Bioinformatics

Example Sheet 1

Petar Veličković

Michaelmas Term 2016

Introduction to genetics

The Bioinformatics course (as well as the field in general) will typically tend to abstract away as many underlying biological concepts as possible, and frame the issues as proper Computer Science problems. Just as a warm-up (and for you not to forget *why are we doing all of this*), I would like you to elaborate on a few of the key genetical concepts and structures that this course will explore.

1. Describe the structure of the *deoxyribonucleic acid* (DNA), and highlight the ways in which it differs from the *ribonucleic acid* (RNA). Distinguish the concepts of a *gene* and a *genome* with respect to DNA structure.

2. Explain, with the aid of a diagram, the process of *gene expression* (synthesis of a protein based on the genetic information contained within DNA). Your answer should contain the following terms:

   - DNA
   - messenger RNA
   - codon
   - amino acid
   - protein
   - transcription
   - translation

3. How are different genes delimited within the DNA molecule? Can you relate this to a similar concept used within a programming language (covered within the Tripos)?

4. How many different codons exist? How about different amino acids? Provide an *evolutionary* explanation for the discrepancy between your two answers.

Sequence alignment

You've likely already glanced at some Bioinformatics exam questions—one thing to note within them is that they first and foremost expect you to have a rock-solid knowledge of *all the relevant algorithms*. This section, as well as the following ones, will focus on reinforcing that through having you explain the key points of the algorithms and, on occasion, implementing them.

1. Describe in detail the dynamic programming algorithm used for computing a *global alignment* between two DNA sequences, noting its inputs, outputs and time complexity. Explain the significance of the *score matrix* in this context, and provide a score matrix that would convert this problem into the familiar LCS (*longest common subsequence*) problem.

2. Implement your algorithm from question 1 in a language of your choice, and use it to compute the global alignment (and alignment score) of the sequences `CGTGAA` and `GACTTAC`, with the following parameters:

   - Match: $+5$
   - Mismatch: -3
   - Insertion/deletion: -4

   You may also verify your result by computing it manually—it is good practice for the examinations.

3. Outline the key transformations that need to be made to the algorithm in order to find optimal *local alignments*, as well as incorporating *affine gap penalties*. Why are these features useful? Have you changed the time complexity of the algorithm by doing so?

4. Provide an explanation (accompanied with a brief pseudocode and diagram) of how the storage complexity of the global alignment algorithm can be significantly reduced, while keeping time complexity the same. Provide an informal proof that the asymptotic time complexity does not increase.

5. **(OPTIONAL)** Implement the reduced-storage variant of the global alignment algorithm in a language of your choice, and verify that it provides the same result for the inputs in question 2.

6. Describe the Nussinov algorithm for RNA secondary structure prediction, its underlying assumptions and time/storage complexities. What should be the output of the algorithm on the sequence `GCAACGUCG`? (this is a test of understanding—do not actually perform the algorithm on this sequence!)

Phylogeny

1. Compare and contrast the *distance-based* with *parsimony-based* approaches to constructing phylogenetic trees, and comment on their relative merits.

2. For distance-based approaches, it is often desirable for the distance matrix to be *additive*. Provide a mathematical formulation of the additivity property. [Hint: Consider any four leaf nodes (i, j, k, l) in the tree.]

3. What does it mean for a phylogenetic tree to be *ultrametric*, and what assumption does it make on the underlying evolutionary process?

4. Compare the three covered approaches to distance-based phylogeny (Additive, UPGMA and Neighbour Joining), in terms of properties and computational complexities.

5. Apply UPGMA and Neighbour Joining to the following distance matrix:

$$\begin{array}{ccccc} & A & B & C & D \\ A & 0 & 2 & 4 & 6 \\ B & & 0 & 4 & 6 \\ C & & & 0 & 6 \\ D & & & & 0 \end{array}$$

6. Implement either UPGMA or NJ in a language of your choice, and verify your findings from the previous question.

7. What is the input, output and time complexity of the dynamic programming algorithm for *small parsimony*? Illustrate the algorithm on a small example of your choice.

8. Briefly explain, with the aid of a diagram where appropriate, the greedy heuristic for *large parsimony.* Illustrate the candidate trees generated by the algorithm for the small example you used in the previous question.

Multiple alignment

1. Compare and contrast the *dynamic programming*, *greedy*, and *progressive* approaches to aligning k sequences of length n, highlighting their respective time and memory complexities.

2. Explain the inputs and steps performed by the CLUSTALW algorithm.

3. How would you assign a *"quality" score* to an obtained multiple alignment?

Approximate search

1. Present and justify the primary design choices behind the *Basic Local Alignment Search Tool* (BLAST), outline its key steps and time complexity and describe its outputs.

2. Typically, BLAST uses $w = 12$ for processing DNA sequences. Explain the potential tradeoffs involved with this decision, particularly having in consideration the following two sequences:

   Database: `GAGTACTCAACACCAACATTAGTGGGCAATGGAAAAT`
   Query: `GAATACTCAACAGCAACATCAATGGGCAGCAGAAAAT`