CSE548/AMS542 Spring 2008 Analysis of Algorithms

Jie Gao*

March 13, 2008

Due **March 27th** before class. Each problem, unless specified otherwise, has a maximum of 10 points. (i) You write down the solution clearly. If we can not recognize your writing then you may lose points. (ii) Avoid too many details. A succinct and clean proof is the best. You may use the algorithms we covered in class without referring to the details. (iii) If you discuss some of the problems with another fellow student (at most 2 students per group), write down his/her name and the problems. If you consult any books/webpages, cite them.

Homework 4

1. A string C is a super-sequence of A if and only if A is a subsequence of C. Give an efficient algorithm for finding the shortest common super-sequence of two strings A and B.

2. Given an array of integer (possibly negative) numbers $A[1..n]$, we define the weight of a subarray $A[i..j]$ for $1 \le i \le j \le n$ to be the sum of the elements contained in this subarray, $weight(i, j) = \sum_{k=i}^{j} A[k]$. The absolute weight is the absolute value of the weight, $|weight(i, j)|$. Notice that the numbers in the array are allowed to be negative, but the absolute weight is always non-negative.

   (a) Design and analyze an algorithm that finds a subarray with the minimum absolute weight with running time $o(n^2)$. (10pts)

   (b) **Extra credit.** Design and analyze an algorithm that finds a subarray with the minimum weight with running time $O(n)$. (10pts)

3. Given a tree T with integer weights on the edges. The weights may be negative, zero, or positive. Give a $O(n)$ time algorithm to find the shortest simple path in T. The length of a path is the sum of the weights of the edges in the path. A path is simple if no vertex is repeated. (10pts)

4. Consider the following variant to the Towers of Hanoi puzzle. Assume that the pegs are placed at the vertices A, B, and C of an equilateral triangle. Further assume that we add the following restrictions to the rules: Discs my only be moved in clockwise order. So, any disc moved from peg A must go to peg B, any disc moved from peg B must go to peg C, and any disc moved from peg C must go to peg A. Assume all discs start on peg A. It is still the case that you only can move one disc at a time, and that no larger disc can go on top of a smaller disc. The goal is to get all the discs to peg B using the fewest possible number of disc moves. Give an algorithm for this problem. (10pts)

*Department of Computer Science, Stony Brook University, Stony Brook, NY 11794, USA, `jgao@cs.sunysb.edu`.

5. Given k lists of sorted arrays, each list has n elements.
   (a) Design an algorithm that merges the k lists to a sorted list of kn elements with running time $o(nk^2)$. (5pts)
   (b) Give a lower bound that any algorithm based on comparisons will need to take. (5pts)

6. Given a sequence of n points $p_1, \cdots, p_n$ in the Euclidean plane. You are to find the shortest routes for two taxis to service these requests in order. Let us be more specific. The two taxis start at the origin. If a taxi visits a point p_i before p_j then it must be the case that $i < j$. Each point must be visited by at least one of the two taxis. The cost of a routing is just the total distance traveled by the first taxi plus the total distance traveled by the second taxi. Design an efficient algorithm to find the minimum cost routing. (10pts)

7. The input to this problem consists of an ordered list of n words. The length of the ith word is w_i, that is the ith word takes up w_i spaces. (For simplicity assume that there are no spaces between words.) The goal is to break this ordered list of words into lines, this is called a layout. Note that you can not reorder the words. The length of a line is the sum of the lengths of the words on that line. The ideal line length is L. No line may be longer than L, although it may be shorter. The penalty for having a line of length K is $L - K$. There are two ways to define the total penalty as shown below. The total penalty is the maximum of the line penalties. The problem is to find a layout that minimizes the total penalty in polynomial time. (10pts)

8. **Extra credit.** An x-monotone polygonal chain is a piecewise linear polygonal chain that has at most 1 intersection with any vertical line. Given two x-monotone polygonal chains P and Q with a total of n vertices between then. Find their intersections in $O(n)$ time. (10pts)