CS 375: Theory Assignment #1

Due on February 12, 2016 at 2:20pm

Professor Lei Yu Section B1

I have done this assignment completely on my own. I have not copied it, nor have I given my solution to anyone else. I understand that if I am involved in plagiarism or cheating I will have to sign an official form that I have cheated and that this form will be stored in my official university record. I also understand that I will receive a grade of 0 for the involved assignment for my first offense and that I will receive a grade of F for the course for any additional offense.

Tim Hung

Problem 1

(10 points) Given the pseudo code below for bubble sort:

function BUBBLESORT(A)
    for $i = 1$ to $(length[A] - 1)$ **do** ▷ store next smallest element in A[i]
        for $j = length[A]$ downto $(i + 1)$ **do**
            if $A[j] < A[j - 1]$ **then**
                swap $A[j]$ and $A[j - 1]$
            end if
        end for
    end for
end function

a) (5 points) Let length[A] = n. What is the count for BubbleSort(A)? Show the steps necessary to derive your final answer. This question requires you to use the instruction count method from the textbook (also introduced in lecture 2 slides). Answers using asymptotic notations will receive 0 point.

Solution

function BUBBLESORT(A)
    for $i = 1$ to $(length[A] - 1)$ **do** ▷ c_2 n
        for $j = length[A]$ downto $(i + 1)$ **do** ▷ c_3 $n - 1$
            if $A[j] < A[j - 1]$ **then** ▷ c_4 $\sum_{j=2}^{n} t_j$
                swap $A[j]$ and $A[j - 1]$ ▷ c_5 $\sum_{j=2}^{n} t_j$
            end if
        end for
    end for
end function

b) (5 points) Show the worse case and best case time complexity in term of instruction counts.

Solution

Best Case: n. (Sorted array)
Worst Case: n^2 (Reversed array).

Problem 2

(28 points) Fill in all the missing values. For the f(n) column, you need to compute the sums and fill in the exact format of f(n) for the last two rows. For the last three columns, you need to fill in each cell with either yes or no.

$f(n)$	$g(n)$	$f(n) = O(g(n))$	$f(n) = \Omega(g(n))$	$f(n) = \Theta(g(n))$
$n^{2.125}$	$n^2 lg(n)$	yes	no	no
$\sqrt{n}$	n	yes	no	no
$n!$	$(n+1)!$	yes	no	no
$2^{n/2}$	2^n	yes	yes	yes
$\sum_{i=1}^{n} i = \frac{n^2+n}{2}$	n^2	yes	yes	yes
$\sum_{i=0}^{n-1} 4^i = \frac{4^n-1}{3}$	$n4^{(n-1)}$	yes	no	no

Problem 3

(10 points) Order the functions below by increasing growth rates (no justification required).

$$n^n \quad n\ln n \quad n^\epsilon (0 < \epsilon < 1) \quad 2^{\lg n} \quad \ln n \quad 10 \quad n! \quad 2^n$$

Let $g_i(n)$ be the *ith* function from the left after the ordering (the leftmost function has the slowest growth rate). In the order, $g_i(n)$ should satisfy $g_i(n) \in O(g_{i+1}(n))$.

Solution

$$10 \quad \ln n \quad n^\epsilon (0 < \epsilon < 1) \quad n\ln n \quad 2^{\lg n} \quad 2^n \quad n! \quad n^n$$

Problem 4

(12 points) Let $f(n)$ and $g(n)$ be asymptotically positive functions. Prove or show a counter example for each of the following conjectures.

a) $f(n) \in O(g(n))$ implies $2^{f(n)} \in O(2^{g(n)})$.

b) $f(n) \in O(g(n))$ implies $g(n) \in \Omega(f(n))$.

Solution a
This conjecture is true.
For example, let $f(n) = n$ and $g(n) = n^2$.
$f(n) \in O(g(n)) \Rightarrow n \in O(n^2)$ is true.
If $2^{f(n)} \in O(2^{g(n)})$, then $2^n \in O(2^{n^2})$, which is true as well.

