

Biostatistics & Medical Informatics / Computer Sciences 576
Introduction to Bioinformatics
Fall 2004 Midterm Exam

Name: _____

Write your answers on these pages and show your work. You may use the back sides of pages as necessary. Before starting, write your name on every page of the exam. Also, make sure your exam has every page (numbered 1 through 9).

Problem	Score	Max Score
1.	_____	16
2.	_____	16
3.	_____	16
4.	_____	16
5.	_____	12
6.	_____	12
7.	_____	12
Total	_____	100

1. Pairwise Global Alignment:

1a. (12 points) Show how the dynamic programming approach would be used to determine a global alignment for the two sequences below. Matching bases should be scored +1, mismatching bases should be scored -1, and each gap position should be penalized -2 (i.e. the gap penalty function is linear). Show the filled-in scoring matrix and the traceback pointers in the matrix.

x: **ATT**
y: **ACAT**

			A	C	A	T
	0	← -2	← -4	← -6	← -8	
A	↑ -2	↖ 1	← -1	←↖ -3	← -5	
T	↑ -4	↑ -1	↖ 0	←↖ -2	↖ -2	
T	↑ -6	↑ -3	↖↑ -2	↖ -1	↖ -1	

1b. (4 points) For these two sequences and the scoring scheme used above, show all of the optimal global alignments.

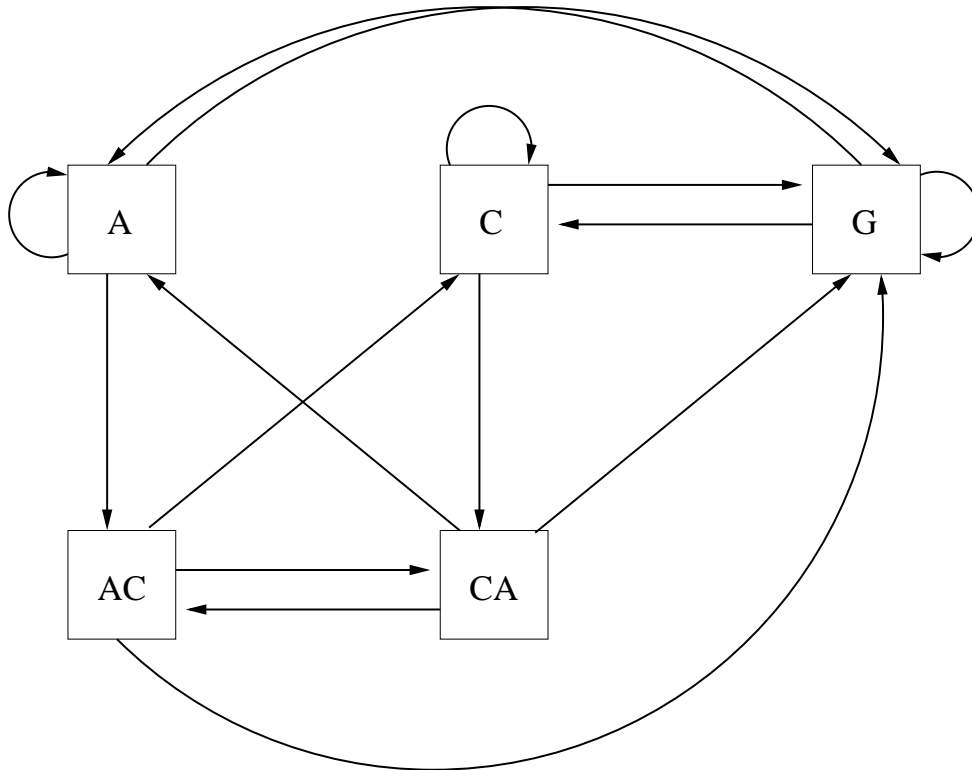
A - T T
 A C A T

A T - T
 A C A T

2. Markov Chain Models:

2a. (12 points) In a *variable-order* Markov model, not all of the states have the same order. Instead, states may represent different length histories (e.g. some states may be first-order, some may be second order, etc.). Suppose we are given the two training sequences below, and we decide to include a state in our model if we have seen the corresponding history at least *three* times in the training data. Show all of the states and the allowable transitions of the resulting Markov chain model. You do not need to include **start** or **end** states, and you do not need to estimate transition probabilities. Note that **T** has been left out of the alphabet to simplify the model.

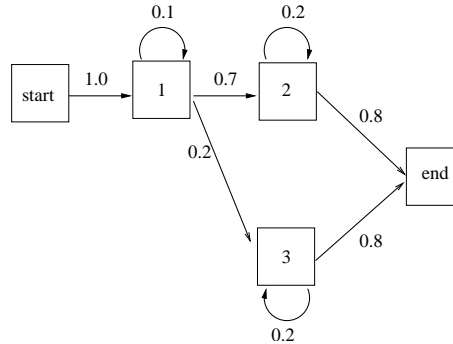
GCACACG
GACCAGG



2b. (4 points) Briefly describe a significant advantage that the variable-order approach used in question **2a.** might have over a model that uniformly used either lower-order or higher-order states. (E.g. Instead of using either a first-order or a fourth-order model, why might we prefer a variable-order model whose states ranged from first-order to fourth-order?)

