Algorithms Midterm

April 20, 2020, 09:10 - 12:10 (please measure 3 hours on your own)

Please answer the following questions and submit a photo/scan of your answer as a single pdf file on COOL. This is a closed book exam. All exam rules apply. You are not allowed to discuss with anyone or read any related materials, with the exception of one double-sided, hand-written A4 note previously prepared.

You may assume that all pairwise comparisons, additions and multiplications take $\Theta(1)$ time. If you want to apply any result or theorem that has been taught in class (including homeworks), you may do so but you must state the result or theorem clearly before using it.

Problem 1 Let $f(n)$ be a strictly increasing function (i.e. $f(x) < f(y)$ if $x < y$). All functions are positive functions from $\mathbb{R}^+$ to $\mathbb{R}^+$. Prove or disprove the following three statements. You may only use the definitions of asymptotic notations. Any other property of asymptotic notations must be proved before using. Answering true/false without any explanation will not receive any credit.

1. (4%) $0.5n^2 - n \in \Omega(n^2)$.
2. (8%) If $g(n^2) \in O(h(n^2))$, then $g(n) \in O(h(n))$.
3. (8%) If $g(f(n)) \in O(h(f(n)))$, then $g(n) \in O(h(n))$.

Problem 2

1. (5%) Solve $T(n) = 8T(\frac{n}{2}) + n^4$, $T(1) = 1$. You only need to obtain the asymptotic solution (in $\Theta()$ notation). If you use the master theorem, you must specify all parameters and briefly verify all conditions.
2. (10%) Let $T(n) = T(n-2) + 2T(\lfloor \frac{n}{2} \rfloor) + n$, for all $n \geq 3$. $T(1) = T(2) = 1$. Prove that $T(n) \in O(2^n)$ and $T(n) \in \Omega(n^2)$.

Problem 3 (15%) Given an array A of positive numbers $A[1], A[2], \ldots, A[n]$ with $A[1] = 1$ and $A[n] = 2^n$. Design an $O(\log n)$-time algorithm which finds a pair of numbers $(A[i], A[i+1])$ such that $A[i+1] > 2A[i]$. Briefly justify the correctness and the running time of the algorithm. Any algorithm which requires $\Omega(n)$ running time will receive very little credit.

Problem 4 (14%) Given n coins with k of them being fake coins. All real coins have equal weights and fake coins are lighter than real ones (different fake coins may have different weights). You can only use a balance to identify all fake coins. Each time, you can put any number of coins on each side of the balance, the balance only gives you three possible results: either the left side is heavier/lighter or two sides have equal weight. Prove that any method for identifying all fake coins must use the balance $\Omega(k \log \frac{n}{k})$ times in the worst case.

In problems 5 and 6, a company wants to construct machines to produce surgical masks for its own workers. The requirement is n masks per day. There are m types of machines $T_1, T_2, \ldots, T_m$. Machine T_i costs c_i and can produce n_i masks per day. Due to insufficient supply, only one machine of each type can be made. The goal is to satisfy the requirement (n per day) while minimizing the total cost.

For simplicity, assume that the machine types are sorted such that the price per mask is in the increasing order. In other words, $\frac{c_1}{n_1} \leq \frac{c_2}{n_2} \leq \ldots \frac{c_m}{n_m}$. Also, assume that $n_i \leq n$ for all i.

Problem 5 In this problem, consider the greedy algorithm which constructs $T_1, T_2, \ldots$ until there is enough supply.

1. (8%) Find an example in which the algorithm output is at least ten times more expensive than the optimal solution. If you cannot manage this problem, find an example in which the algorithm does not find the optimal solution gives partial credits.

2. (8%) Given an extra assumption that all c_is are within a factor of four to each other (i.e. $c_i \leq 4c_j$ and $c_j \leq 4c_i$ for every i, j). Prove that the algorithm output is at most five times more expensive than the optimal solution.

Problem 6

1. (10%) Design an algorithm to find the optimal solution (cheapest set of machines which satisfy the requirement). Your algorithm must run in time $O(mn)$ and use $O(mn)$ memory to receive full credits. If you cannot manage this problem, any algorithm which computes the minimum cost in polynomial time gives partial credits. Briefly justify the correctness and analyze the running time.

2. (10%) Given an additional assumption: For every integer i, if machine T_{2i-1} is made, then machines T_{2i} and T_{2i+1} cannot be made. (Machines T_{2i} and T_{2i+1} do not conflict. Machines T_{2i-1} and T_{2i+3} do not conflict.) Design a polynomial time algorithm which finds the minimum cost possible. Briefly justify the correctness and analyze the running time. You may assume that m is an even number.

Administrative issues:

1. The exam score distribution will be announced by email before 04/28(Tue). I will go through the solutions briefly in class next week (4/27).

2. If you think you are not performing well in the exam, you may redo this exam as a homework (all homework rules apply) and submit *online* before 04/26(Sun) midnight. This extra work will NOT affect your score unless your total score for the whole semester is one of the following 3 cases:

a) an undergrad student with total score 55-60

b) a grad student with total score 65-70

c) a phd student doing qualify exam subsitution with score 75-80 (please email me)

In the above three cases, you will be raised to the lowest passing grade if you perform reasonably well in this extra homework.