ECS 124A Theory and Practice of Bioinformatics

Lab/Homework Assignment #2

Assigned: 1/27/2023
Due: 2/10/2023 midnight

Scope: The goal of this lab is to (a) introduce you to dynamic programming(DP) (b) modify a Perl alignment programe to incorporate user-defined match and penalty scores © learn how to use regular expressions (in Perl) (d) learn how ot compute longest common subsquences between two strings.

Exercise 1(10 points)
Download the Perl program needleman.pl from the Smartsite Resources folder.
Be sure you have it running, and try to understand how it works. This is a Perl version of the Needleman-Wunsch alignment algorithm that we studied in class, using the dynamic programming recurrence relations, instead of an alignment graph. Even though you might not learned all the Perl constructs used in this program, you should be able to understand what the different parts of the program does.

Modify the program so that it asks the user for a match value **V**, a mismatch cost **Cm**, and an indel cost **Im**. It should read the inputs from the keyboard and assigns them to variables.

Modify the program so that it finds the maximum value of any possible alignments of the two input strings, where the objective function is

maximize{V * (#matches) - Cm * (#mismatches) - Im * (#indels)}

Run your program for the two strings in Exercise 1.
If you use the same weights, do you find the same results?

Turn in the code (modification part only is fine too) and results.

Exercise 2(10 points)
Extend the needleman.pl program of exercise 2 to incorporate gaps.

Note that you will have to modify the way spaces are handled.Turn in the code (modification part only is fine too).

PART II: Perl and Regular Expressions

Exercise 3(25 points)

Please read Introduction_to_Perl_2.pdfby D. Gusfield and K. Stevens. Turn in exercises 2.1, 2.2, 2.3, 2.4, 2.5, 2.6, 2.7, 2.14 (Please number them as 4.1, 4.2, 4.3, 4.4, 4.5, 4.6, 4.7, 4.14 in your report).
(Optional) Read what lists and arrays are (from anywhere, e.g. if you google it you can find interactive tutorials in Java).

Exercise 5(15 points)

In this exercise you will work with short regular expression programs.

Run the first three programs (ex5_program1.pl, ex5_program2.pl, and ex5_program3.pl) and understand how the regular expression works.
In the starting comments of each program, you are asked to run the program in certain ways.
This is for your own advancement, you don't have to turn in any scripts or results.

5.1.

Program accession.pl is very similar to ex5_program3.pl, but has some extensions.
Modify accession.pl so that it only prints the accession numbers it finds, and prints each one on a separate line.
That is, it prints a list of accession numbers it finds in the input file, but does not print anything else.
Use ex5_testfile.txt as the test input file.

Turn in the code (part with the modifications if you don't want to turn it all)and the output.

5.2.

The meaning of [,.;:?] in the regular expression is that the digits of an accession number must be followed by any ONE of the six characters listed between the brackets [...].
So this is an OR of the six characters.

Now in the program ex5_program2.pl (it looks for a DNA string in an input line) we used **"|"** to indicate an OR. That regular expression had (A|T|C|G|a|t|c|g). Replace that with [ATCGatcg] to see if the program works.

Use **ex5_testfile.txt** as the test input file. **Turn in the result.**

PART III: Longest Common Subsequence

Exercise 6(20 points)

6.1.

By setting V to 1, and Cm and Im to zero, the program *needleman.pl*(see Exercise 2) will produce the length of the longest common subsequence (LCS) between the two sequences.
That is the alignment that simply maximizes the number of matches that can be obtained, without regard for how many spaces and mismatches are involved.

The LCS between two strings can be a proxy of the similarity for two strings. (As we discussed in class)

Create a pair of random strings of length 30 each and compute the LCS of those two strings.

What result did you get.

6.2.

In this exercise, you will determine what is the correct expected length of the LCS is. **Randomdna.pl** is a Perl program to produce random DNA strings. Get this program running and make sure you understand how it works.
Then modify it so that it asks the user how many random strings to produce, it generates that many strings and writes them into a file (Hint: the main modification is to put a loop around part of the existing program).
Put everything together into a single program that generates the random strings, computes the LCS lengths, computes and reports the averages.

(1)Now generate 100 random strings of length 10 each, and use your modified Needleman-Wunsch from Exercise 6.1 to compute the length of the LCS between the first of your strings and each of the other ones.
(2) Compute the average LCS length obtained.

Then repeat with strings of length 20, then 50, then 100, 200.
What do you observe about the average LCS length?

Turn in your results, code and your observations.

ADDITIONAL COMMENTS

A comprehensive list of alignment tools can be found at:

http://en.wikipedia.org/wiki/List_of_sequence_alignment_software

and:

http://molbiol-tools.ca/Alignments.htm

emboss is The European Molecular Biology Open Software Suite

http://emboss.sourceforge.net/

has links to online tools at:
http://emboss.sourceforge.net/interfaces/#we