The potential advantage that a variable-order model has over a lower-order model is that it can selectively remember more history. Thus it should be able to more accurately model the dependencies in a given class of sequences (assuming that we can accurately estimate all of the transition probabilities). The potential advantage that a variable-order model has over a higher-order model is that it has fewer parameters to estimate. Since it includes a longer history only when it has seen it sufficiently often in the training data, it should be able to achieve accurate estimates for all of its transition probabilities. In summary, the variable-order model addresses the key trade-off between lower-order and higher-order models.

3. Hidden Markov Models (16 points): Consider the hidden Markov Model shown below. The transition probabilities are shown in the figure, and the emission probabilities are shown in the table. The *start* and *end* states are silent. Suppose each emitting state represents a particular class of sequence, and we want to predict the class of each character in the sequence **AGTT**. Show how you would make these predictions using the appropriate dynamic programming algorithm for HMMs.



state 1	state 2	state 3
$e_1(A) = 0.5$	$e_2(A) = 0.2$	$e_3(A) = 0.1$
$e_1(C) = 0.2$	$e_2(C) = 0.2$	$e_3(C) = 0.3$
$e_1(G) = 0.1$	$e_2(G) = 0.4$	$e_3(G) = 0.2$
$e_1(T) = 0.2$	$e_2(T) = 0.2$	$e_3(T) = 0.4$

We should use the Viterbi algorithm. The underlined v values below indicate the pointers for the traceback.

$$\begin{aligned}
 v_{start}(0) &= 1 \\
 v_1(1) &= e_1(A) \times \underline{v_{start}(0)} \times a_{start,1} &&= 0.5 \times 1.0 = 0.5 \\
 v_1(2) &= e_1(G) \times \underline{v_1(1)} \times a_{1,1} &&= 0.005 \\
 v_2(2) &= e_2(G) \times \underline{v_1(1)} \times a_{1,2} &&= 0.14 \\
 v_3(2) &= e_3(G) \times \underline{v_1(1)} \times a_{1,3} &&= 0.02 \\
 v_1(3) &= e_1(T) \times \underline{v_1(2)} \times a_{1,1} &&= 0.0001 \\
 v_2(3) &= e_2(T) \times \max\{\underline{v_2(2)} \times a_{2,2}, v_1(2) \times a_{1,2}\} &&= 0.0056 \\
 v_3(3) &= e_3(T) \times \max\{v_1(2) \times a_{1,2}, \underline{v_3(2)} \times a_{2,2}\} &&= 0.0016 \\
 v_2(4) &= e_2(T) \times \max\{v_1(3) \times a_{1,2}, \underline{v_2(3)} \times a_{2,2}\} &&= 0.000224 \\
 v_3(4) &= e_3(T) \times \max\{\underline{v_1(3)} \times a_{1,3}, v_3(3) \times a_{3,3}\} &&= 0.000128 \\
 v_{end}(4) &= \max\{\underline{v_2(4)} \times a_{2,end}, v_3(4) \times a_{3,end}\} &&= 0.0001792
 \end{aligned}$$

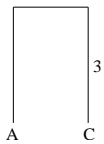
So the traceback is: $v_1(1) \leftarrow v_2(2) \leftarrow v_2(3) \leftarrow v_2(4) \leftarrow v_{end}(4)$.

We would predict that the first character belongs to the class represented by state 1 and the last three characters belong to the class represented by state 2.

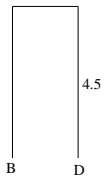
4. Distance-Based Phylogeny:

4a. (12 points) Given the following distance data for five species, show how UPGMA would produce a phylogenetic tree for these species. Show the partial tree at each step of the algorithm and indicate the distances represented by edges in the final tree.

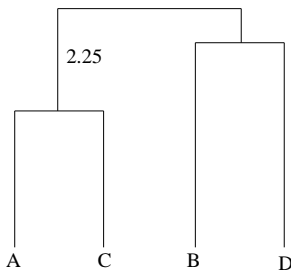
	A	B	C	D	E
A	0	11	6	10	14
B		0	11	9	15
C			0	10	14
D				0	14
E					0



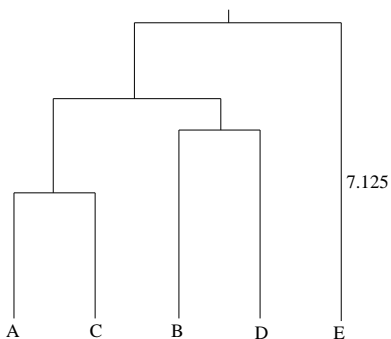
	AC	B	D	E
AC	0	11	10	14
B		0	9	15
D			0	14
E				0



	AC	BD	E
AC	0	10.5	14
BD		0	14.5
E			0



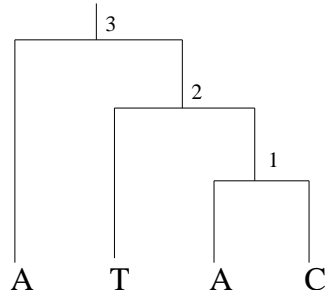
	ABCD	E
ABCD	0	14
E		0



4b. (4 points) Do you think UPGMA is the most appropriate distance-based algorithm for this data set? Briefly justify your answer.

