Université d'Ottawa
Faculté de génie

École d'ingénierie et de
technologie de l'information

uOttawa

L'Université canadienne
Canada's university

University of Ottawa
Faculty of Engineering

School of Information
Technology and Engineering

CSI 5126. Algorithms in Bioinformatics
Final Examination

Instructor: Marcel Turcotte

December 2009, duration: 3 hours

Identification

Student name: _____

Student number: _____ Signature: _____

Instructions

1. This is a closed book examination
2. No calculators or other aids are permitted
3. Write comments and assumptions to get partial marks
4. Beware, poor hand writing can affect grades

Question 1: Sequence alignment (15 marks)

Devise an algorithm for **counting the total number of optimal alignments** (read the sentence carefully). Give the details (recurrence equation, data-structure and pseudo-code) for an algorithm that counts the total number of optimal (global) alignments for two input sequences.

For example, the global alignment of the following two sequences, with a substitution score of -1, match score of 1, and a linear gap scoring function, where the cost for an indel is -2, produces 5 alignments, all with an optimal score of -3. Below is one of the 5 optimal alignments. The algorithm **displays a single alignment**, **as well as the total number of optimal alignments**.

```
Score = -3
Number of optimal alignments = 5
UUGGUGGUAUAGCAUAGAG-G
UGCCUGG-CGGCCUUAGCGCG
```

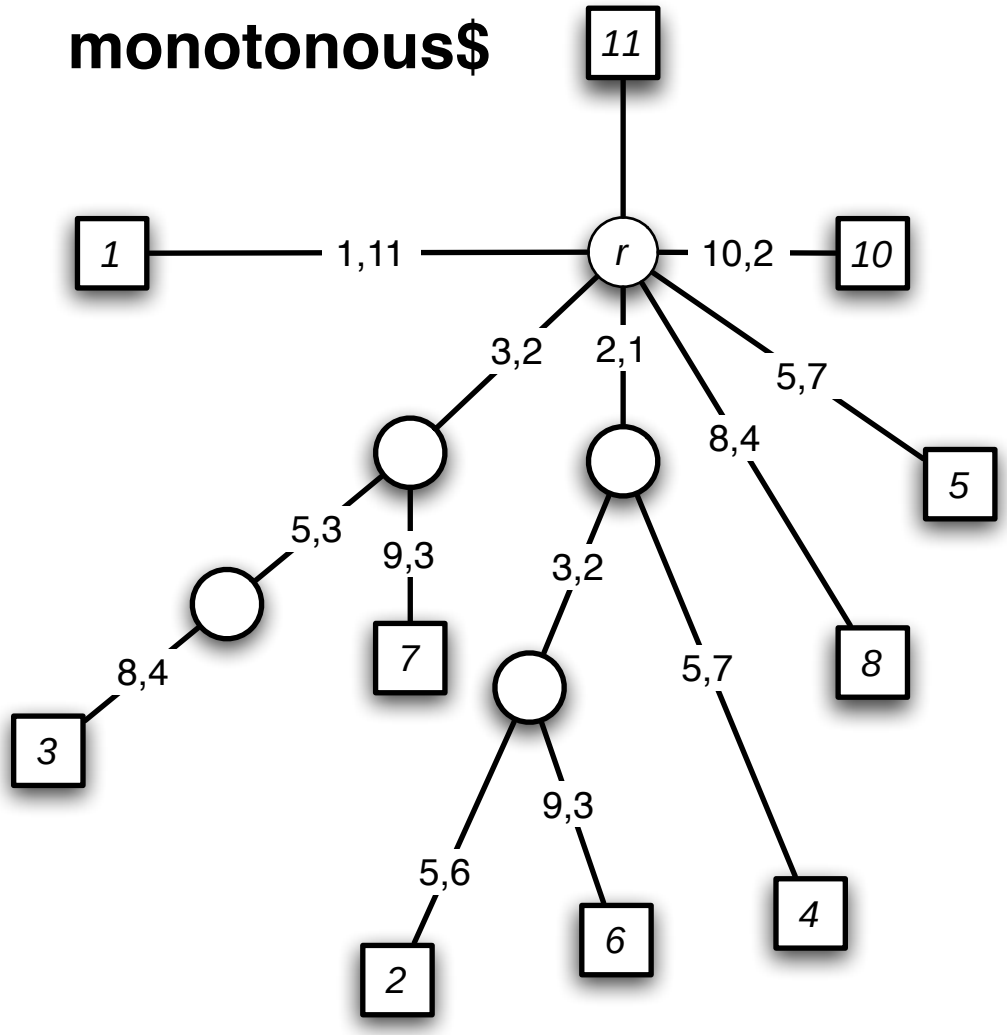
There are at least two strategies for solving this problem. The naive strategy takes exponential time to run on the worst case, and a maximum of 12 marks can be obtained for this strategy. A better solution always runs in quadratic time and space, full marks can be obtained for this solution.

