DEPARTMENT OF INFORMATION TECHNOLOGY, NITK SURATHKAL

END SEMESTER EXAMINATION, APRIL 2018

IT252: DESIGN AND ANALYSIS OF ALGORITHMS

Class: IV SEM B.TECH (IT)
Date: 20/04/2018

Time: 3 Hrs.
Marks: 100

Register No. | 16IT202

NOTE:
1. There are six questions in this paper.
2. Each question has multiple parts. Read the entire question carefully.
3. Use Pseudo-code to describe algorithms, unless asked otherwise.

Problem 1 [10 x 2 = 20 marks]

State if the following statements are True or False. Give clear justifications for your answer. If a statement is True give an argument to prove this. If it is False, give a counter-example if relevant.

a) Consider a stable matching instance of n men and women, where there is a man m who is last on every woman's preference list, and there is a woman w who is last on every man's preference list. Then in all possible stable matchings of this instance, m and w will be paired.

b) The FFT algorithm on a sequence of N points does $(N/2)\ log_2N$ complex multiplications and $N\ log_2N$ complex additions.

c) To find the longest path between two vertices s and t in a graph G, we can simply find the shortest s-t path in a graph G', where G' has the exact same vertices and edges as G, but the edge weights are negated. That is, if an edge e has weight w_e in G, then its weight in G' is $-w_e$.

d) Any function $f(n)$ that is $2^{O(n)}$ is also $O(2^n)$.

e) If an undirected connected graph G has a unique minimum spanning tree, then all the edge weights of G are distinct.

f) Given n numbers $a_1, \ldots, a_n$ (where $n > 20$), the median of the smallest ten numbers and the largest ten numbers among them can be computed in $O(n)$ time.

g) Let G be a graph with a negative weight cycle. Then the shortest path between all pairs of vertices is undefined.

h) Consider a Disjoint-Set data structure that implements both *Union-by-rank* and *Path-compression* heuristics. Then at any point in time, all the trees in the Disjoint-Set forest will have height one. In other words, the *parent* pointer of all the non-root nodes of a tree is the root (the representative element).

i) For any instance of the Stable Matching Problem, if there exists a perfect matching that is not stable due to m and w wanting to be together, then (m, w) will be in every stable matching of that instance. In other words, if (m, w_0) and (m_0, w) are both in a perfect matching and m prefers w over w_0, and w prefers m over m_0, then (m, w) will be in every stable matching.

j) Let X be an NP-Complete problem. If we can prove that X cannot be solved deterministically in polynomial time, then $P \neq NP$.

Problem 2 [16 marks]

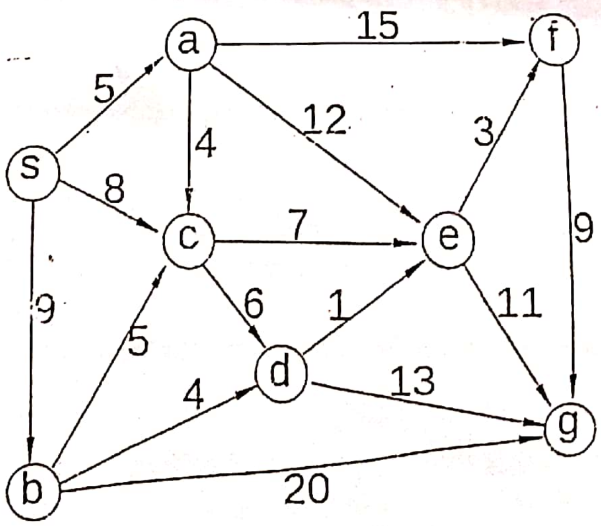
Given two arrays A and B each with n numbers $a_1, \ldots, a_n$ and $b_1, \ldots, b_n$ respectively, we want to rearrange them in the order $a_1, b_1, a_2, b_2, \ldots, a_n, b_n$. E.g. if $n = 4$ and $A = (1, 3, 5, 7)$ and $B = (2, 4, 6, 8)$, then the final order is $(1, 2, 3, 4)$ $(5, 6, 7, 8)$. This output must be stored in the same arrays as the input, i.e. in A and B. You can think of the input as an array of size $2n$, the first n elements belonging to A and the next n elements that of B, and we want to rearrange these values as specified. You are however only allowed $O(logn)$ amount of *temporary space*. Temporary space means the total amount of space among all data structures and temporary variables except the arrays A and B. In particular, the output has to be written in A and B (not printed to the terminal for e.g.).

Give an algorithm to do the above rearrangement, that runs in $O(nlogn)$ time and uses only $O(logn)$ temporary space. Give a clear, full specification of your algorithm. Argue why your algorithm is correct and justify the running time and temporary space usage of your algorithm. (Note: If you present an $O(n^2)$ time algorithm with $O(logn)$ temporary space or a $O(nlogn)$ time and $n + O(1)$-temporary space algorithm then you can get at most 5 marks.)
(Hint: It might be useful to think of a divide and conquer algorithm.)

(Bonus) [2 marks] Give an $O(n)$ time, $O(logn)$ temporary space algorithm to solve this problem. Give a formal proof of correctness of your algorithm.

Problem 3 [10 + 12 = 22 marks]

a) Run Bellman-Ford's shortest path algorithm on the graph shown below, with source node s. Clearly show the intermediate distance values of all the nodes after each iteration of the algorithm. Also draw the final shortest-path tree.



22

b) Given an weighted undirected graph G=(V,E), with weight w_e on edge e, design an algorithm to find the maximum spanning tree of G. That is, find a spanning tree of G, the weight of whose edges is maximal. Clearly describe the steps in your algorithm. Give a proof that your algorithm is indeed correct, i.e. it outputs a maximum spanning tree. What is the runtime of your algorithm?

Problem 4 [6 + 6 = 12 marks]

a) Write the Divide-and-Conquer based algorithm to compute the FFT of $X = (x_0, x_1, \ldots x_{n-1})$.

b) Huffman's algorithm is used to get an encoding of the symbols $\{a,b,c\}$ with frequencies f_a, f_b, f_c respectively. In each of the following cases, either give an example of frequencies (f_a, f_b, f_c) that would yield the specified code, or explain why the code cannot possibly be obtained (no matter what the frequencies are).

(i) Code: {1, 01, 00}
(ii) Code: {0, 1, 11}
(iii) Code: {10, 01, 00}

Problem 5 [4 + 12 = 16 marks]

a) The dynamic programming algorithm to compute the edit distance of strings of length m and n uses a table of size $m \times n$ and thus $O(mn)$ space. Show how this algorithm can be modified to use only $O(n)$ space, if we just want to compute the value of the edit distance (and not the actual optimal alignment).

b) You are given n coins of value $x_1, x_2, \ldots x_n$ and using them you need to make change for some amount m. Develop a dynamic programming algorithm to find out if it is possible to make change for the amount m, (by using each coin at most once). For e.g. if there are four coins whose values are 1, 5, 10 and 20, then it is possible to make change for 16 (1+5+10) but not for 40 (cannot use two coins of 20). Give a correctness argument for your algorithm. Also compute its run-time.

Problem 6 [2 + 12 = 14 marks]

a) State the decision version of the Travelling Salesman problem (TSP), clearly specifying all the inputs and outputs of the problem.

b) Prove that decision TSP is NP-Complete, by giving a suitable reduction from one of these problems: 3-SAT, Clique, Hamiltonian Cycle or Hamiltonian Path.