COMP 462/561 – Computational Biology Methods
Midterm examination – October 18th 2012

NAME: _____

1) Short answer questions (30 points; 5 points each)

a) What is the main difference between the structure of genes in Prokaryotes (e.g. bacteria) and in Eukaryotes (e.g. human) ?

Eukaryotic genes generally have introns, whereas prokariotic genes do not.

b) Name and describe briefly the three main biochemical steps that lead to the production of a protein from a eukaryotic gene.

1)Transcription, which transcribes the DNA of a gene into pre-messenger RNA

2)Splicing, which removes introns, to obtained the mature messenger RNA.

3)Translation, which converts the mRNA to a protein sequence.

c) In protein-coding regions of genomes, why are insertions and deletions of 3 consecutive nucleotides more frequently observed than those of 1 or 2 nucleotides?

Because indels of size that is not a multiple of 3 result in frame shifts, so that all codons downstream are affected. This means that the amino acid encoded by the sequence downstream of the indel are likely to be completely changed. Indels of length 3 affect only one or two amino acids.

d) If only 16 amino acids existed (instead of 20), what would be the minimal length of codons? Why? Watch out: that's a trick question!

3. You'd need 16 codons to encode the 16 amino acids, plus one for the stop codon. So you'd need ceiling ($\log_4$ (16+1)) = 3 nucleotides per codon.

e) Give one advantage and one disadvantage of using a Blast seed of size w=11 versus a blast seed of size w=12.

Advantage of w=11: Better sensitity.
Disadvantage of w=11: Higher running time.

f) What is the expected length of an open reading frame (ORF) in a random sequence where the nucleotides are chosen independently with probability $p_A = p_T = 1/3$, $p_C = p_G = 1/6$? Recall that the three stop codons are TAA, TAG, and TGA.

*Probability of a stop codon = 1/3*1/3*1/3 + 1/3*1/3*1/6 + 1/3*1/6*1/3 = 2/27. So the expected length of an ORF is 1 / (2/27) = 27/2 = 13.5 codons = 40.5 nucleotides.*

2) Pairwise sequence alignment (20 points)

We have studied the Needleman-Wunsch dynamic programming algorithm that finds the optimal alignment between two sequences, given a substitution cost matrix M and a gap penalty scheme. We have studied two types of gaps penalties:
(i) linear gap penalty where the cost of an insertion or deletion of size n is cost(n) = a * n,
(i) affine gap penalty where the cost of an insertion or deletion of size n is cost(n) = a * n + b.

Write a polynomial-time dynamic programming algorithm that is able to find the optimal alignment for any given *arbitrary* convex indel cost function (i.e. a gap cost function cost(n) such that cost(n+ n') <= cost(n) + cost(n') for all n,n' >= 0; an example would be cost(n) = log(n)).
Hint: The running time of the algorithm should be $O(n^3)$.

We use a variant of the NW algorithm. Consider sequences S=s1…sm, T = t1…tn. Let X(i,j) be the score of the optimal alignment between prefixes of length i and j of S and T resepectively. Then,

X(i,0) = cost(i) for all i =1…m
X(0,j) = cost(j) for all j = 1…n

For i=1…m,
For j= 1…n
X(i,j) = max {
X(I,j) + M(si,sj),
$max_{g \text{ in } \{1 \ldots i\}}$ *{ X(i – g , j) + cost(g)}*
$max_{g \text{ in } \{1 \ldots j\}}$ *{ X(i , j - g) + cost(g)}*
}

Then the algorithm proceeds as for NW…

3) Multiple sequence alignment (20 points)
The algorithm seen in class for progressive multiple alignments aims at optimizing the sum-of-pairs score. Suppose that we are instead in the following problem:

Maximum Parsimony Multiple Alignment Problem
Given: A set of sequence S1,…,Sn, and a tree T with one leaf per sequence
Find: The alignment of S1,…,Sn such that the parsimony score of the alignment on tree T is minimized.

Here, for the purpose of computing the parsimony score of a given alignment column, we consider a gap as a fifth character, other than A, C, G, and T.

Explain how to modify the progressive alignment algorithm seen in class to adapt it to the Maximum Parsimony Multiple Alignment Problem.

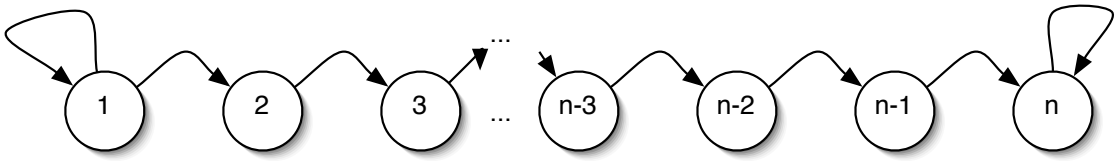
The only change that is that when computing the alignment for node u in the tree, we score the matching of one set of aligned nucleotides (coming from the left subtree of u) to another set of aligned nucleotides (coming from the right subtree of u), one use calculates the parsimony score rather than the sum-of-pairs score.

4) Hidden Markov Models (20 points)

a) (5 points) Consider a general hidden Markov model with n states. What is the worst-case running time of the Viterbi algorithm on a sequence of length L ?

$O(L\ n^2)$

b) (15 points) Now, consider the following hidden Markov model with n states. Is it possible to modify the Viterbi algorithm so that, on this particular type of linear HMM, the running time is asymptotically faster than in (a) ? Describe the modification required, and give the running time of the improved algorithm.



Use the following recurrence for the Viertbi algorithm:

*When k=1, V(k, i) = V(k,i-1) * T(1,1) * E(k, xi)*
*When k= 2… n-1, V(k, i) = V(k-1,i-1) * T(k-1,k) * E(k, xi)*
*When k= n, V(k, i) = max { V(k-1,i-1) * T(k-1,k) * E(k, xi), V(k, i-1) * T(k,k) * E(k, xi) }*

BONUS question (5 points)

Prove that the neighbor-joining algorithm will always produce the correct tree topology if given an ultrametric distance matrix on four species.

Sorry, ran out of time to write this one…