BNFO 591
INTRODUCTION TO HIGH PERFORMANCE COMPUTING IN BIOINFORMATICS AND THE LIFE SCIENCES

FORTRAN HOMEWORK 4: SEQUENCE ALIGNMENT PROGRAM

TARYNN M. WITTEN

Abstract. In this assignment you will be writing a simple program to align one sequence against another. In particular, you will be given a probe sequence to align against a larger sequence called Titin.

1. Overview

Two of the very first programs ever written for supercomputing environments were BLAST and FASTA. Your homework assignment is to write a simple alignment program as follows. Remember, we are not looking for efficient code. We just want code that works.

2. PART 1: Perfect Alignment/Single Probe

- The file `TitinFastaFormat.txt` contains a FASTA formatted sequence of a protein. Look Titin up and learn a little bit about it.
- The file `Wph_1Probe.txt` contains a FASTA formatted sequence of an endolysin. Look Wph_1 up and learn a little bit about it.
- Write a `FORTRAN` program that reads in the `TitinFastaFormat.txt` file and stores it in an array. Make sure that your program stores **only** the actual sequence and nothing else. Remember, it's in FASTA format. You may not preprocess the files. You must do all processing within the `FORTRAN` program itself.
- Next, your program should ask the user what probe the user wants to align to Titin. The user will input the name of the datafile. And your program will read that into an array as well.
- I encourage you to check and make sure you are correctly reading things into the arrays. Remember, all you know is that the files contain sequences that are in FASTA format. If you don't know what that format is, look it up.
- Once you have read in both arrays you are to carry out the following steps
  (1) Determine the frequency of each unique letter in the Titin sequence. Store that in a file called `unique_letter_frequencies.dat`. You will need that file so that you can print out a table that shows, in alphabetical order, the letter and its frequency in Titin. Hand that table in as part of this assignment.
  (2) Next, consider the correlation matrix M in which you have a two dimensional matrix that stores the pairwise adjacency of each letter with its neighbor to the right of it. For example, if the sequence was

AAAABACCABB

then you would have a matrix that is 3×3 of the form illustrated in Table [1]. Adjacency matrices are very important in network analysis and have even been used to study "omic" sequence structure in order to see how letter distributions between sequences of the same species may or may not be different.

Date: September 28, 2015.

	A	B	C
A	3	2	1
B	1	1	0
C	1	0	1

TABLE 1. In this table we illustrate a sample correlation or adjacency matrix M in which the element in the matrix represents the number of times the letter in the row is followed by the letter in the given column for the sequence AAAABACCABB

Your job is to calculate this matrix M for the Titin sequence and print it out. Obviously, you will need the results from your unique letter analysis to help you decide what the rows and columns should look like. Rows and columns need to be in alphabetical order so that I can compare results. You will hand this in as part of your assignment.

3. PART 2: EXACT ALIGNMENT

- Your next part of the program should match the probe sequence Wph_1, to the Titin sequence, beginning from the left end of Titin all the way down the line to the right end. That is, you align Wph_1 at the left end of Titin and march it down the Titin sequence until the right end of Wph_1 hits the end of the Titin sequence.
- You are to output the number of exact matches found of Wph_1 within the Titin sequence. How many did you find?

4. PART 2: PARTIAL ALIGNMENT

- Your next part of the program should match the probe sequence Wph_1, to the Titin sequence, beginning from the left end of Titin all the way down the line to the right end. That is, you align Wph_1 at the left end of Titin and march it down the Titin sequence until the right end of Wph_1 hits the end of the Titin sequence as discussed in the previous section 3. HOWEVER,
- You are going to loosen the exact match constraint to some percentage match, say 90% or 95%.
- Alter your code to ask the user to input the percentage match.
- Then rewrite your code to repeat the Section [3] exercise only now you are allowed to match to a given percentage. How does this alter your match numbers? Use 90% and 95% as your test examples. Make sure to output your results and, of course, hand them in.

NOTE: Sometimes your percentage will end up being a decimal number of sequence letters. In this case you must truncate to an integer value. For example, if I wanted 83% of the example sequence from Table[1], that would be 83% of 11 letters or a value of 9.13 letters. So you round down (always) to 9 letters have to match.

5. PART 3: TIMING

Once you have your program working, you are to make a set of timing runs without parallelization. You are to time the portion of your code that actually calculates the matches. You are to do this as follows.

- Store the execution times in a datafile named `no_opt_sequence_times.dat`.
- The file should contain three lines consisting of the match percentage and the execution time for that percentage.
- When you have completed running your timing runs and built your output datafile, use `Gnuplot` and plot, using a histogram, the execution time versus the match percentage for the three runs. Label your plot with a title, and axis labels having the correct dimensions. What do you see in your plot? Discuss the possible relationship
- Hand in a copy of your programs for this part, the datafiles, the plots and show that you executed the program on both `Compile` and `Stampede`.

6. PART 4: Bonus Points - Only After You Complete Everything Else

- Separate the different sequences in the file `EndolysinSequences.txt` into separate files in a known directory. You may use any method to do this. It doesn't have to be in `FORTRAN`.
- Create a bash script that goes to the hypothetical data directory, finds each probe file, runs it against Titin and outputs the probe name and the number of 100% 95% and 90% matches into a file. Print the file and hand it in along with everything else (of course your script too).

7. Closing Comments

Just a reminder. Make your assignment orderly. Hand in all code and data files and results.

Center for the Study of Biological Complexity, Virginia Commonwealth University, Richmond, VA 23284-2030 US

E-mail address: `tmwitten@vcu.edu`