CS 5633 Analysis of Algorithms – Spring 10

2/25/10

6. Homework

Due **3/4/10** before class

1. **Range tree counting queries (4 points)**
   Show how to augment a 1-dimensional range tree of n elements such that range **counting** queries can be answered in $O(\log n)$ time. Argue that your augmentation does not change the asympotic preprocessing time and the asymptotic space complexity.

2. **Saving space (6 points)**

   a) (2 points) The bottom-up dynamic programming algorithm computing the n-th Fibonacci number $F(n)$ takes $O(n)$ time and uses $O(n)$ space. Show how to modify the algorithm to use only constant space.

   b) (4 points) Suppose we only want to compute the *length* of an LCS of two strings of length m and n. This means we do not need to store the whole dynamic programming table for a later traceback.

   Show how to alter the dynamic programming algorithm such that it only needs $\min(m, n) + O(1)$ space. (Notice that it is *not* $O(\min(m, n))$, but plain $\min(m, n)$.)

3. **Traceback (8 points)**

   a) (4 points) Show how to perform the traceback in order to construct a longest common subsequence from the DP table *without* using the auxiliary arrow-table. Make your algorithm as efficient as possible. What is the runtime of this traceback algorithm?

   b) (4 points) Show how to perform the traceback in order to construct an optimal parenthesization for the matrix chain multiplication problem *without* using the auxiliary s-table. How much time does this traceback algorithm need? Justify your answer.

4. **Intervals (8 points)**
   Let $A[1..n]$ be an array of n integers (which can be positive, negative, or zero). An *interval* with start-point i and end-point j, $i \leq j$, consists of the numbers $A[i], \ldots, A[j]$ and the *weight* of this interval is the sum of all elements $A[i] + \ldots + A[j]$.

   The problem is: Find the interval in A with maximum weight.

   (a) **(2 points)** Describe an algorithm for this problem that is based on the following idea: Try out all combinations of i, j with $1 \leq i < j \leq n$. What is the runtime of this algorithm?

   (b) Describe a dynamic programming algorithm for this problem. Proceed in the following steps:

      i. **(2 points)** Develop a recurrence for the following entity: $S(j)$ = maximum of the weights of all intervals with end-point j.

      ii. **(1 point)** Based on this recurrence describe an algorithm that computes all $S(j)$ in a dynamic programming fashion, and afterwards determines the end-point j^* of an optimal interval.

      iii. **(2 points)** Given the end-point j^* find the start-point i^* of an optimal interval by backtracking.

      iv. **(1 point)** What are the runtime and the space complexity of this algorithm?