CS 507 HW6

Stanley Akor

November 30 , 2022

Question 1

(1) $\frac{n(n+1)}{2} \in O(n^3)$

$\lim_{n\to+\infty} \frac{f(n)}{g(n)} < +\infty \implies f(n) \in O(g(n))$

$\lim_{n\to+\infty} \frac{n^2+n}{2} \times \frac{1}{n^3} = \lim_{n\to+\infty} \left(\frac{1}{2n} + \frac{1}{2n^2}\right) = 0 < +\infty$

Hence the statement is True.

(2) $\frac{n(n+1)}{2} \in \Theta(n^3)$

$0 < \lim_{n\to+\infty} \frac{f(n)}{g(n)} < +\infty \implies f(n) \in \Theta(g(n))$

$\lim_{n\to+\infty} \frac{n^2+n}{2} \times \frac{1}{n^3} = \lim_{n\to+\infty} \left(\frac{1}{2n} + \frac{1}{2n^2}\right) = 0 \notin (0, +\infty)$

Hence the statement is False.

(3) $\frac{n(n+1)}{2} \in \Theta(n^2)$

$0 < \lim_{n\to+\infty} \frac{f(n)}{g(n)} < +\infty \implies f(n) \in \Theta(g(n^2))$

$\lim_{n\to+\infty} \frac{n^2+n}{2} \times \frac{1}{n^2} = \lim_{n\to+\infty} \left(\frac{1}{2} + \frac{1}{2n}\right) = \frac{1}{2} \in (0, +\infty)$

Hence the statement is True.

(4) $\frac{n(n+1)}{2} \in O(n^2)$

$\lim_{n\to+\infty} \frac{f(n)}{g(n)} < +\infty \implies f(n) \in O(g(n))$

$\lim_{n\to+\infty} \frac{n^2+n}{2} \times \frac{1}{n^2} = \lim_{n\to+\infty} \left(\frac{1}{2} + \frac{1}{2n}\right) = \frac{1}{2} < +\infty$

Hence the statement is True.

(5) $\frac{n(n+1)}{2} \in \Omega(n^2)$

$0 < \lim_{n\to+\infty} \frac{f(n)}{g(n)} \implies f(n) \in \Omega(g(n))$

$\lim_{n\to+\infty} \frac{n^2+n}{2} \times \frac{1}{n^2} = \lim_{n\to+\infty} \left(\frac{1}{2} + \frac{1}{2n}\right) = \frac{1}{2} > 0$

Hence the statement is True.

Question 2

$i = 1$

sum $= 0$;

while$(i \leq n)$

if(f(i) $> k$)

then sum$+=$ f(i)

$i = 2 * i$

Analysis

Line 1 : We have an assignment operator, so the running time is 1. Thus line 1 $\in \Theta(1)$

Line 2 : We have an assignment operator so the running time is 1. Thus line 2 $\in \Theta(1)$

Line 3 : We have a loop whose iterate increases by a factor of two. Here we want to know how many times the condition $i \leq n$ is checked, and the number of times the loop body executes. Well, if the loop executes Q times, the number of times the condition $i \leq n$ is checked is Q+1. The loop body executes for i $= 2^0, 2^1, 2^2, ..., 2^{\log_2 n}$. That is the loop executes for Q=1+$\log_2 n$ times. Therefore, the condition $i <= n$ is checked for 2+$\log_2 n$ times. Thus line 3 $\in \Theta(\log n)$

Line 4 : Since f(i)$\in$ $\Theta(i)$ and 'if' condition has a running time of $\Theta(1)$ which yields $(\Theta(1) + \Theta(i)) = \Theta(i)$.

Line 5 : Here we have addition $\Theta(1)$, assignment $\Theta(1)$ and a function call $\Theta(i)$, which yields $\Theta(i)$ when combined.

Line 6 : Multiplication and assignment operations have a combined running time of $\Theta(1)$.

Now summing the running times from line 1 to line 6;

$$
\begin{aligned}
T(n) &= \Theta(1) + \Theta(1) + \Theta(\log n) + \Theta(\log n)\left[\Theta(i) + (\Theta(i) \cdot \Theta(i)) + \Theta(1)\right], \text{since lower-order terms are relatively insignificant} \\
&= \Theta(1) + \Theta(1) + \Theta(\log n) + \Theta(\log n) \cdot \Theta(i^2) \\
&= \Theta(i^2 \log n)
\end{aligned}
$$

Therefore, the running time is $\Theta(i^2 \log n)$.

Question 3

Definition

$\Omega(g(n)) = \{f(n) : \exists \text{positive constants}\, c \,\text{and}\, n_o \,\text{such that}\, 0 \leq cg(n) \leq f(n)\,, \forall n \geq n_o\}$

Here $f(n) = 4n^3 + 5n - 6$. Using the above definition, we have; $f(n) \geq cn^2, \forall n \geq n_o$.

Now let us check this condition: if $f(n) = 4n^3 + 5n - 6 \geq cn^2$, then

$$4n^3 + 5n - 6 \geq cn^2 \tag{1}$$

$$4n + \frac{5}{n} - \frac{6}{n^2} \geq c, \forall n \geq n_o \tag{2}$$

Now, let us choose $n_o = 1$, such that;

$$4n + \frac{5}{n} - \frac{6}{n^2} \geq c, \forall n \geq 1$$

Now substituting the smallest value of n (i.e 1) in the expression above, we shall have $c \leq 3$. To check that these values are correct, we substitute the values of $n \geq 1$ and $c \leq 3$ in equation 2.

$$4n + \frac{5}{n} - \frac{6}{n^2} \geq 3, \forall n \geq n_o = 1 \tag{3}$$

We observe that when n = 1 equation 3 is true. We conclude that Ω hold for $n \geq n_o = 1\,, c \leq 3$. Larger values of n_o leads to larger values of c (i.e $n_o = 2$, $c \leq 9$) but in any case, the statement is TRUE.

Question 4

The Ω notation is used to provide a lower bound. So when $f(n) = \Omega(g(n)) \Leftrightarrow \exists\, c, n_o : f(n) \geq cg(n) \geq 0,\, \forall n \geq n_o$. Therefore, saying that the running time of an algorithm f(n) is $\Omega(1)$ is meaningless is because describing the growth of a function with $\Omega(1)$ implies that f(n) has the same or higher rate of growth than that of a constant function. Well, this is trivial because virtually all algorithms with at least one step will have a complexity of $\Omega(1)$. Implying that $f(n)$ is $\Omega(1)$ does not give us any new or extra information.

Question 5

Arranged in the order of growth from lowest to highest: $\log^2 n$, $n^{\frac{1}{3}}$, $5\log(n + 100)^{10}$, $0.001n^4 + 3n^3 + 1$,3^n,2^{2n}