Bioinformatics written exam – 03/07/2020 - 2h

Open questions (7,5 points each, passed with 9 points over 15)

A. Describe the global alignment algorithm in details, including scoring schemes, and Dynamic programming technique. Provide both theoretical descriptions and practical examples.
B. Describe what K-mer frequency arrays are and how they can be used for mapping genomic long-reads.

Coding question (17 points overall. Passed with 9 points over 17)

Write a program that computes global alignment for a set of sequences stored in a FASTQ file. The aligner you will implement is quality-aware, so that sequences with poor average quality are discarded and not considered for the alignment procedure.

Consider what follows:

- The Python script that implements the alignment procedure receives as input two file paths: a FASTQ file storing the sequences to be aligned and a FASTA file storing the references. Additionally, it receives an integer representing the minimum quality score for a query to be considered.

  Usage: aligner.py <QUERIES PATH> <REFERENCES PATH> <QUALITY THRESHOLD>

- The average quality of a query can be approximated by the average of its nucleotide quality scores. The quality score of a nucleotide can be obtained by subtracting 33 from the ASCII representation of its quality char. For example, if a nucleotide has quality '@', then its quality

score is ASCII(‘@’) - 33 = 64 – 33 = 31. Remember that the ASCII code for a character can be computed in Python using the ord() function.

```
>>> c = ‘@’
>>> ord(c)
64
```

- Each step of your aligner must be dumped on the screen using the print() function such that the user of your script can always keep track of what is going on (For example: data have been loaded, a given query has been discarded due to low average quality, a query-reference pair has been aligned with a given score, etc...).

- You can assume that the number of queries your aligner will handle is small, so you are allowed to read all the queries and all the reference sequences at the beginning of the script.

In order to implement your aligner, you can follow the instructions below:

- Implement a function for loading sequences to be aligned and references from the input files. Assume that FASTQ files always ends with ‘.fq’ and FASTA files always ends with ‘.fa’. You can memorize headers, sequences and qualities in the data structure you prefer.

Signature: load_data(path) Max points: 4

- Implement a function that computes the approximated average quality of a sequence.

Signature: compute_quality(quality_string) Max points: 3

- Implement a function that computes the global alignment between a pair of sequences using the Needleman-Wunsch algorithm. The function is expected to return the alignment score and the alignment position in the reference sequence.

Signature: needleman_wunsch(query, reference) Max points: 7

- Put the three functions above (and any other function you implemented) together in order to implement the main of your script.

Max points: 3

Please, remember that your exam will be checked by human beings, so make sure to be tidy and use comments in your source code!