Homework 2 Solution

CS 4364/5364
Spring 2021

Due: 25 February 2021

This homework is worth 5 points out of the total 25 points of homework in the class.

1. *Given two sequences S and T (not necessarily the same length), let G, L, and H be the scores of an optimal global alignment, an optimal local alignment, and an optimal global alignment without counting the indels at the beginning of S and the end of T, respectively.*

   (a) Give an example of S and T so that all 3 scores, G, L, and H, are different.

   (b) Prove or disprove the statement $L \geq H \geq G$.

   part (a) Let $A \in \Sigma^n$ be an arbitrary string of length $n \geq 1$ over some alphabet, and let $x, y, z, w \notin \Sigma$ be 4 characters not contained in A. Define $S = \{x\}^k \cdot A \cdot \{y\}^\ell$ and $T = \{z\}^\ell \cdot A \cdot \{w\}^k$, where $\cdot$ is the concatination operator, $\{a\}^i$ is a string consisting of the letter a repeated i times, and $k > \ell$.

   The best local alignment will align the substrings A in both S and T to each other, any extension would decrease the score (assuming the indel and mismatch values would penalize those operations). The best semi-global alignment would add 2ℓ mismatches or indels (whichever provides a lower penalty) to the local alignment since the beginning of T and the end of S must be aligned and those characters cannot be matches. The best global alignment would have an additional $2(k - \ell)$ mismatches or $2k$ indels (again depending on the penalties).

   part (b) Let $f(\mathbb{A})$ be the alignment score for alignment $\mathbb{A}$. And let $G(A, B)$ be the optimal global alignment of sequences A and B. In the local sequence alignment algorithm we are searching over the set of all substrings, so we can actually restate that as searching for the best alignment out of the set

$$\mathcal{A}_\ell =: \{G(S[i...j], T[k...\ell]) \mid 1 \leq i, j \leq |S|, 1 \leq k, \ell \leq |T|\} \cup \{\mathbb{A}_\varnothing\},$$

   where $\mathbb{A}_\varnothing$ is the alignment of two empty strings. We can rewrite the definition of local alignment as: $\mathbb{A}_\ell =: \operatorname{argmax}_{\mathbb{A} \in \mathcal{A}_\ell} \{f(\mathbb{A})\}$.

Similarly, the set of alignments being considered by semi-global alignment (as defined in the problem) is

$$\mathcal{A}_s =: \{G(S[i...|S|], T[1...\ell]) \mid 1 \le i \le |S|, 1 \le \ell \le |T|\} \cup \{\mathbb{A}_\varnothing\}$$

and the semi-global alignment problem reduces to $\mathbb{A}_s =: \text{argmax}_{\mathbb{A}\in\mathcal{A}_s} \{f(\mathbb{A})\}$. Finally let the global alignment be $\mathbb{A}_g =: G(S,T)$ and thus the set considered by the problem contains only one element $\mathcal{A}_g =: \{\mathbb{A}_g\}$.

As defined $\mathcal{A}_g \subset \mathcal{A}_s \subset \mathcal{A}_\ell$. We know that $f(\mathbb{A}_g) \le f(\mathbb{A}_s)$ since $\mathbb{A}_g \in \mathcal{A}_s$, and $f(\mathbb{A}_s) \ge f(\mathbb{A}')$ for all $\mathbb{A}' \in \mathcal{A}_s$. The same argument applies to the other cases. □

2. *Given two strings $S[1...n]$ and $T[1...m]$, we would like to find the two non-overlapping alignments $(S[i_1...i_2], T[j_1...j_2])$ and $(S[i_3...i_4], T[j_3...j_4])$ such that $i_2 < i_3$ and $j_2 < j_3$ and the total alignment score is maximized in running time $O(mn)$. Hints: remember that the local alignment between two sequences is the alignment of a pair of substrings from S and T such that the alignment score is maximized, and that the question does not say anything about the relationship between i_1 and i_2 nor i_3 and i_4 that if $i_3 > i_4, S[i_3...i_4]$ is the empty string.*

   Algorithm

   (a) using the recurrence for local alignment, create the matrix V, and a maximum value array M, where

   $$M(i,j) =: \max \begin{cases} M(i-1,j-1) \\ M(i-1,j) \\ M(i,j-1) \\ V(i,j) \end{cases}.$$

   The M array keeps track of the maximum value local alignment between $S[1...i]$ and $T[1...j]$.

   (b) using the same procedure as Step 2a create V' and M' for the reverse of S and T.

   (c) perform a linear scan to find

   $$(\hat{i},\hat{j}) = \underset{(i,j):1\le i<n, 1\le j<m}{\text{argmax}} \{M(i,j) + M'(i+1,j+1)\}$$

   (d) let (i_2, j_2) be the index in $V[i_2][j_2] = M(\hat{i},\hat{j})$ (assuming $M(i,j) \ne 0$, if it is 0, set $(i_2,j_2) = (1,1)$)

   (e) perform a traceback in V to find (i_1, j_1). (if $V(i_2,j_2) == 0$ set $(i_1,j_1) = (0,0)$ to signify the empty strings.)

(f) let (i_3, j_3) be the index in $V'[i_3][j_3] = M'(\hat{\imath}+1, \hat{\imath}+1)$ (assuming $M'(\hat{\imath}+1, \hat{\imath}+1) \neq 0$, if it is 0, set $(i_3, j_3) = (n, m)$).

(g) perform a traceback in V' to find (i_4, j_4). (if $V(i_3, j_3) == 0$ set $(i_4, j_4) = (n - 1, m - 1)$ to signify the empty strings.)

The linear scan in Step 2c finds the boundary between the regions where the two maximal sets of substrings exist. Keeping the M array keeps us from having to do a quadratic time scan for each possible partition of the local alignment space. We can then find the maximal substrings in each of the regions using the techniques we already know. It is possible for one of the two substrings to be empty, in this case either $M(i,j) = 0$ or $M'(i+1, j+1) = 0$, in that case the whole maximal set of substrings would be in one partition, and nothing positive would be in the other.

Filling in V, M, V', and M' in Steps 2a & 2b takes $O(mn)$-time each. The linear scan in Step 2c takes $O(mn)$-time. The two linear scans in Steps 2d & 2f take a total maximum of $O(mn)$-time. Finally, the tracebacks in Steps 2e & 2g take a total maximum time of $O(mn)$-time. This means there is a constant number of $O(mn)$-time steps. $\square$