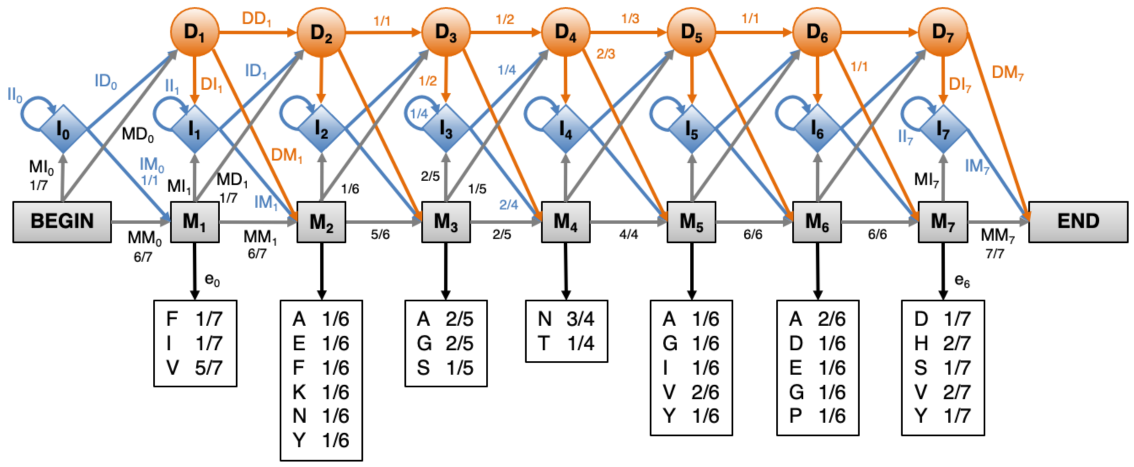
Assignment “Hidden Markov Models”

In this assignment you will implement a hidden Markov model, in particular the functionality to set its parameters (“train” it) based on a multiple sequence alignment. The algorithm is described in section 5.3 of *Biological Sequence Analysis*.

	M_1	M_2	M_3	I_3		M_4	M_5	M_6	M_7
–	V	F	A	–	A	–	A	G	Y
–	V	–	–	–	A	N	V	D	V
–	V	E	–	–	–	–	V	A	H
–	V	K	G	–	–	–	–	–	D
–	V	Y	S	–	–	T	Y	E	S
–	F	N	A	–	–	N	I	P	H
A	I	A	G	A	D	N	G	A	V

	B→1	1→2	2→3	3→4	4→5	5→6	6→7	7→E
#sequences	**7**	**7**	**6**	**5**	**4**	**6**	**6**	**7**
M→M	6/7	6/7	5/6	2/5	4/4	6/6	6/6	7/7
M→I	1/7	0	0	2/5	0	0	0	0
M→D	0	1/7	1/6	1/5	0	0	0	0
#insertions	**1**	**0**	**0**	**4**	**0**	**0**	**0**	**0**
I→M	1/1	0	0	2/4	0	0	0	0
I→I	0	0	0	1/4	0	0	0	0
I→D	0	0	0	1/4	0	0	0	0
#sequences	**0**	**0**	**1**	**2**	**3**	**1**	**1**	**0**
D→M	0	0	0	0	2/3	0	1/1	0
D→I	0	0	0	1/2	0	0	0	0
D→D	0	0	1/1	1/2	1/3	1/1	0	0



An example HMM (bottom) trained on an alignment of seven sequences (top left). There are 7 match states $M_1 \ldots M_7$*, at positions for which at least half the sequences have an amino acid. By counting the transitions in each sequence, the table on the top right can be filled. Dividing by the total number of sequences passing through a state then normalizes these counts to transition probabilities. Emission probabilities are indicated at the bottom of the HMM. Notes: only transition probabilities > 0 are shown; hints for using indices are shown only in the beginning and in the end.*

Use the code skeleton in **`assignment4_skeleton.py`** to:

- Read in aligned protein sequences from a FASTA file (reusing code from the previous assignment).

- Implement a function to calculate the match states. Here we consider a sequence position to correspond to a match state if at least half of the aligned sequences contain an amino acid at that position (see the figure above).
  Hint: return a list with the length of the alignment which indicates if an alignment position is a match state

- Implement a function that writes a reduced alignment to a fasta file. The reduced alignment should only include the match states.

- Implement a function to count the number of transitions between states and the emissions in the match states that occur in the sequences. Hints:
  - the indexes are provided in the figure
  - go through each sequence and use the list of match states calculated in point 2
  - you always start in a match state, keep track of the state that you determine, you will need it for the next transition

- Implement a function to normalize these counts to transition probabilities (next to the arrows in the figure above) and emission probabilities (in the boxes), using formulas on *Biological Sequence Analysis* p.108.

- Implement a function to sample a sequence from the HMM, i.e. generate a sequence by choosing transitions and emissions using a random number generator. A support function **`sample`**, to pick a key from a dictionary containing probability values, is supplied in the skeleton file.

The data for the HMM in the above figure is available as **`develop.fasta`** for development purposes.

Questions:

1. Read in the supplied small file **`test.fasta`** (example from *Biological Sequence Analysis*). What is the number of match states needed?

2. Generate an alignment file with only the match states. Generate a sequence logo for this alignment using the SeqLogo server (https://services.healthtech.dtu.dk/service.php?Seq2Logo-2.0). Put both the alignment and the logo in your answers.

3. Train the HMM, i.e. the emission and transition probabilities. For emissions in the insert states, use the values supplied in **`pa`**. Print the estimated transition and emission probabilities. Explain whether they make sense.

4. List at least 10 random sequences generated using your HMM.

5. Read in the supplied large file **`test_large.fasta`** and repeat steps 2-4. Report the logo (from SeqLogo), the emission probabilities of the first and last match state, the transition probabilities of the first and last transition of each type, and 10 sampled sequences.

6. Pick one randomly generated sequence and search for it in the PFAM database (http://pfam.xfam.org/). To what family does the sequence belong?

Note: If the Pfam website does not work, search in InterPro (https://www.ebi.ac.uk/interpro/) and report the hit in Pfam (ID starts with PF)

7. What is the time complexity of estimating the emission and transition probabilities?

8. Which part of your code is deterministic and which part is randomized?

Please, submit your Python script and a PDF file containing the answers to the questions on BrightSpace no later than 22:00 on the day before the lecture.