Design and Analysis of Algorithms

Final Examination (Volume A)

January 06, 2012

Notice: This exam is closed book, no books, papers or recording devices permitted. You may use theorems from class, or the book provided you can recall them correctly. Add some annotation to your algorithm and pseudo code when essential.

Name________________No.______________Score________________

No.	1	2	3	4	5	6	7	8	Total
Percent									
Score									

Problem 1. Fundamentals (2*10=20 points)

1. $T(n)=3T(n/3)+\log_2 n$, the tight asymptotic bounds for the recurrence is___n_____.
2. You are given a sequence of n distinct numbers $<a_1, a_2, \ldots a_i, \ldots a_n>$. We call it an inversion when $a_i>a_j$ and $i<j$. An efficient algorithm to count the number of inversions in the sequence could be accomplished in _nLogn____.
3. At most ________ comparisons are sufficient to find both the minimum and the maximum of a set of n elements.
4. Use quicksort algorithm to select the first element as a pivot element around which to partition the array (46, 79, 56, 38, 40, 84), the result of first partition is____________________. 3 5 6 1 24
5. To an unsorted array with size n, the asymptotic time of building a heap is _O(n)__and the asymptotic time of building a red-black tree is O(n lgn).
6. While Class P problems indicate problems could be solved efficiently, Class NP problems mean____.
7. Consider Maximum Bipartite matching in G. Let $n=|X|=|Y|$, and let m be the number of edges of G. Assume that there is at least one edge incident to each node in the problem, the time to compute a maximum matching is ___________.
8. List all topological orderings of the DAG in Figure1.
9. Draw the tree after deleting a node z from the binary search tree in Figure 2
10. A sequence of stack operations (POP, PUSH, COPY) is performed on a stack whose size never exceeds k. After every k operations, a copy of the entire stack is made for back up purposes. By assigning suitable amortized cost to the various stack operations, the cost of n stack operations (including COPY) is _________.

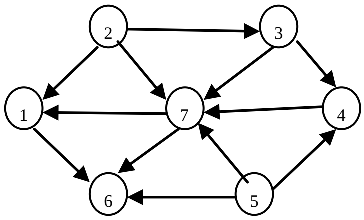


Figure 1

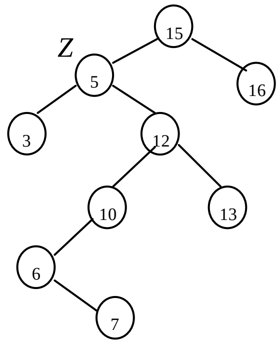
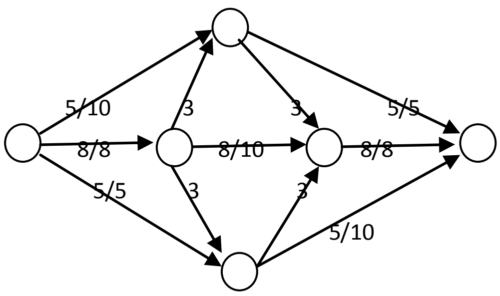


Figure 2

Problem 2 Computation and Justification (True or false: if True, give a short explanation; if False, give a counter example) (20 points)

1) Suppose we are given an instance of the Minimum Spanning tree problem on a graph G, with edge costs that are all distinct. Let T be a minimum spanning tree for this instance. Now suppose we replace each edge cost C_e by its square C_e^2, thereby creating a new instance of the problem with the same graph but different costs. T must still be a minimum spanning tree for this new instance. True or false? (4 points)

2) Maximum Flow (8 points)
   a) Following figure shows a flow network on which an s-t flow has been computed. What is the value of this flow? Is this a maximum (s, t) flow in the graph.



   b) Find a minimum s-t cut in the flow network and also says what is its capacity is.

3) Frequencies of 8 characters ***a-h*** are happened to be the first 8 Fibonacci numbers (F1, F2,…F8). What are the Huffman codes of these characters? (4 points)

4) Describe the Relationship between P, NP and NPC problems. Justify why NPC problem is the core of argument that whether P=NP. (4 points)

Problem 3. (10 points)

The input consists of two arrays A[1…*n*] and B[1…*n*+1] containing positive integers, all distinct and both in sorted order. Give a fast algorithm to find the median of all the 2*n*+1 numbers i.e. a value A[*i*] or B[*j*] such that exactly half the numbers are less than this value and half greater. For example, if A = [3, 12, 14, 44] and B = [5, 17, 28, 31, 40], then the median is 17 = B[2]. For full credit, your algorithm must run in time O(log *n*).

Problem 4. (12 points)

1. We have introduced both Greedy Algorithms and Dynamic Programming during the course. Could you describe the major differences of the manner in which they solve the optimization problem? (3 points)
2. Prove that the fractional knapsack problem has the greedy-choice property and optimal substructure. (3 points)
3. Give a dynamic-programming solution to the 0-1 knapsack problem that runs in $O(nW)$ time, where n is number of items and W is the maximum weight of items that the thief can put in his knapsack. (6 points)

Problem 5. (14points)

Suppose you are managing the construction of billboards on the Yan'an Highway, a heavily traveled stretch of road that runs west-east for M miles. The possible sites for billboard are given numbers $x_1, x_2, \ldots x_n$ each in the interval [0, M] (specifying their position along the highway, measured in miles from its western end). If you place a billboard at location x_i, you receive a revenue of $r_i > 0$.

Regulations imposed by the country's Highway Department require that no two of the billboard be within less than or equal to 5 miles of each other. You would like to place billboards at a subset of the sites so as to maximize your total revenue, subject to this restriction. Example: Suppose M=20, n=4 and

$$\{x_1, x_2, x_3, x_4\} = \{6, 7, 12, 14\}, \{r_1, r_2, r_3, r_4\} = \{5, 6, 5, 1\}$$

Then the optimal solution would be to place billboards at x_1 and x_3, for a total revenue of 10. Give an algorithm that takes an instance of this problem as input and returns the maximum total revenue that can be obtained from any valid subset of sites.

a) Argue that this problem exhibits optimal substructure.
b) Define recursively the value of an optimal solution.
c) Present an efficient algorithm to compute the revenue of an optimal solution and analyze your algorithm

Problem 6. (14 points)

Consider a network of computers represented by a graph $G = (V, E)$: the vertices are computers and an edge represents a communication link between the two endpoints. Each edge e has a number c_e associated with it which is the maximum rate of data transmission it can support. You need to send data form your computer to your friend's computer at the maximum possible rate. If P is a path in G between the vertex u representing your computer and the vertex v representing your friend's, then the maximum rate of sending data along P is determined by the minimum rate c_e of an edge on the path P: this is called the bottleneck rate of the path P. Thus you want to find a path P between u and v with maximum bottleneck rate.

1. Give a greedy algorithm to solve the problem.
2. Argue that your algorithm is correct and optimal.
3. What is the running time of your algorithm? Justify with the appropriate data structures.

Problem 7. (10 points)

Let $G = (V, E)$ be a bipartite graph, that is, a graph in which the vertex set V can be partitioned into subsets L and R in such a way that all edges are between a vertex in L and a vertex in R. Suppose that every vertex has exactly five incident edges. It can be shown that this implies $|L| = |R|$. Prove that G admits a maximum matching of size $|L|$.