Algorithms for genomic data analysis
Assignment 1

winter semester 2023/2024, ver. 2

Task

Implement a read mapping algorithm that

- is designed to work on reads of length $\sim 1kbp$ with error rate $5-10\%$,
- may fail to map some reads, but cannot return incorrect alignments,
- is efficient (i.e. fast) and has good quality (i.e. high proportion of mapped reads).

In your program you can use:

- the codes from the classes,
- libraries included in standard Python distribution, as well as NumPy, SciPy and Biopython,
- available code for building suffix arrays or similar structures (e.g. the attached implementation of the Karkkainen-Sanders algorithm).

In your program you cannot use:

- programs and libraries to read assembly, mapping, alignment, etc.
- multiprocessing commands,
- subprograms written in other languages.

The solution should include:

- program file written in Python 3,
- slides with short description of your approach.

Specification and attached files

Minimum performance requirements are following: given the reference sequence of length $\leq 20Mbp$, a collection of r reads should be processed in at most $2 + r/10$ minutes (on students server) and memory $< 1GB$, resulting in $\geq 80\%$ reads mapped and no read mapped incorrectly.

Program should be executable using syntax:

```
python3 mapper.py reference.fasta reads.fasta output.txt
```

Input data are in fasta format. Output file should consist of one line for each mapped read, consisting of read identifier, start and end positions of read alignment, separated by tabs.

It can be assumed that all reads come from the reference sequence, but contain errors (substitutions, insertions and deletions of single nucleotides) resulting from the sequencing process. Errors occur independently at each position at the assumed error rate, so the total number of errors may slightly exceed 10% of the read length. A read is mapped correctly when the mapping coordinates differ from the actual coordinates of the fragment it comes from by $\leq 20bp$.

Attached file package contains:

- example input and output files,
- example mapping program (not good enough to meet the minimum requirements),
- Python implementation of the Karkkainen-Sanders algorithm.

Terms and conditions

The assignment can be completed individually or in 2- or 3-person teams. Schedule:

- Submit your team by email to *dojer@mimuw.edu.pl* till November 15.
- Submit your solution to moodle till December 10.
- Present your approach in class on December 12.

Assessment

Every solution that meets the minimum requirements receives 2 points and can get additional points for

- mapping quality:

  3 points $\geq 99\%$ reads mapped,
  2 points $\geq 95\%$ reads mapped,
  1 point $\geq 90\%$ reads mapped,

- mapping time:

  3 points ≤ 1 second per read,
  2 points ≤ 2 seconds per read,
  1 point ≤ 3 seconds per read,

- meeting deadlines and presentation quality: up to **2 points**,
- team size:

  2 points 1 person,
  1 point 2 persons.