Homework 5

BCB 548

Due Tuesday, Oct 17, 2006

1 Second best pairwise alignment, 10 points total

Describe an algorithm that finds the second best scoring pairwise sequence alignment under all alignments, for both local and global sequence alignments. Justify the running time of your algorithm.

2 Longest Common Subsequence, 10 points

The longest common subsequence (LCS) between two sequences is the longest subsequence shared by both sequences. For example, if $s = \text{friday}$ and $t = \text{daisy}$, the LCS between $s$ and $t$ is $\text{day}$. Note that this is different from the longest common substring problem, where the characters need to appear contiguously in both sequences. Show how to modify the dynamic programming pairwise sequence alignment algorithm to find the longest common subsequence.

3 Wild-card matching, 20 points total

Consider a character string with at most one wild-card which we denote "*"; this wild-card can match any sequence of zero or more characters. As an example, let’s call $p$ such a character string with $p = \text{can*not}$. If $t = \text{somecandiewhynot}$, then $p$ matches $t$ beginning at position 5 and ending at position 16. The wild-card "*" matches diewhy. For any given $p$ (with at most one asterisk) and $t$, design an efficient algorithm to find all exact matches of $p$ in $t$. You can assume that $t$ has $n$ characters and $p$ has $m$ characters. Justify the running time of your algorithm.

4 Circular DNA, 20 points total

Bacteria have circular DNA molecules. Consider a full-sequence comparison between two circular DNA molecules, which can be considered as circular strings of length $m$ and $n$. A naive way to solve this problem would be to consider all possible linearizations of each circular string ($m$ and $n$ linearizations respectively) and solve the full-sequence alignment for each pair. However, this would entail $mn$ alignments, each taking $O(nm)$ time, for a total running time of $O(m^2n^2)$. Find a more efficient algorithm and justify it’s running time.
5 Middle matching, 20 points total

Imagine that you are doing a pairwise sequence alignment between two sequences, $t$ and $s$. There is a subsection of both sequences, say $s_i, ..., s_j$ and $t_l, ..., t_p$, that you want be aligned within the global alignment. You can assume that $s_i$ will be aligned with $t_l$ and that $s_j$ will be aligned with $t_p$. Design an algorithm that will calculate the best pairwise alignment between $s$ and $t$, given the previous restriction and an arbitrary scoring function.

6 Linear space alignment, 20 points

The algorithm for performing a pairwise sequence alignment using linear space involves splitting the problem in half and solving each half recursively. What if, instead of splitting it in half, you split it into portions of $1/3$ and $2/3$. How would this affect the space and running time of the alignment algorithm? Justify your answer.