Bilkent University

CS481: Bioinformatics Algorithms

Homework Assignment #2

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) EXACT PATTERN MATCHING

Aim: In this assignment, given two sequences T and P we ask to find whether P occurs exactly within T, and if it does, the locations of P in T. The use of `getopt` function is **compulsory** for C/C++ programs. Python programs **MUST** use `argparse` module. Java programs **MUST** use an argument parser such as `ArgParser` [1]. Your program should implement the following algorithms:

- Brute force search
- Knuth-Morris-Pratt
- Rabin-Karp

Input: Two strings T and P, where $|T| \geq |P|$. These two strings will be given in two files in FASTA format[2]. Note that FASTA file allows a single string to be represented in multiple lines.. T file must be passed using the `-i` flag and P file must be passed using the `-p` flag.

Output: For each of the four algorithms to be implemented, report:

- Whether P is in T, and if it is, the location of P within T (1-based coordinate).
- Number of character comparisons performed.
- Run time in microseconds.

Finally, report the algorithm that performed the best.

---

[1] https://www.cs.ubc.ca/~lloyd/java/doc/argparser/argparser/ArgParser.html

[2] https://blast.ncbi.nlm.nih.gov/Blast.cgi?CMD=Web&PAGE_TYPE=BlastDocs&DOC_TYPE=BlastHelp

2) EXAMPLE

file T.fa

```
>text
TAAGTCTATACCATCGTAGTCTAATTAACGTTATGGTAGGAT
ATCAAGGACGGAATGACCGCAGAGGCGACGTTAATGCGCCGT
CAGAGACGCCCTAAAGATTGCGGTAGGGTCCCGTTGTTAAAG
AGACTTGAGTGGGTGCTTGATGGGAGTGTATTAAGGGCATGT
ATAAGTGTTGCTGGGTCTAAGGCATTAAAGCTGAGTCAATAG
TTACATTGCAGATTAACGAGATCTGAAATTAAGGGAGAGATT
CCCAGAGTGGCCTAGTACTTAAGGGCACCCACGCCGCAGGCG
GCCCTACGCCCGTTAATGGTTCGAGTGCTATTCACTAACACA
TTAACGGACGTTTAGTGTGGATTATAGGTGAAGGGTCTGCGC
CACTCCAAGGCAGGGAACATATGTGTTGTTACTATCTTAACG
```

file P.fa

```
>pattern
TGGGTCTAAGGCATTAAAGCTGAGTCAATAGT
```

command

```
$> ./hw2 -i T.fa -p P.fa
Found pattern at position 180.
Performed 458234865267169716234 comparisons.
Runtime was 10ms.
Best algorithm was Knuth-Morris-Pratt
```