No, the data is not ultrametric, so neighbor joining would be a better choice. We can see this using either the ultrametric test discussed in class, or just by noting that the inferred tree does not accurately represent all of the distances in the initial matrix (consider the distance from A to B).

5. Maximum Parsimony Phylogeny (12 points): Show how the weighted version of Fitch's algorithm would determine the cost of the tree below using the following symmetric cost matrix. Show your calculations and the resulting minimal cost characters for the internal nodes.



	A	C	G	T
A	0	0.8	0.3	0.9
C	0.8	0	0.7	0.5
G	0.3	0.7	0	0.1
T	0.9	0.5	0.1	0

$$R_1(A) = S(A, A) + S(A, C) = 0 + 0.8 = 0.8$$

$$R_1(C) = 0.8 + 0 = 0.8.$$

$$R_1(G) = 0.3 + 0.7 = 1.0$$

$$R_1(T) = 0.9 + 0.5 = 1.4$$

$$R_2(A) = S(A, T) + \min\{S(A, A) + R_1(A), S(A, C) + R_1(C), S(A, G) + R_1(G), S(A, T) + R_1(T)\}$$

$$= 0.9 + \min\{\underline{0.8}, 0.8 + 0.8, 0.3 + 1.0, 0.9 + 1.4\} = 1.7$$

$$R_2(C) = 0.5 + \min\{0.8 + 0.8, \underline{0.8}, 0.7 + 1.0, 0.5 + 1.4\} = 1.3$$

$$R_2(G) = 0.1 + \min\{0.3 + 0.8, 0.7 + 0.8, \underline{1.0}, 0.1 + 1.4\} = 1.1$$

$$R_2(T) = 0 + \min\{0.9 + 0.8, 0.8 + 0.8, \underline{0.1 + 1.0}, 1.4\} = 1.1$$

$$R_3(A) = 0 + \min\{1.7, 0.8 + 1.3, \underline{0.3 + 1.1}, 0.9 + 1.1\} = 1.4$$

$$R_3(C) = 0.8 + \min\{0.8 + 1.7, \underline{1.3}, 0.7 + 1.1, 0.5 + 1.1\} = 2.1$$

$$R_3(G) = 0.3 + \min\{0.3 + 1.7, 0.7 + 1.3, \underline{1.1}, 0.1 + 1.1\} = 1.4$$

$$R_3(T) = 0.9 + \min\{0.9 + 1.7, 0.5 + 1.3, 0.1 + 1.1, \underline{1.1}\} = 2.0$$

The minimal cost characters are G for node 1, G for node 2, and G or A for node 3.

6. Short Answer (12 points): Briefly define each of the following terms:

ultrametric

When the molecular clock assumption holds, the resulting distance data is ultrametric. For any triplet of sequences, the pairwise distances are either all equal or two are equal and the third is smaller.

outgroup

A taxon (species/gene) that is known to be distantly related to the rest of the taxa in a phylogenetic tree. The purpose of including the outgroup is to determine where the root of the tree should be.

two-hit method

An approach used in BLAST to determine which seeds are expanded. BLAST looks for two hits that are relatively close to one another, and on the same diagonal.

m -estimate

A method for "smoothing" probability estimates. In estimating a probability, observed counts are augmented with m virtual instances as follows:

$$Pr(a) = \frac{n_a + p_a \times m}{(\sum_i n_i) + m}$$

where n_a is the observed count for outcome a , and p_a is the prior probability of outcome a .

7. Whole Genome Sequence Alignment (12 points): Consider the task of doing pairwise alignment on whole genomes (or chromosomes). In this setting, the sequences to be aligned are millions of bases long, and thus it's not practical to simply use dynamic programming to compute the entire alignment. Briefly, describe how you might extend the BLAST algorithm to handle this task. In particular, discuss specific aspects of the BLAST algorithm you would employ for your solution.

There are several issues that you could have talked about here. I was looking for a solid discussion of at least three of them. Some of the ones I had on my list were:

- As in BLAST, you'd want to find seeds to anchor alignments.
- As in BLAST, you could then use DP between/around seeds.
- You might use the two-hit method of BLAST-2.
- Since this alignment setting would involve DNA sequences, you might want to use partial matches as seeds.
- In contrast to BLAST, you'd be doing a global alignment so you would want to retain all sufficiently good hits.