

Problem Set 4

Due at the beginning of class on Wednesday, March 11.

Collaboration is allowed in groups of at most three students, but you must submit separate write-ups. Please also write the names of all your collaborators on your submissions. If you are working alone, we will drop the problem with the lowest score.

Problem 1. (25 points) *Chaining Local Alignments*

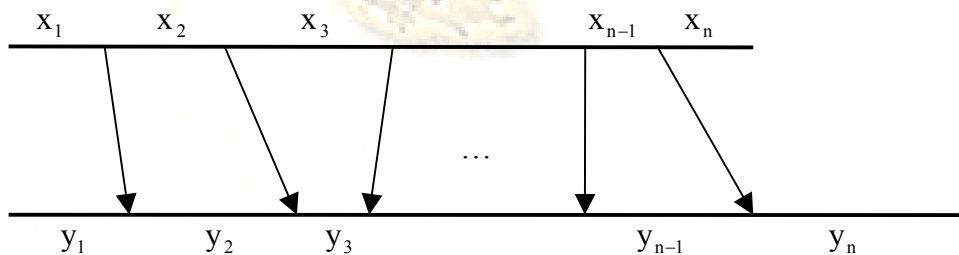
(a)(10 points) Consider the following problem:

Let S be a sequence of k numbers n_1, \dots, n_k . Let each number have an associated weight w_1, \dots, w_k . We want to find the heaviest increasing subsequence of S ; that is, a subsequence $n_{i_1} < \dots < n_{i_m}$ (where $i_1 < \dots < i_m$), such that $w_{i_1} + \dots + w_{i_m}$ is maximum.

Show how to solve this problem in $O(k \log k)$ time by reducing it to the chaining problem presented in class, for which a sparse dynamic programming algorithm was described.

(b) Let X and Y be DNA sequences of lengths L_X and L_Y respectively. Let A_X and A_Y be two sequences of $(n - 1)$ anchors that will be used to align X and Y . Anchors in A_X and A_Y are ordered from left to right (anchor i of A_X will match anchor i of A_Y , and is located to the left of anchor j of A_X if and only if $i < j$).

Throughout this problem, assume the lengths of the anchors themselves are negligible. Let x_i be the distance between anchors $i-1$ and i in A_X , and similarly for y_i . We also include the distances between the beginning of the sequence and the first anchor, and between the last anchor and the end; therefore, the sets $X_d = x_1, \dots, x_n$ and Y_d both have cardinality n , one greater than the number of anchors.



We will globally align X and Y by fixing the alignments of corresponding anchors to each other, and using Needleman-Wunsch to align the portions of X and Y between neighboring anchors. We want to calculate the running time of this anchored alignment in terms of x_i and y_i . For the following two questions, consult the list of definitions at the end of this problem.

(i) (7 points) Suppose $L_X = L = L_Y$ and the anchors in X and Y are spaced in exactly the same way; that is, $x_i = y_i$ for all i. Write down the running time as a function of only those terms: L, n, and $\text{Var}(X_d)$.

(ii) (8 points) Without making any assumptions about the relationship between the anchors in X and Y, write down the running time as a function of only those terms: L_X , L_Y , n, and $\text{Cov}(X_d, Y_d)$.

Definitions:

Mean: $\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i$

Variance: $\text{Var}(X_d) = \frac{1}{n} \sum_{i=1}^n (x_i - \bar{x})^2$

Covariance: $\text{Cov}(X_d, Y_d) = \frac{1}{n} \sum_{i=1}^n ((x_i - \bar{x})(y_i - \bar{y}))$

Problem 2. (25 points) Multiple Sequence Alignments

(a)(12 points) Consensus multiple alignment versus sum-of-pairs multiple alignment.

