

### PhD Qualifying Exam, Algorithms: Spring 2001

Choose three problems from the first *easy* group, and two from the *hard* group. **You should answer FIVE questions total.**

#### Easy

1. Give a definition and draw a graph to illustrate each of the following.
  - $O(f(n))$
  - $\Theta(f(n))$
  - $\Omega(f(n))$
2. Give Big-O values for Quicksort, Mergesort, Heapsort, and Insertion Sort. Be careful with Quicksort—you might want to mention average and worst case values for this.
3. Why would we prefer using a binary search tree to a linked list?
4. We've got a road map with estimated driving times and the total number of miles for each segment of road. What algorithm would you use to determine the way to get from one city to another by driving the least number of miles? What algorithm would you use to minimize total driving time?
5. We have five different blocks (all the same size, but different colors). How many different ways can we stack them into a tower 5 blocks tall?
6. Hash tables are used in many programs to improve computational complexity. Describe an application which commonly utilizes a hash table, and explain how the hash table improves performance.
7. Give an example of a problem for which dynamic programming is appropriate, and sketch pseudocode for the problem.
8. We're in Las Vegas, using a slot machine. It has three wheels, each with 8 different values (Elvis, Horseshoe, 7, Dollar Sign, Hearts, Clubs, Spades, Diamonds). The machine takes one quarter, and pays out one quarter if two of the values match, and one dollar if all three match. What are the odds that we'll get a quarter back? What are the odds that we'll get a dollar back?

#### Hard

1. Let  $P_1, P_2, \dots, P_n$  be  $n$  programs to be stored on a single disk. Program  $P_i$  requires  $s_i$  kilobytes of storage, and the capacity of the disk is  $D$  kilobytes, where  $D < \sum_{i=1}^n s_i$ . We want to maximize the number of programs held on the disk; give an optimal greedy algorithm to do this, and prove that the algorithm is correct.
2. Arbitrage is the use of discrepancies in currency exchange rates to transform one unit of currency into more than one unit of the same currency. For example, suppose that 1 US dollar buys 0.7 British pound, 1 British pound buys 9.5 French francs, and 1 French franc buys 0.16 US dollars. Then, by converting currencies, a trader can start with 1 US dollar and buy  $0.7 \times 9.5 \times 0.16 = 1.064$  US dollars, thus turning a profit of 6.4 percent.

Suppose that we are given  $n$  currencies  $c_1, c_2, \dots, c_n$ , and an  $n \times n$  table  $R$  of exchange rates, such that one unit of currency  $c_i$  buys  $R[i, j]$  units of currency  $c_j$ .

Give an efficient algorithm to determine whether or not there exists a sequence of currencies  $c_{i_1}, c_{i_2}, \dots, c_{i_k}$  such that the sequence allows an increase in value.

3. Suppose we have  $n$  people, and some of these people dislike each other. We need to gather them at a round dinner table, and cannot place two people who dislike each other in adjacent seats. Describe an algorithm that will find a seating plan, or will state that no such plan is possible. Prove that this problem is *NP* complete.