Rob: In class, we started a discussion about the kind of degenerated situations that would lead to an exponential number of **optimal** alignments. Here is a follow up, if someone had the bad idea of using a scoring scheme where the cost of all the operations (match, mismatch, insertion, deletion) was the same, say 0, then all the alignments would have the same cost, 0. For the above example, there would be 260,543,813,797,441 optimal alignments.

Hint: Think about all the techniques that can be used to implement the traceback.

(Question 1: continued)

Question 2: Suffix tree: representation (15 marks)



The above suffix tree is for the string `monotonous`. The technique known as "edge label compression" has been used to represent the labels on the edges of the tree. Each pair of numbers represents a **position** in the input string and a **length**; the first position is 1. Unfortunately, 5 errors were made in constructing this suffix tree.

A. Using Roman numerals (I, II, III, IV, V), identify five (5) errors in the above suffix tree.

B. For each error, briefly explain its nature.

C. Briefly explain how to correct each error.

(Question 2: continued)

Question 3: Suffix tree: application (15 marks)

Important discoveries often originate from the observation of recurring patterns. Accordingly, pattern discovery is an important bioinformatics activity. In the case of DNA sequences, "interesting patterns" can be represented as substrings that are occurring more often than expected by chance.

Assuming the existence of a statistical test $f(s)$ that determines how likely the substring s would occur by chance, as well as a cutoff c, such that any substring s is "interesting" if $f(s) > c$.

Show how to use a suffix tree to find all interesting substrings in a database of size m. Give an outline of the method (using pseudo-code for instance). Make sure to describe the necessary initializations. Give the time and space complexity of this algorithm.

(Question 3: continued)

Question 4: Markov chains (10 marks)

A. A substitution matrix is a table that indicates the likelihood of a symbol being replaced by another one. A substitution matrix is used to assign the score of a match or mismatch in the alphabet-weighted edit distance problem. The PAM matrix is a popular substitution score used to align protein sequences. In fact, it is a family of matrices with the two most frequently used ones being PAM_{120} and PAM_{250}. For each of the following statements, indicate if the assertion is true or false.

(a) Let $m_{ij}^{(n)}$ be the entry (i, j) of a PAM_n matrix, $m_{ij}^{(n)}$ represents the probability that amino acid i is replaced by amino acid j after n units of time.

True or False

(b) PAM matrices with a large value of n are most suited to align sequences that are highly similar.

True or False

(c) PAM matrices with a large value of n are most suited to align sequences that diverged from a common ancestry long time ago.

True or False

(d) For high values of n the numbers on the diagonal are high compared to the off-diagonal terms.

True or False

B. For the sequence alignment problem, the insertion and deletion cost cannot be arbitrary values. Explain the relationship (constraint) between the indel costs and the substitution score (entries of a PAM matrix, for instance).

C. We saw in class that a finite Markov chain can be used for modeling the evolutionary substitution process. This model was used for building the PAM family of substitution matrices. It follows that the probability of observing a certain amino acid at a given position of a sequence at time $t + 1$ only depends on the current occurrence (state).

Similarly, the formalism can be used for modeling relationships between neighboring positions along a DNA sequence. In this case, the distance between the elements plays the role of time.

