Algorithms Midterm

April 18, 2019, 09:10 - 12:10

Please answer the following questions on the answer sheets provided. Be sure to write your name and student ID on all answer sheets you use. No books, notes, or calculators may be used during the exam with the exception of one double-sided, hand-written A4 note. Read all questions first. You may not request for clarification after 9:40am.

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 all functions be positive. Prove or disprove the following three statements. You may only use the definitions of asymptotic notations. Any other property of asymptotic notations must be derived before using. Answering true/false without any explanation will not receive any credit.

1. (6%) If $f(n) \in \Omega(n^3)$, then $f(n) \in \omega(n^2)$.
2. (7%) If $f(n) \in O(g(n))$ and $g(n) \in o(h(n))$, then $f(\frac{1}{2}n) \in o(h(n))$.
3. (7%) If $f(n) \in O(g(n))$ and $g(n) \in o(n)$, then $f(\frac{1}{2}n) \in o(n)$.

Problem 2 Solve the following recurrences. 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.

1. (5%) $T(n) = 25T(\frac{n}{5}) + n^2 + n$, $T(1) = 5$.
2. (10%) $T(n) = \sum_{i=1}^{n-1}(i^2 - 2)T(i)$, for all $n \geq 3$. $T(1) = T(2) = 2$.

Problem 3 Cyclic shift is a permutation which shifts all elements of an array by a fixed number of positions (wrapped around). For example: the sequence $1, 2, 3, 4, 5$ becomes $5, 1, 2, 3, 4$ when cyclic-shifted by one position, and becomes $3, 4, 5, 1, 2$ when cyclic-shifted by three positions.

1. (12%) Given an array of n distinct elements $A[1 \ldots n]$ which is a cyclic shift of a sorted array. Design an $O(\log n)$-time algorithm to find the minimum. Briefly justify the correctness and the running time. Any algorithm which requires $\Omega(n)$ running time will receive very little credit.
2. (12%) Prove that any algorithm which solves the above problem using only pairwise comparisons must use $\Omega(\log n)$ comparisons in the worst case.
3. (6%) If the array may contain identical elements, prove that any algorithm must use $\Omega(n)$ time in the worst case. (hint: construct an example which forces the algorithm to read all inputs.)

Problem 4 (15%) In this problem, consider loading items onto trucks. There are n items with weights $w_1, w_2, \ldots, w_n$. Each truck has a weight limit W. The goal is to load all items onto trucks using as few trucks as possible. Assuming $W \geq w_1 \geq \cdots \geq w_n$ Consider the following greedy algorithm:

Order all trucks from $1, 2, 3, \ldots, i, \ldots$

For j = 1 to n,

  Put item j onto the first truck which the item can fit in.

1. Prove that for all trucks used by the algorithm, at most one of them can be less than half full.
2. Use the above result, suppose that the optimal solution uses k trucks, prove that the algorithm output uses at most $2k$ trucks. (You may use the result even if you cannot prove it.)
3. Find an example in which the algorithm output is not optimal.

Problem 5 (20%) In this problem, the goal is to plan a trip of total distance n kilometers. The trip may take several days. The only possible stopping points at night are integer points (points at $1, 2, \ldots, n$ kilometers from the start). Stopping at i costs c_i and recovers enough stamina to walk up to b_i kilometers the next day (i.e. the next stopping point can be anywhere from $i + 1$ to $i + b_i$). The trip starts at 0 on the first day and can only travel b_0 kilometers on the first day. The trip must stop at n (and spend cost c_n) on the last day.

1. Design an algorithm to find the trip with minimum cost. Your algorithm must only use $O(n^2)$ time and $O(n)$ memory space. Briefly justify the correctness and the running time / memory usage.
2. Instead of walking up to b_i kilometers, after staying at i, you can either take a fast trip of up to f_i kilometers or a slow trip of up to s_i kilometers the next day. However, taking fast trips on any two consecutive days is not allowed. Also, the trip must end in k days. Design a polynomial-time algorithm to find the minimum cost possible (do not need to find the corresponding trip). Briefly justify the correctness and analyze the running time. Any algorithm which runs in time polynomial in nk suffices. You may assume that such a trip exists.

Administrative issues:

1. The exam score distribution will be announced by email before 04/25(Thu). The answer sheets will be given back on 04/25(Thu) in class. I will go through the solutions briefly.

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 04/24(Wed) midnight. This extra work will NOT affect your score unless you are one of the following 3 cases:

a) an undergrad student with final score 55-60

b) a grad student with final 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.