[CENG 315 All Sections] Algorithms

THE 8

Available from: Wednesday, January 4, 2023, 8:00 AM
Due date: Thursday, January 5, 2023, 2:00 AM
Requested files: the8.cpp (Download)
Type of work: Individual work

Can Cat match the pattern?

Last week, thanks to you, Cat was able to attend her Bioinformatics midterm. This week, she is working on her term project for the Bioinformatics class. She needs a pattern-matching program to find a specific subsequence of a given sequence so that she can identify the structures she is studying. To help her, you offer to write a pattern-matcher. Given a sequence of symbols and a pattern to search for in the sequence your program should match all overlapping and non-overlapping occurrences of the pattern, and report the starting indices of the occurrences.

There is one twist: two different symbols can be valid in each other's places. For example, given the mapping T <-> U and the pattern AGTU, we can conclude that AGTU, AGUU, AGTT, and AGUT are all valid patterns.

To be more specific, you are given a string *sequence* of length N, a *mapping* between two symbols, and a *pattern* of length M. You should scan the *sequence* and find the occurrences of the given *pattern*. You are expected to print the starting indices of each occurrence of the pattern, in increasing order.

You are expected to implement a **Finite State Automaton Matcher**. The construction of FSA can be implemented without any optimization and can even run in O(M^4) time, but, as expected, the FSA matching algorithm should run in linear time (i.e. O(N)) by making exactly one comparison for each character in the text. ***Even if you pass the test cases and time limits, solutions that do not obey these specifications will get a 0 grade.***

For this THE, as we expect you to implement the algorithmic details, you are not allowed to use any advanced algorithm libraries. For this reason, **your code is not allowed to include any additional headers/libraries** other than the ones that we already provide you.

NOTE: *Indices start from 0, as usual.*

Example IO:

1) Given **text = "AGTGTCGTGACGAGU"**, **mapping=("T","U")**, and **pattern ="GU" :**

- Valid patterns are "GU" and "GT"
- The indices of the matched patterns are 1, 3, 6, and 13. So the output should be "1 3 6 13".
- Notice that a valid match index value is the index of the text where the match begins.

2) Given **text = "AGTGTCGTGACGAGU"**, **mapping=("T","U")**, and **pattern ="TT" :**

- Valid patterns are "TT", "UT", "TU", and "UU".
- There is no match, so you should print "NONE".

3) Given **text = "ABABAT"**, **mapping=("T","B")**, and **pattern ="ABAB" :**

- Valid patterns are "ABAB", "ABAT", "ATAB", and "ATAT".
- The indices of the matched patterns are 0 and 2. So the output should be "0 2".
- Notice how the patterns overlap, but both of them are reported as separate matches.

Specifications:

- There is **1 task** to be solved in **18 hours** in this take-home exam.
- You will implement your solution in **the8.cpp** file.
- You are free to add other functions and structs to **the8.cpp**
- Do **not** change the first line of **the8.cpp**, which is **#include "the8.h"**
- Do **not** change the arguments of the function **matchPattern().**
- *<iostream>*
  <climits>
  <cmath>
  <string>
  <ctime>
  <string>
  <vector>
  <array>
  <list>
  <forward_list>
  <cmath>
  <random>
  <cstring>
  <cstdlib>
  are included in "the8.h" for your convenience.

- Do **not** include any other library and do not write *include* anywhere in your the8.cpp file (not even in comments).
- You are **not** allowed to modify the strings given to you (they are given as *const* variables). You also should **not** create separate copies to be able to modify them.
- Your solutions are evaluated both in terms of correct output and correct running time. To receive credit, your solution should pass *both* tests.
- The grade you see in the lab is **not** your final grade, your code will be reevaluated with additional inputs after the exam.
- If you are sure that your solution works in the expected complexity constraints but your evaluation fails due to limits in the lab environment, the *constant factors* may be the problem.

Constraints:

- Maximum sequence size (i.e. max N value) is **20000000** (20 millions).
- Maximum pattern length (i.e. max M value) is **1000**.
- Maximum number of matches is **1000**.
- There is always ***exactly one mapping*** that maps two symbols, and ***each symbol always consists of one letter.***
- The sequence and the patterns to be matched are strings of ***uppercase English letters.***

NOTE: To work on your local machine or see the sample test cases, you can download the *THE8Files* file shared on the course page. You can either compile your program using the Makefile by running `` `make` `` command or you can run `` `g++ -Wall -std=c++11 -o the8 main.cpp the8.cpp` ``. To run your program you should run the command `` `./the8 sample_inputs/test0x.inp` `` replacing `` `0x` `` with the test case number you want to use.

Requested files

the8.cpp

```
#include "the8.h"

//DO NOT ADD OTHER LIBRARIES


void matchPattern(const std::string& sequence, const std::pair<std::string,std::string> mapping, const std::string& pattern){


    /*****************
     *
     *
     * YOU SHOULD PRINT THE STARTING INDEX OF EACH MATCH
     * IN THE SAME LINE, SEPARATED BY A SINGLE SPACE
     *
     * IF THERE IS NO MATCH, YOU SHOULD PRINT "NONE"
     *
     * *************/

}
```

VPL