Bilkent University

CS481: Bioinformatics Algorithms

Homework Assignment #1

Fall 2020

INSTRUCTIONS

- Solve the following problems.
- You must write your code yourself. Sufficient evidence of plagiarism will be treated the same as for plagiarism or cheating.
- Non-compiling submissions will not be evaluated.
- Your code must be complete.
- Do not submit the program binary. You must submit the following items:
  - All of the source files
  - A script to compile the source code and produce the binary (`Makefile`).
  - A `README.txt` file that describes how the compilation process works.
- Submit your answers **ONLY** through the Moodle page.
- Zip your files and send them in only one zipped file. File name format `surname_name_hw1.zip`
- C / C++, Python, Java will be used as programming language. STL is allowed.
- All submissions must be made strictly before the stipulated deadline.
- The overall fastest implementation wins. **Bonus** will be given for the fastest code.

1) MEDIAN STRING SEARCH

Aim: In this assignment, given an integer k and a set of DNA sequences we ask to find a k-mer pattern that minimizes `Total_Distance(pattern, DNA)` over all strings of that length under Hamming distance. We call such a k-mer a median string for a set of DNA sequences. k must be equals or less than 16, `pattern` must be of a length equal or less than 500 and the number of DNA sequences must be equal or less than 100.

Input: A text file whose first line is an integer k following by a set of DNA sequences of the same length.

Output: A text file containing the k-mer `pattern` that minimizes `Total_distance(pattern, DNA)` over all k-mers. (If multiple k-mers exist, you may return any of them, one per line.)

Example

(input0.txt):

```
6
TAAGTCTATACCATCGTAGTCTAATTAACGTTATGGTAGGAT
ATCAAGGACGGAATGACCGCAGAGGCGACGTTAATGCGCCGT
CAGAGACGCCCTAAAGATTGCGGTAGGGTCCCGTTGTTAAAG
ATAAGTGTTGCTGGGTCTAAGGCATTAAAGCTGAGTCAATAG
TTAACGGACGTTTAGTGTGGATTATAGGTGAAGGGTCTGCGC
CACTCCAAGGCAGGGAACATATGTGTTGTTACTATCTTAACG
```

(output0.txt):

```
TTAACG
```

Example

(input1.txt):

```
3
GACATAATCCCTA
CGCCCATCTTCTA
CACCCGTCTCTGT
GGGTCCAGTTCAA
GTGCTCGGAGAGC
```

(output1.txt):

```
CCC
```

Example

(input2.txt):

```
9
TAGTGGTCTTTTGAGTGTAGATCTGAAGGGAAAGTATTTCCACCAGTTCGGGGTCACCCAGCG
CGCGACTCGGCGCTCACAGTTATCGCACGTTTAGACCAAAACGGAGTTGGATCCGAAACTGGA
GTTACTTGTGAGCCTGGTTAGACCCGAAATATAATTGTTGGCTGCATAGCGGAGCTGACATAC
AACATCAGGCTTTGATTAAACAATTTAAGCACGTAAATCCGAATTGACCTGATGACAATACGG
ACCACCGGATAGGCTGCTTATTAGGTCCAAAAGGTAGTATCGTAATAATGGCTCAGCCATGTC
TAGATTCGAATCGATCGTGTTTCTCCCTCTGTGGGTTAACGAGGGGTCCGACCTTGCTCGCAT
GAAATGGTTCGGTGCGATATCAGGCCGTTCTCTTAACTTGGCGGTGCAGATCCGAACGTCTCT
```

(output2.txt):

```
AGATCCGAA
```