Homework 3

CS 4364/5364
Spring 2022

Due: 23 February 2022

1. **(25 points)** ***Profile Alignment Problem*** **Given two sequence profiles T and S, of sizes $\sigma \times n$ and $\sigma \times m$ respectively (that is each represents a sequence of length n $[m]$, but with probabilities of each character from the alphabet at each position), determine the optimal alignment (i.e. which columns of S align with which columns of T) under the scoring scheme δ.**

   Your task: modify the Needlman-Wunch global alignment algorithm to consider these profiles rather than sequences. You can assume that the replacement costs are defined in a function $\delta(a, b) \rightarrow \mathbb{Z}, \forall a, b \in \Sigma \cup \{'-'\}$. Give the algorithm, an explanation of correctness, and analysis of it's running time.

   An example alignment is shown below over the alphabet $\Sigma = \{\texttt{A}, \texttt{C}, \texttt{T}, \texttt{G}\}$, as well as it's alignment score. Note that the score for a column is now no longer the value of δ for the two characters being aligned, but the weighted sum of these values.

		1	2	3	4	5	6	7	8	9	10	11	12	13
S	A	0.9	0.0	0.0	1.0	0.9	-	-	-	0.0	0.0	0.2	0.6	
	C	0.0	0.2	0.5	0.0	0.1	-	-	-	0.4	0.3	0.2	0.4	
	T	0.1	0.1	0.2	0.0	0.0	-	-	-	0.6	0.3	0.3	0.0	
	G	0.0	0.7	0.3	0.0	0.0	-	-	-	0.0	0.4	0.3	0.0	
		|	|		|	|				|	|	|	
T	A	0.7	0.1	-	0.7	0.2	0.4	0.0	1.0	0.0	0.1	0.0	-	
	C	0.0	0.1	-	0.1	0.8	0.2	0.0	0.0	0.5	0.3	0.0	-	
	T	0.3	0.1	-	0.1	0.0	0.2	0.3	0.0	0.5	0.3	0.5	-	
	G	0.0	0.7	-	0.1	0.0	0.2	0.7	0.0	0.0	0.3	0.5	-	
Column Score		0.61	1	-2.7	1	-0.2	-2.8	-2.3	-3	-0.3	-0.3	-0.3	-3	

Figure 1: Alignment of two profiles

δ	A	C	T	G	-
A	2	-1	-2	-1	-3
C	-1	2	-2	-1	-3
T	-2	-2	1	-1	-3
G	-1	-1	-1	3	-2
-	-3	-3	-3	-3	x

Figure 2: Scoring scheme

Solution

the algorithm

- Let there be a function $M : \mathbb{R}^\sigma \times \mathbb{R}^\sigma \times \mathbb{R}^{\sigma \times \sigma} \to \mathbb{R}$ which takes as input two profiles (real-valued vectors of length σ) and a delta function (a real-valued $\sigma \times \sigma$ matrix) then returns the cost of aligning the two columns under the delta function. Define it as follows:

$$M(A, B, \delta) := \sum_{a,b \in \Sigma} A[a] * B[b] * \delta(a, b).$$

- Let there be two other functions of a similar type; $I : \times\mathbb{R}^\sigma \times \mathbb{R}^{\sigma \times \sigma} \to \mathbb{R}$ and $D : \times\mathbb{R}^\sigma \times \mathbb{R}^{\sigma \times \sigma} \to \mathbb{R}$ that provide the score of inserting or deleting a profile respectively. Define them as:

$$I(B, \delta) := \sum_{b \in \Sigma} B[b] * \delta('-', b),$$

and

$$D(A, \delta) := \sum_{a \in \Sigma} A[a] * \delta(a, '-').$$

- Alter the recurrence relation from Needlman-Wunsch to consider these new scores as follows, assuming we can represent each sequence of profiles such that the first dimension is the position in the sequence and the second is the frequency of a character:

$$V(i,j) = \max \begin{cases} V(i-1, j-1) + M(S[i], T[j], \delta) & \text{//align positions } i \text{ and } j \text{ from} \\ & \text{//} S \text{ and } T \text{ respectively} \\ V(i-1, j) + D(S[i], \delta) & \text{//delete position } i \text{ from } S \\ V(i, j-1) + I(T[j], \delta) & \text{//insert position } j \text{ from } T \end{cases}$$

- Initialize the V matrix as follows:

$$\begin{aligned} &V(0,0) = 0 \\ &V(i,0) = V(i-1,0) + D(S[i]) \quad 1 \le i \le n \\ &V(0,i) = V(0,i-1) + I(T[j]) \quad 1 \le i \le m \end{aligned}$$

- Fill in the remaining parts of V table and perform traceback as described in the original algorithm outputting profiles instead of single characters in the alignment.

correctness

The initializations match the original initializations in that the best alignment between any string and the empty string is to remove (insert/delete) all of the characters up to this point (this is the only choice) The choice made in the recurrence relation is still to decide on what the last column should be in the optimal alignment between the prefixes $S[1...i]$ and $T[1....j]$, thus as long as the previous steps provide the optimal scores you can intuition though in a inductive manner to show each step is then correct. Since we know the base cases (initializations) are true, it follows that the best alignment score of the full sequences will be held in $V(n, m)$ as in the original algorithm. Because the paths are the same as in the original algorithm with respect to how to output matches/mismatch, insertion, and deletion columns we know the traceback will output the correct alignment in line with the example.

run-time analysis

the first two bullets are simply definitions, and thus have no running time. The initialization fills in $n + m + 1$ cells of the table, and from the definition we can see that calculating both I and D take $O(\sigma)$ time. Therefore, the total running time of the initializations is $O(\sigma(m + n))$. Filling in the table still means calculating values for $n \times m$ values, but at each cell we must run M, I, and D which take $O(\sigma^2)$, $O(\sigma)$, and $O(\sigma)$ time respectively. This means calculating the value at each cell is $O(\sigma^2)$-total time, and thus filling in V is $O\left(\sigma^2(mn)\right)$-time. The alignment can be at most $n + m$ columns long, and printing each profile can take $O(\sigma)$-time, therefore the traceback and printing take $O(\sigma(m + n))$. Thus the overall running time, dominated by filling in V, takes $O\left(\sigma^2(mn)\right)$-time.