Algorithms   Homework #1

Due Oct 12, 2020 before class.

Collaboration policy: You can discuss the problem with other students, but you must obtain and write the final solution by yourself. Please specify all of your collaborators (name and student id) for each question. If you solve some problems by yourself, please also specify "no collaborators". Homeworks without collaborator specification will not be graded.

Problem 1 (10%)

Prove or disprove the following statements. You may only use the definition of asymptotic notations.

1. $0.5n^2 - n \in \Omega(n^2)$.
2. If $f(n) \in \Omega(n^3)$, then $f(n) \in \omega(n^2)$.

Problem 2 (10%)

Sort the following functions into a list $f_1, f_2, \ldots$ such that $f_i = \Omega(f_{i+1})$ for all i. Identify all pairs of functions $f(n), g(n)$ such that $f(n) = \Theta(g(n))$. No explanation is needed.

$8^{\log_2 n}$	$n!$	$n^{0.01}$	2^{2n}	n^3+5n^2	$\log_2 n$
$\sqrt{n}+3$	$\ln n$	$(\log_2 n)!$	n^n	$(2n)!$	

Problem 3 (10%)

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. You may assume that all inputs to T are powers of 3.

1. $T(n) = 9T(\frac{n}{3}) + n^3, T(1) = 5$.
2. $T(n) = 9T(\frac{n}{3}) + n^2 + 20n \log n + 3, T(1) = 5$.

Problem 4 (20%)

Prove or disprove the following statements. You may only use the definition of asymptotic notations.

1. Let $f_1(n), f_2(n), \ldots, f_i(n), \ldots$ be an infinite series of functions. $f_i(n) = O(n)$ for all i. Let $g(k) = \sum_{j=1}^{k} f_j(j)$ (i.e. $g(1) = f_1(1), g(2) = f_1(1) + f_2(2)$, and so on), then $g(n) = O(n^2)$.
2. Let $f(n) = O(n)$. Let $g(k) = \sum_{j=1}^{k} f(j)$ (i.e. $g(1) = f(1), g(2) = f(1) + f(2)$, and so on), then $g(n) = O(n^2)$.

Problem 5 (20%)

Solve the following recurrences. You only need to derive the asymptotic solution (in $\Theta()$).

1. 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 $T(n) \in \Omega(n^2)$.
2. $T(n) = 3T(\frac{n}{3}) + \frac{n}{2\log_3 n}$, $T(1) = 1$.

Problem 6 (20%)

Prove or disprove the following statements:

1. If $T(n) = 2T(\frac{n}{2}) + f(n)$ and $f(n) = \Theta(n^2)$ then $T(n) = \Theta(f(n))$ for all $n = 2^k$.
2. If $T(n) = 2T(\frac{n}{2}) + f(n)$, and $f(n) = \Omega(n^2)$, then $T(n) = O(f(n))$ for all $n = 2^k$.

Problem 7 (20%)

A divide and conquer algorithm X works in the following way. Given a problem instance of input size n. If n is less than 4, then the algorithm solves the problem instance directly with a constant number of steps. If n is at least 4, algorithm X divides the given problem instance into exactly two smaller problem instances and solve them recursively. Furthermore, whenever algorithm X divides a given problem instance of size n into smaller instances with sizes a and b, then $a + b = n$ and $\frac{1}{4}n \leq a, b \leq \frac{3}{4}n$.

1. If the splitting and merging procedures require $2n$ steps for a problem instance of size n, derive the asymptotic running time $T(n)$. (in $\Theta()$).
2. If the splitting and merging procedures require n^2 steps for a problem instance of size n, derive the asymptotic running time $T(n)$. (in $\Theta()$).