

Computer Science 549 – Computational Biology

Prof. Steven Skiena

Fall 2011

Homework 2

Due Thursday, November 3, 2011

October 12, 2011

Type and print your answer, which is appreciated, or handwrite clearly and neatly. Each of the problems should be solved on a separate sheet of paper to facilitate grading. Limit the solution of each problem to one sheet of paper unless otherwise stated.

Many of these problems are deliberately open-ended and vague; do the best you can on them but don't make yourself crazy. Please don't wait until the last minute to look at the problems.

You must do this assignment in groups of 2 to 3 people.

1. Implement the Rabin-Karp randomized string matching algorithm. Conduct experiments with reasonable sized strings and patterns showing the effect of the modulus size on running time. Try extremely small values of the modulus (e.g. 2, 3, 4, 5) first to maximize the effect on running time. How small can the modulus be while still getting roughly the same running time as a very large modulus?

Write a 1 page paper with your data and experiences.

2. Build an implementation of the Smith-Waterman local similarity algorithm. Play with it to see how different insertion/deletion/substitution costs give you different alignments, perhaps including the BLOSUM80 and BLOSUM62 scoring matrices. What kind of interesting areas of local similarity do you find either DNA or plaintext strings? Attach a printout of your program and a one page description of your experiments.

A simple global alignment program written in C will be available on the course webpage.

Those who want a more macho experience can try to support affine gap costs (i.e. the cost of a gap of length is $A + Bx$ for constants A and B) and/or linear space. However, these are not required for the assignment.

3. Write a simple program to look for long open reading frames in genome sequences.

Analyze one or more (1) viral genomes, (2) bacterial genomes, and (3) human sequences (you do not need to do the full genome) looking for long open reading frames and report what you find. What is the size distribution among the long (say ≥ 100 codon) ORFs you find, and how does that compare to random sequences.

Write a 1-2 page paper on your experiences with such simple gene recognition techniques. Attach printouts of your programs.

4. A is called a non-contiguous *supersequence* of B if B is a non-contiguous subsequence of A (i.e. it can have gaps). Give a polynomial-time algorithm to find the *shortest* non-contiguous supersequence of two sequences B_1 and B_2 . Demonstrate the correctness of your algorithm and analyze its running time.
5. A palindrome is a string which is not changed when reversed, e.g. “MADAM”. We seek an algorithm which takes an input string and return a palindrome by inserting the smallest number of letters. You are allowed to insert characters at any position of the string. For example, aab can be turned into palindrome $baab$ by one insertion, and abc into $abcba$ with two insertions.

Give a polynomial-time algorithm for finding the minimum-length palindrome, with a time analysis.

6. The pizza picking problem posits a circular pizza pie of n slices, where slice i has area S_i . Eaters Alice and Bob take turns picking slices, but it is rude to create multiple gaps in the pie. Thus each eater is restricted to taking one of the two slices adjacent to the open region. Alice goes first, and both eaters seek as much pie as possible. Give a dynamic programming algorithm to determine how much pie Alice eats, if both Alice and Bob play perfectly to maximize their pizza consumption