

PhD Exam – Spring 2003 – Algorithms
Answer five of the six questions.

1. Give formal definitions for O , Ω , Θ , P , and NP .
2. The matrix-rounding problem is: Given an $m+1$ by $n+1$ matrix A of real numbers. Assume that each element in column $n+1$ contains the sum of the rest of the elements in the row. Assume also that each element in row $m+1$ contains the sum of the rest of the elements in the column. The problem is to round all the numbers so that the row and column sums are maintained. The first application of this problem was encountered when US census data had to be published.

5.231	6.222	8.345	19.798
7.789	3.234	7.567	18.59
8.135	4.984	8.555	21.674
21.155	14.44	24.467	

Note that simply applying a rounding rule on each number does not solve the problem. For example $.65+.65+.65=1.95$. If each number is rounded using the rule that a fraction of $.5$ or above is rounded to 1 , we would get $1+1+1=2$. If we rounded the numbers by discarding the fraction we would get $0+0+0=1$.

Transform the problem into an integer programming problem.

3. The LARGEST BOUNDED-WEIGHT SUBSET problem is: given a set of n distinct numbers x_1, x_2, \dots, x_n and a number B find a maximal cardinality subset S of T for which the sum of the numbers in S is at most B . Give an $O(n \lg n)$ algorithm for this problem. There is an $O(n)$ algorithm for this problem. What is it?
4. Define the optimization problem LONGEST-PATH-LENGTH as the relation that associates each instance of an undirected graph and two vertices with the number of edges in the longest simple path between the two vertices. Define the decision problem $\text{LONGEST-PATH} = \{(G, u, v, k) : G = (V, E) \text{ is an undirected graph, } u, v \in V, k \geq 0 \text{ is an integer, and there exists a simple path from } u \text{ to } v \text{ in } G \text{ consisting of at least } k \text{ edges}\}$. Show that the optimization problem LONGEST-PATH-LENGTH can be solved in polynomial time if and only if $\text{LONGEST-PATH} \in P$.
5. Suppose you have n unique integers; you could place them into a binary search tree in many different ways, making many different trees. Describe a recursive formulation to calculate the number of possible different trees, and give dynamic programming pseudocode to calculate that number.
6. Discuss the differences between the Ford-Fulkerson method and the Edmonds-Karp algorithm. What are they used for? Why would we prefer one to the other?