Integrative Transcriptomics
Prof. K. Nieselt,
Institute for Bioinformatics and Medical Informatics Tübingen
Prof. S. Nahnsen,
Institute for Bioinformatics and Medical Informatics Tübingen

EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

Lecture: Grundlagen der Bioinformatik — SoSe 2023

Assignment 9 (20 points)

Hand out: Thursday, June 29 18:00
Hand in due: Thursday, July 6 18:00
Direct inquiries via the ILIAS forum or to your respective tutor at:
caroline.jachmann@uni-tuebingen.de
meret.haeusler@student.uni-tuebingen.de
simon.hackl@uni-tuebingen.de
carolin.schwitalla@qbic.uni-tuebingen.de

1. Viterbi Algorithm by hand (4P)

Given the following Hidden Markov Model with a state alphabet $Q = \{b, P, N, e\}$ and an emission alphabet $\Sigma = \{R, Y\}$. The transition and emission probability matrices, resp., are given as

	P	N	e
b	0.3	0.7	0
P	0.4	0.4	0.2
N	0.6	0.3	0.1

	b/e	R	Y
b/e	1	0	0
P	0	0.8	0.2
N	0	0.3	0.7

Your task is to compute the most probable path for the sequence RYY using the Viterbi algorithm (by hand).

We have pre-filled the matrix. E.g. $v_P(1) = e_P(R) \max_k(v_k(0)p_{kP}) = 0.8 \cdot 1 \cdot 0.3 = 0.24$. You are asked to finalize the missing entries of the matrix and provide the decoded (path) sequence via traceback.

		sequence				
	V	b	R	Y	Y	e
States	b	1	0	0	0	
	P	0	0.24			
	N	0				
		0	1	2	3	4

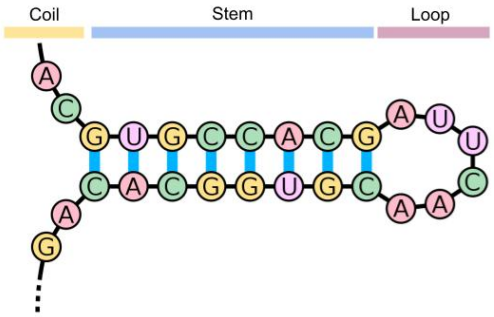
Please hand in the final Viterbi matrix and decoded path computed by traceback.

2. HMM for RNA sequence structure prediction (4P)

We will discuss in a further chapter of the lecture (Chapter 11) that RNA sequences might fold into different structures, such as the hairpin loop structure. This structure consists of the following elements: the stem which is created by two complementary sequences that are not necessarily direct neighbours in the RNA sequence, the loop which refers to the non-complementary subsequence between both paired sequences, and the coil referring to the rest of the bases. The figure[1]

[1] Adapted from: `https://en.wikipedia.org/wiki/Stem-loop`

below shows a hairpin loop and it is labeled with its three structural elements.



Your task is to define an HMM that could be used to address the classification of an RNA sequence into a hairpin loop structure with these 3 structural elements. [2]

For this, we ask you to set up a graph structure (as introduced in the lecture), showing the hidden states and symbols of the HMM and all necessary edges. You do not need to come up with transition or emission probabilities, but you should label the appropriate edges with these parameters correctly. How many states and how many symbols does your HMM have?

Practical Assignments

3. Decoding data using the Viterbi algorithm (12P)

Please implement a method that applies the Viterbi algorithm to a given sequence of symbols and returns the decoded sequence of states. For this task you can use the `HMM.java` and `Viterbi.java` files as templates. We also provide you with the already knwon `FastaReader` and `Fasta` classes in order to parse input sequences (for simplicity, they will be provided in fasta format for this task).

To solve this task:

(a) Complete the method `read` from the `HMM.java` template. The template already comprises a code skeleton to store and access the information of a HMM, i.e., states, symbols and matrices. The method should be able to parse the information from a `.hmm` file. You can get familiar with the format for the HMM model by looking at the file `cpg.hmm` in the resources directory or in the script on page 164. **Hint:** The code skeleton expects a `Reader` parameter – you may adapt the reader of the previous assignment to also read the emission states and matrix. However, feel free to change the implementation to fullfill this task.

(b) Complete the function `runViterbi` in the `Viterbi.java` template by implementing the four steps of the algorithm: *Initialization*, *recursion*, *termination* and *traceback*. You could use for each step one helper function (see template). The method should return a string of decoded states. Since the frequent application of multiplications of small numbers can lead to an arithmetic underflow in long input sequences, we ask you to find a way to solve this problem, i.e., to adjust the recursion of the Viterbi algorithm so that no underflow of the calculated probabilities occurs.

(c) Implement the function `print` in the `Viterbi.java` template that formats the given sequences to match the desired output as follows:

[2] Disclaimer: HMMs are not a common approach for the classification of an RNA sequence to be a hairpin loop. However, you might want to look at the paper by Krogh et al. (JMB, 2001) that you can find in the 'Additional Material folder' on ILIAS, which has defined an HMM for the prediction of a transmembrane protein topology.

```
Sequence: actgtgacgtgt...       (symbols of observed sequence, 60 char per line)
Viterbi:  --+++++-----...       (decoded states, 60 char per line)
```

Thus any state from the nonCPG islands (lower case letter) are represented by a '-' and any state from the CPG islands (upper case letter) are represented by a '+'. This is visually more intriguing than the original decoded path.

(d) **Apply your implementation:** Apply your program to the four sequences in the FASTA file `sequences.fasta` using the HMM model from the file `cpg.hmm`. Both files can be found in the resources directory. Export the output as a `txt`-file and include it, together with your adjusted version of the Viterbi algorithm to prevent underflow, in your hand in.