

Homework 3 for computer scientists
1-INF-301: Methods in bioinformatics
Deadline: Wednesday December 13, 2023 22:00

You can write your answers in English or Slovak.

(1) RNA structure and dynamic programming. In the lecture, we considered well-parenthesized RNA structures (without pseudoknots). In this task, we will consider an even more restricted set of secondary structures with *nested basepairs*, which are defined as follows. Let i, i', j, j' be four positions in the RNA sequence such that $i < j$, $i' < j'$, $i < i'$, bases at positions (i, j) are paired and (i', j') are also paired. Then in a nested basepair structure we must have $i < i' < j' < j$.

Below we see two examples of well-parenthesized secondary structures written in parenthesis notation. The first structure has nested basepairs whereas the second does not.

```
.(((.)...)) . nested  
.(.).(.) . not nested
```

- a) Design an algorithm which for a given RNA sequence x_1, \dots, x_n finds a secondary structure with the highest possible number of nested base pairs. Consider only canonical base pairs A with U and C with G. We recommend using dynamic programming. Write a recurrence and describe how it will be used to compute the answer. How can we determine the structure as a list of base pairs? What is the asymptotic running time of your algorithm?
- b) Write the rules of a stochastic context-free grammar (without listing particular values of probabilities) that can produce all RNA structures with nested base pairs (two bases of a base pair are always produced in the same production rule).
- c) Real RNA structures are stabilized by so called stacked pairs that follow each other on both sides. We will say that base pair (i, j) is supported if base pair $(i + 1, j - 1)$ is also part of the secondary structure. In the example below, we see a structure with 6 base pairs but only 4 of them are supported. The left parentheses of these supported pairs are highlighted by \sim below the structure.

```
ACGCACGCAAAGCGGCGA  
.((((.(((...)))))).  
  ~ ~ ~ ~
```

Modify your dynamic programming from part a) to find the nested structure with the highest possible number of supported base pairs. We recommend using two matrices in your dynamic programming: one for the overall highest number of supported base pairs within some substring and one where we restrict only to solutions that have the first and the last base of the substring paired.

In parts a) and c) make sure that your algorithm correctly solves the required problem and try to make it efficient, preferably $O(n^2)$.

(2) Bioinformatics tools and databases. The goal of this task is to gain some experience with bioinformatics tools and databases.

Download a short DNA sequence from <https://compbio.fmph.uniba.sk/vyuka/mbi-data/a3.fasta>. This sequence comes from an unknown bacteria contaminating a sequencing sample (real story).

- a) Use BLAST to compare this sequence with known genomes to find out from what species it probably comes from. Go to <https://blast.ncbi.nlm.nih.gov>, select *Nucleotide BLAST*, enter the provided sequence as a query, and as the database, select *Refseq Genome Database*. In the box *Organism* enter *Gammaproteobacteria (taxid: 1236)*, so the program will search only one group of bacteria (in real use, we would not have this information, and thus we would have to look for it for example in all bacteria). Leave the other settings at default values and start the search.

List details of the alignment with the highest score found by the program (% identity, E-value, source organism). What can you infer about this sequence based on the results?

- b) We would like to see if this DNA sequence encodes some proteins and what their function might be. Instead of an HMM for gene finding, we will simply consider open reading frames (ORF). An ORF is a sequence of codons that does not contain a stop codon and could therefore potentially encode some protein. We need to consider ORFs on both strands and also in all three reading frames, which are three possible shifts where codons could start within the sequence. In bacteria, genes usually do not have introns, and so the search for ORFs is a relatively successful and simple procedure for finding candidate genes.

Find the ORFs in our sequence using the tool https://www.bioinformatics.org/sms2/orf_find.html. Look for ORFs with at least 180 codons. Check all three reading frames and both strands. Allow any start codon and use the standard genetic code. List protein sequences of the found ORFs; we will use these in the subsequent subtasks. Most programs require protein sequences in the FASTA format, in which each sequence is preceded by a line starting with > sign followed by the name of the sequence, e.g. >orf1. Write the protein sequences in this format.

You should obtain 4 ORFs, which we will name as orf1 to orf4. The starts of the corresponding protein sequences should be as follows: orf1 QPKA, orf2 PEKD, orf3 TRVA, orf4 QVRN.

- c) In this subtask, we determine the structure of orf1 and orf3 proteins using AlphaFold2. If you wish, you can run AlphaFold2 yourself, using Google Colab <https://bit.ly/alphafoldcolab>. However the computation may take quite long, so you can find produced images on the course website.

In addition to the image of the structure, AlphaFold2 provides two plots representing the confidence in the structure, abbreviated pLDDT and PAE. Read about these indicators at <https://alphafold.ebi.ac.uk/faq>, sections “How confident should I be in a prediction?” and “How should I interpret the relative positions of domains?”.

Based on this, describe what you can infer about reliability of the structure predictions for the two proteins. Are some parts more reliable than others? Which of the proteins has an overall more reliable prediction? Are any domains visible on the PAE plots? Can you make any other observations?