Algorithms for genomic data analysis
Assignment 2

winter semester 2023/2024

Task

Design and implement an assembly algorithm, destined to work on single-end reads originating from the same strand of a single chromosome.

In your program you:

- can use the codes from the classes,
- can not use programs and libraries to read assembly, mapping, alignment, etc.
- can not use multiprocessing commands.

The solution should include:

- program file `assembly`, executable using syntax:

```
./assembly input_reads.fasta output_contigs.fasta
```

- `readme` file with short description of your approach,
- program code (if executable is binary).

Input and minimum performance requirements

Typical parameters of input dataset:

- number of reads: 1000,
- read length: $80bp$,
- average percentage of mismatches: $\leq 5\%$,
- average coverage: $\geq 5\times$.

Typical input dataset should be processed in time less then $1h$ on a common laptop, using up to $0.5GB$ memory.

Output and evaluation

The solutions will be evaluated on simulated data (i.e. artificial reads generated from a reference sequence). Output contigs will be locally aligned to the reference sequence and resulting alignments will be processed in the following way:

- ambiguous alignment fragments (sharing a reference sequence interval) will be trimmed away,
- alignments of length $< 300bp$ will be excluded.

Alignments passing filtering criteria will be scored according to the following formula:

$$S = \frac{ref\_cov \cdot cont\_cov \cdot \max(0.5, 10 \cdot (ident - 0.9))}{\log_5(4 + n\_alments)}$$

where:

- ref_cov is the proportion of the reference sequence covered by the alignments,
- $cont\_cov$ is the proportion of the contigs' sequence covered by the alignments,
- $ident$ is the identity proportion in the alignments,
- $n\_alments$ is the number of alignments.

Training data

You can download from moodle a training data package consisting of:

- directory `reference/` with a reference sequence file `reference.fasta` and `bowtie2` index files for this sequence,
- directory `reads/` with simulated read files:
  - `reads1.fasta` – 1000 reads containing ~1% mismatches,
  - `reads2.fasta` – 1000 reads containing 1-3% mismatches,
  - `reads3.fasta` – 1000 reads containing 3-5% mismatches,
- scripts to evaluate your assembly.

Scripts require `bowtie2` program and Python module `pysam`. Usage:

```
./evaluate.sh contigs.fasta
```

Two reference algorithms have been tested on the training datasets, the table below presents resulting scores:

dataset	algorithm 1	algorithm 2
`reads1.fasta`	0.06	0.59
`reads2.fasta`	0.03	0.21
`reads3.fasta`	0.03	0.08

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 December 20.
- Submit your solution to moodle till January 21.
- Present your approach in class on January 23.

Assessment

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

- assembly quality:

  6 points scores higher than reference algorithm 2,

  3 points scores higher than reference algorithm 1,
- meeting deadlines and presentation quality: up to **2 points**,
- team size:

  2 points 1 person,

  1 point 2 persons.