Solution b
This conjecture is true.
Definition: $O(g(n)) = f(n)$: there exist positive constants c and n_0 such that $0 \leq f(n) \leq cg(n) \forall n \geq n_0$.
Definition: $\Omega(g(n)) = f(n)$: there exist positive constants c and n_0 such that $0 \leq cg(n) \leq f(n) \forall n \geq n_0$.
Therefore, if $f(n) \in O(g(n))$, then by definition, $g(n) \in \Omega(f(n))$ must be true as well.

Problem 5

(10 points) Prove $n^2 - 3n - 20 \in \Theta(n^2)$ using the original definition of Θ.

Solution

Definition: $\Theta(g(n)) = f(n)$: there exist positive constants c_1, c_2, and n_0 such that $0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n) \forall n \geq n_0$.

$$
\begin{aligned}
0 \leq c_1 g(n) &\leq n^2 - 3n - 20 && = c_2 g(n) \\
0 \leq c_1 n^2 &\leq n^2 - 3n - 20 && = c_2 n^2 \\
0 \leq c_1 &\leq 1 - \frac{3}{n} - \frac{20}{n^2} && = c_2 \\
&\text{We can take } c_1 \text{ to be } \frac{1}{2} \text{ and } c_2 \text{ to be } 1. \\
&\text{Therefore, } n_0 \text{ is } 10. \\
0 \leq \frac{1}{2} &\leq 1 - \frac{3}{10} - \frac{20}{100} && = 1 \\
0 \leq \frac{1}{2} &\leq \frac{100}{100} - \frac{30}{100} - \frac{20}{100} && = 1 \\
0 \leq \frac{1}{2} &\leq \frac{1}{2} && = 1
\end{aligned}
$$

Problem 6

(10 points) Disprove $n^3 \in O(n^2)$ using the original definition of O.

Solution

Definition: $O(g(n)) = f(n)$: there exist positive constants c and n_0 such that $0 \leq f(n) \leq cg(n) \forall n \geq n_0$.

$$
\begin{aligned}
0 \leq n^3 &\leq cg(n) \\
0 \leq n^3 &\leq cn^2 \\
0 \leq n &\leq c && \text{We can take } c \text{ to be 0.} \\
0 \leq n &\leq 0 \\
0 \leq 0 &\leq 0 && \text{Therefore, } n_0 \text{ is 0.}
\end{aligned}
$$

Problem 7

(10 points) Prove $n = \Omega(\lg n^2)$ using limit.

Definition: $\Omega(g(n)) = f(n)$: there exist positive constants c and n_0 such that $0 \leq cg(n) \leq f(n) \forall n \geq n_0$.

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty} \frac{n}{\lg n^2} = \lim_{n\to\infty} \frac{1}{\frac{1}{2n}} = \lim_{n\to\infty} 2n = \infty$$

Therefore, f(n) grows faster than g(n).

Problem 8

(10 points) Prove $n^a = \Omega(\lg^k n)$, where $k > 0, a > 0$ using limit.

Definition: $\Omega(g(n)) = f(n)$: there exist positive constants c and n_0 such that $0 \leq cg(n) \leq f(n) \forall n \geq n_0$.

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty} \frac{n^a}{\lg^k n} = \lim_{n\to\infty} \frac{an^{a-1}}{\frac{k \lg^{k-1} n}{n}} = \lim_{n\to\infty} \frac{a^1 n^a}{k \lg^{k-1} n} = \lim_{n\to\infty} \frac{a^2 n^a}{k(k-1) \lg^{k-2} n}$$

$$= \ldots = \lim_{n\to\infty} \frac{a^k n^a}{k! \lg^{k-k} n} = \lim_{n\to\infty} \frac{a^k}{k!} n = \frac{a^k}{k!} \lim_{n\to\infty} n = \frac{a^k}{k!} \infty = \infty$$

Therefore, f(n) grows faster than g(n).