Definitions: (adapted from Gusfield, p. 352)

- (1) Given a multiple alignment M of a set of strings S, the *consensus character* of column i of M is the character that maximizes the summed score between the character and all the characters in column i. (In case of ties, say by convention that we prefer A over C over G over T over ‘gap’). The score of (gap, gap) is 0. Let d(i) denote that maximum summed score in column i.
- (2) The *consensus string* S_M derived from alignment M is the concatenation of the consensus characters for each column of M.
- (3) The *alignment score* of S_M equals to the sum of column scores $d(S_M) = d(1) + \dots + d(m)$, where M has m columns.
- (4) The *optimal consensus multiple alignment* is a multiple alignment M for input string set S whose consensus string S_M has largest alignment score over all possible multiple alignments of S.

Example: $S = \{ AGC\text{C} \quad AC\text{C} \quad TCC\}$, and match, mis match, gap = +2, -2, -3. Consider the following alignments:

$M_1: \{ AGC\text{C}A-C\text{T}-CC \quad \}; S_{M1} = A-C\text{C}$ and $d(S_{M1}) = (+2) + (-3) + (+6) + (+6) = 11.$
 $M_2: \{ AGC\text{C}A-C\text{G}-TCC \quad \}; S_{M2} = AGC\text{C}$ and $d(S_{M2}) = (+1) + (-3) + (+6) + (+6) = 10.$

Show an example with three or more sequences where **all** optimal multiple alignments according to the above model are different from **all** optimal alignments according to the Sum-Of-Pairs model. **In other words, since there may be several equally-scoring optimal alignments, the**

set of optimal alignments for the consensus model must be disjoint from the set of optimal alignments for the Sum-Of-Pairs model.

Assume **either** a match, mismatch, and gap penalty of (+2, -2, -3) **or** (+2, -1, -1) (**you may find the second set of scoring parameters easier to prove**). Let the alphabet be {A,C,G,T}.

(b) (13 points) Phylogenetic-tree-based alignment.

Definitions:

- (1) Given an input rooted binary tree T with a distinct string (from a set of strings S) written at each leaf, a *phylogenetic alignment* for T is an assignment of one string to each internal node of T , **subject to the additional constraint described below**. Note that the strings assigned to internal nodes need not be distinct and need not be from the input strings S .
- (2) If strings S and S' are assigned to the endpoints of an edge (i, j) , then that edge has *edge distance* $D(S, S')$, which is simply the maximum pair-wise alignment score between the two strings S and S' . The distance of a phylogenetic alignment is the total of all edge distances in the tree.
- (3) The *phylogenetic alignment* problem for T is to find an assignment of strings to internal nodes of T (one string to each node) that maximizes the distance of the alignment.

Additional constraint:

In the maximum pair-wise alignment between a leaf node x and an internal node S , a letter in x must match or mismatch some letter in S , and not be gapped. Also, the maximum pair-wise alignment between two adjacent internal nodes S and S' may not have any gaps. Then, constructing the multiple alignment of the input sequences given the phylogenetic alignment is straight-forward.

Note: it is also possible to define the problem where T is unrooted. If you prefer that definition, please go ahead and use it instead.

Example:

$S = \{x = AC\text{C}y = \text{AGCC}, z = \text{TC}\}$, and match, mismatch, gap = +2, -2, -3. Let $T = [\text{Nodes} = \{x, y, z, v_{yz}, v_{xyz}\}; \text{Edges} = \{(x, v_{xyz}), (v_{xyz}, v_{yz}), (v_{yz}, y), (v_{yz}, z)\}, \text{Root} = v_{xyz}]$, with leafs $\{x, y, z\}$ and root v_{xyz} .

Here is a phylogenetic alignment, which is just a labeling of the internal nodes: Label v_{yz} with “AGCC, and v_{xyz} with “AGCC. Then, the alignment score is $D(x, v_{xyz}) + D(v_{xyz}, v_{yz}) + D(v_{yz}, y) + D(v_{yz}, z) = (+3) + (+8) + (+8) + (-1) = 18$.

Show an example with three or more sequences where **all** their optimal phylogenetic alignments differ from **either** the set of optimal consensus multiple alignments **or** the set of optimal Sum-Of-Pairs multiple alignments.