Let π denote the one-step transition probability:

$$\pi(a, b) = P(S(i) = b | S(i-1) = a)$$

this represents the probability of observing the nucleotide b at position i of sequence S given that the nucleotide a is found at position $i-1$ of S. Here is one-step transition probability matrix P, upper case letters represent specific nucleotides:

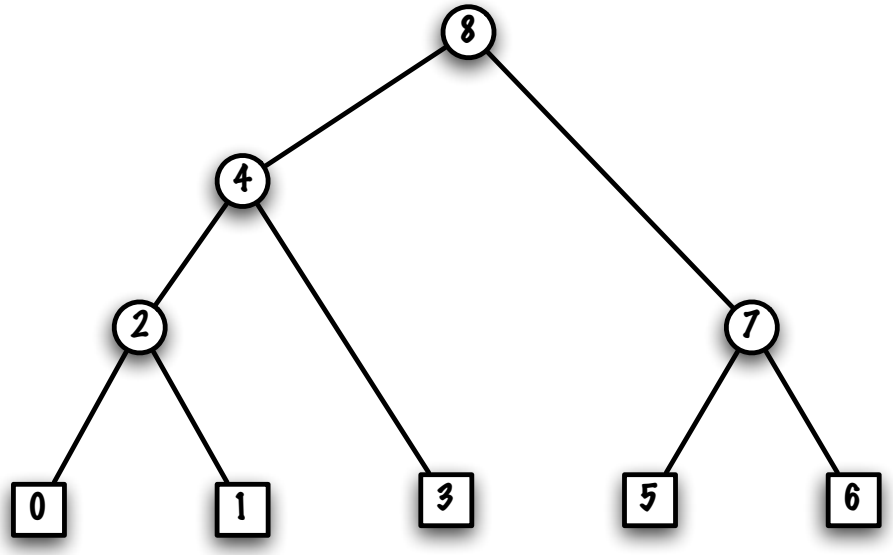
$$P = \begin{bmatrix} \pi(\mathrm{A},\mathrm{A}) & \pi(\mathrm{A},\mathrm{C}) & \pi(\mathrm{A},\mathrm{G}) & \pi(\mathrm{A},\mathrm{T}) \\ \pi(\mathrm{C},\mathrm{A}) & \pi(\mathrm{C},\mathrm{C}) & \pi(\mathrm{C},\mathrm{G}) & \pi(\mathrm{C},\mathrm{T}) \\ \pi(\mathrm{G},\mathrm{A}) & \pi(\mathrm{G},\mathrm{C}) & \pi(\mathrm{G},\mathrm{G}) & \pi(\mathrm{G},\mathrm{T}) \\ \pi(\mathrm{T},\mathrm{A}) & \pi(\mathrm{T},\mathrm{C}) & \pi(\mathrm{T},\mathrm{G}) & \pi(\mathrm{T},\mathrm{T}) \end{bmatrix}$$

Give the two-step transition probability $\pi^{(2)}(a, c)$. Express your answer using the above one-step transition probability matrix. Briefly explain your answer.

$\pi^{(2)}(a, c) = P(S(i) = \mathrm{c} | S(i-2) = \mathrm{a}) =$

Question 5: Small parsimony problem (10 marks)

The small parsimony problem is defined as finding the most parsimonious labeling of the internal vertices in a given evolutionary tree. Its input is a tree T with each leaf labelled by an m-character array. Its output consists of labels (m-character arrays) for all the internal nodes minimising the weighted parsimony score, i.e. $\Sigma_{\text{all edges } (u,v) \text{ in the tree}}\delta(u,v)$.



u	A	C	G	T
0	∞	∞	0	∞
1	0	∞	∞	∞
2	1	2	1	2
3	∞	0	∞	∞
4	2	2	2	3
5	0	∞	∞	∞
6	∞	0	∞	∞
7	1	1	2	2
8	3	3	4	5

The table s on the right was filled using the recurrence equation proposed by David Sankoff for solving the weighted small parsimony problem,

$$s_c(u) = \min_i\{s_i(v) + \delta_{i,c}\} + \min_j\{s_j(w) + \delta_{j,c}\}$$

where $s_c(u)$ is the most parsimonious score obtained when the node u is labelled with character c, here $\delta_{j,c}$ is 0 if $j = c$ and 1 otherwise. Each row of the table s corresponds to node u, and each column corresponds to a symbol c.

