Ph.D. Comprehensive Exam: Math and Algorithms

Spring 2001

The exam includes eight problems;
the length of the exam is three hours.

Problem 1

Prove or disprove the following asymptotic bound:

$\sqrt[3]{n^n} = o(\sqrt{n!})$.

Problem 2

(a) Determine the time complexity (Θ-notation) of the ADDER algorithm, given below. Assume that the "+" operation takes constant time, regardless of the value of n.

(b) Give an efficient nonrecursive algorithm that computes the same value as ADDER.

ADDER(n)
if $n \leq 2001$
    then return 1
$sum \leftarrow 0$
for $i \leftarrow 1$ **to** $n - 1$
    do $sum \leftarrow sum +$ ADDER($n - 1$)
return sum

Problem 3

Consider a binary search tree, and suppose that we need to print all nodes whose keys are between two given values. For example, if the given values are 3 and 5, and the tree includes the keys 1, 2, 3, 4, 5, 6, and 7, then we should print 3, 4, and 5. Write a fast algorithm for this problem; it should be more efficient than standard *inorder tree walk,* that is, it should not traverse the whole tree.

Problem 4

We consider a *directed unweighted graph,* and denote the number of its vertices by V and the number of its edges by E. Describe a data structure for representing the graph that supports all operations listed below. Note that we cannot use the standard adjacency lists or adjacency matrix, since they do not satisfy the given time requirements.

- Check the presence of an edge between two given vertices, in $O(\lg V)$ time
- Add the edge between two given vertices, in $O(\lg V)$ time
- Invert all edges, that is, replace every edge with the opposite, in $O(V + E)$ time
- Perform the breadth-first search, starting from a given vertex, in $O(V + E)$ time

Problem 5

Consider a set of n distinct points in the plane, represented by Cartesian coordinates. We say that (x_1, y_1) dominates (x_2, y_2) if $x_1 \geq x_2$ and $y_1 \geq y_2$; for example, $(2, 3)$ dominates $(1, 2)$. Write an efficient algorithm that inputs a list of n points and prints each point that does *not* dominate any other point in the list. For instance, if the input is "(1,2), (2,3), (0,3)," then the algorithm will print (1,2) and (0,3). Its running time should be $O(n \cdot \lg n)$.

Problem 6

Give an efficient algorithm for finding the least common multiple of two given natural numbers; that is, it should output the smallest natural number that is divisible by both input numbers. For example, if the input numbers are 6 and 8, the algorithm should output 24.

Problem 7

The classical Hamiltonian Cycle Problem requires to input an undirected graph and find a simple cycle through all vertices of the graph; that is, the cycle should include all vertices of the graph and have no self-intersections. We consider a modified version of this problem, which requires to find a *maximal-weight* simple cycle in a *weighted undirected graph;* note that this maximal-weight cycle may not include all vertices. The classical Hamiltonian Cycle Problem is known to be NP-complete. Use this fact to prove that the modified version is also NP-complete.

Problem 8

Consider the problem of finding the *maximal-weight* path between two given vertices of a *weighted directed acyclic* graph. Determine whether it is NP-hard and justify your answer.