Similar to the previous problem, assume **either** a match, mismatch, and gap penalty of (+2, -2, -3) **or** (+2, -1, -1). Let the alphabet be {A,C,G,T}.

Problem 3. (25 points) *Phylogeny*

(a) (6 points) Given the following sequences, aligned with no gaps:

X: TACCCGAT

Y: TAAACGAT

Z: AAAACGCG

W: AAAACGAT

Define the distance d_{uv} between two sequences u and v to be simply the number of letter substitutions (Hamming distance). Is this distance function ultrametric on these four sequences? Regardless of ultrametricity, build the average linkage (a.k.a., UPGMA) tree T_{AL} for these four sequences.

(b) (7 points) On the same sequences, run the Parsimony algorithm as described in lecture on T_{AL} from part (a). Is this the optimal parsimony tree? Either argue why it is, or find the optimal parsimony tree T_P .

(c) (4 points) Recall that the maximum likelihood distance between two sequences under the Jukes-Cantor model is given by $d = -\frac{3}{4} \ln(1 - \frac{4}{3} p)$, where p is the fraction of sites where the two sequences differ. Compute these distances for all pairs of the sequences in part (a).

(d) (8 Points) Assume the optimal parsimony tree T_P you found in part (b) is the correct tree, and the parsimony algorithm correctly determined the sequences at the internal nodes of this tree. Use these sequences to compute the Jukes-Cantor distance for all internal edges of the tree. Do these distances and the distances you computed in part (c) satisfy the additivity assumption?

Problem 4. (25 points) *Gene Prediction*

A simple strategy for locating genes in compact genomes not containing introns is to look for long open reading frames (ORFs). An ORF is defined as a sequence of ?NA beginning with a start codon (ATG) and containing no in -frame stop codons (TAA, TAG, or TGA). ORF scanning works because genes contain long open reading frames which are unlikely to occur by chance.

(a) Suppose we select a random ORF from a section of noncoding ?NA in which all positions are independent and each base is equally likely.

(i) (6 points) What is the probability distribution for the length of such an ORF, where the length includes the start but not the stop codons?

(ii) (2 points) What is the probability that the length is at least 100 codons (300 bp)?

(iii) (4 points) What is the probability that the length is at least 100 codons if the base distribution is $P(A) = 0.2$, $P(C) = 0.3$, $P(G) = 0.3$, and $P(T) = 0.2$?

(b) Suppose we would like to find genes in *Saccharomyces cerevisiae* (baker's yeast) by ORF scanning. We estimate that only about 5% of yeast coding regions are less than 100 codons long based on our experience with other organisms. Therefore, we will predict any ORF of at least 100 codons as a gene. Assume for simplicity that the yeast genome has a uniform distribution of base pairs $P(A) = P(C) = P(G) = P(T) = 0.25$.

(i) (7 points) For a noncoding region of length L , show that the probability of predicting at least one false positive gene in the region is no more than

$$1 - \left[1 - \frac{1}{64} \left(\frac{61}{64} \right)^{99} \right]^{L-299}$$

Explain why this is an upper bound but not the exact probability.

(ii) (3 points) Assume for simplicity that all the *S. cerevisiae* noncoding regions longer than 300 bp have length 500 bp and that there are 2,000 such regions. Using the upper bound from part (i), and ignoring the possibility of multiple false positives in a single region, provide an estimate for the total number of false positive genes we will predict. Given that the yeast genome has about 6,000 real genes (95% of which we will correctly predict), what is the sensitivity and specificity of this approach?

Definitions:

Sensitivity = true genes predicted correctly / total true genes

Specificity = true genes predicted correctly / total genes predicted

(iii) (3 points) Now suppose we would like to try ORF scanning in the human genome. Since many human exons are short, we set our threshold at 50 codons (150 bp). Inspired by part (ii), give an upper bound on the probability that we will predict a false positive coding region in a typical human intron of length 2,000 bp. Give a similar bound for a typical human intergenic region of length 50,000 bp.