A. This tree models the evolutionary relationships for a single site and five species. Label all the leaves of the tree using the information from the table s. In other words, show the extant symbols for the five species.

B. Considering now the solution to the small parsimony problem, is the solution unique? If not, how many optimal labeling of the internal nodes are there?

C. Give one solution to the problem, that is to say, give one possible labeling of the internal nodes.

Question 6: RNA secondary structure (15 marks)

RNA (ribonucleic acid) secondary structure is formed by the juxtaposition in space of nucleotides that are far apart in the sequence. The nucleotides interact and form base-pairs. Context-free grammars can be used to represent RNA secondary structures.

$S \rightarrow aS_1u$
$S_1 \rightarrow aS_2u \mid uS_2a \mid gS_2c \mid cS_2g$
$S_2 \rightarrow aS_3$
$S_3 \rightarrow gS_4c \mid cS_4g$
$S_4 \rightarrow gS_5$
$S_5 \rightarrow aS_6 \mid uS_6 \mid gS_6 \mid cS_6$
$S_6 \rightarrow aS_7 \mid gS_7$
$S_7 \rightarrow a$

A. Draw the RNA secondary structure corresponding to the above grammar.

B. Give an example of an RNA sequence accepted by the above context-free grammar.

C. Assuming that nucleotides occur with the same probability, i.e. $P(A) = P(C) = P(G) = P(U) = \frac{1}{4}$.

(a) Calculate the probability that a random sequence would match the secondary structure described by the above grammar. What are your assumptions? You don't need to reduce the expression (a simple sum or product of terms suffice).

(b) Let P_G denote the probability that a sequence would match the above structure by chance (the quantity you just calculated), how many occurrences would you expect in a database of m nucleotides?

Question 7: Short answer (20 marks)

A. Genome rearrangement methods.

(a) Explain the role of permutations in genome rearrangement methods. What are the elements of a permutation representing in this context?

(b) What are signed and unsigned permutations?

(c) What is the meaning of the identity permutation?

(d) Explain the meaning of sorting a permutation in the context of genome rearrangement methods.

(e) Name and briefly describe two genome rearrangement operations.

(Question 7: continued)

B. In inferring phylogenies, either using character based approaches or genome rearrangement methods, two problems need to be solved: the inner and outer problems (a.k.a. the small and large problems). Explain these two problems, and their relationship.

C. The exact solution to the multiple sequence alignment problem requires resources that grow exponentially with respect to the number of input sequences. To circumvent this limitation, a heuristic, called progressive multiple sequence alignment, was developed. Explain this technique.

(Question 7: continued)

D. Associate **each** of the following definitions (left column) with **one** of the following terms (right column).

(a) Cell or organism lacking a membrane-bound, structurally discrete nucleus and other sub-cellular compartments.

(b) This is the process that converts a messenger RNA sequence into a chain of amino acids that form a protein.

(c) All the genetic material in the chromosomes of a particular organism needed create and maintain the organism alive.

(d) Two sequences that evolved by the process of speciation.

(e) The study of evolutionary relationships amongst organisms.

(f) The set of rules that tells the cell how to make a specific protein. Specifically, the set of rules specifies a mapping between nucleic and proteic sequences.

- **central dogma**
- **eukaryote**
- **gene**
- **genetic code**
- **genome**
- **messenger RNA**
- **orthologue**
- **paralogue**
- **phylogeny**
- **prokaryote**
- **replication**
- **transcription**
- **translation**

E. Errors occurring during translation have little or no effect on evolution. Is the statement **True** or **False**. Give a brief explanation for your answer.

F. Assumptions, and their violation.

(a) Why do sequence alignment methods assume that sequence sites (positions) are independent one of the other?

(b) Give at least two kinds of biological sequences that violate the assumption of independence.

Question 8: Bonus (5 marks)

In Assignment 2, Question 2, I asked you to write a dynamic programming algorithm **to count the total number of possible alignments for two input sequences of length** m **and** n **respectively**. Since, the algorithm has no traceback, it is possible to devise a solution that runs in quadratic time and **linear** space. Explain the necessary changes that would make the algorithm run in **linear** space.

(blank space)