=y=z

It is not possible to create a metric. To create a metric we need to ensure the indentity clause is fulfilled, by setting s(x,y) = 0, when x=y. Thereby, we would cause the match case to be as favorable as the leading/trailing end gap case. Setting s(x,y) = 0 leads to more optimal alignments, including an alignment where both sequences are aligned to gaps.

Exercise 3 - Gotoh Algorithm

Consider the following sequences $S1$, $S2$ and the similarity scoring via $s(x, y)$ and $g(k)$.

$$S1 = TCCGA \tag{5}$$

$$S2 = TACGCAGA \tag{6}$$

$$s(x,y) = \begin{cases} +1 & if\ x = y \\ 0 & else \end{cases} \tag{7}$$

$$g(k) = -4 - 1 * k \tag{8}$$

Question 3A Which optimization scheme (minimization/maximization) needs to be applied to generate an useful sequence alignment.

Solution Maximization since we are scoring a similarity and not a distance.

Question 3B Fill the dynamic programming matrices $D_{i,j}$, $Q_{i,j}$ and $P_{i,j}$ using the Gotoh algorithm! (Remember: $D_{i,j}$ is the match/mismatch matrix. $Q_{i,j}$ corresponds to gaps in $S1$ whilst $P_{i,j}$ corresponds to gaps in $S2$)

Hint1: Formulae

$$g(k) = \alpha + k * \beta \tag{9}$$

$$\tag{10}$$

$$D_{i,j} = min \begin{cases} D_{i-1,j-1} + s(a_i, b_i) \\ P_{i,j} \\ Q_{i,j} \end{cases} \tag{11}$$

$$\tag{12}$$

$$Q_{i,j} = min \begin{cases} D_{i,j-1} + g(1) \\ Q_{i,j-1} + \beta \end{cases} \tag{13}$$

$$\tag{14}$$

$$P_{i,j} = min \begin{cases} D_{i-1,j} + g(1) \\ P_{i-1,j} + \beta \end{cases} \tag{15}$$

Hint2: Add the missing values

Solution

Question 3C Calculate all optimal alignments and the according score!

Solution Score: -3

```
T---CCGA    TCC---GA    TCCG---A
|   |:||    |:|   ||    |:||   |
TACGCAGA    TACGCAGA    TACGCAGA
```

Question 3D Calculate the alignments using the Waterman-Smith-Beyer algorithm instead.

Solution The same alignments as in 3C, since the same scoring function is optimized. Calculation is just less efficient.

Exercise 4 - Programming assignment

Programming assignments are available via Github Classroom and contain automatic tests.

We recommend doing these assignments since they will help you to further understand this topic.

Access the Github Classroom link: Programming Assignment: Sheet 04.