Take Home Assignment- CS2023-Week 2

INDEX NO: 210670N

QUESTION 1

Little Oh Notation (o)

$$\textbf{For every } \epsilon > 0 \textbf{ there exis a constant N such that}$$
$$|f(n)| < \epsilon|g(n)|$$

Definition using limits : $f(n)\ \epsilon\ o((g(n))\ if\ \ \lim_{n\to\infty}\frac{f(x)}{g(x)} = 0$

Examples:

$$let\ f(n)\ = 2n^2 + 1\ and\ g(n)\ =\ n^3$$

$$\lim_{n\to\infty}\frac{2n^2+1}{n^3} = \lim_{n\to\infty}\frac{2\frac{1}{n}+\frac{1}{n^3}}{1} = 0$$

$$f(n)\ \epsilon\ o((g(n))$$

$$let\ f(n)\ = c_x n^x + c_{x-1}n^{x-1} + c_{x-2}n^{x-2} + \ldots + c_1 n^1 + c_0\ and\ g(n)\ =\ n^{x+1}$$

$$f(n)\ \epsilon\ o((g(n))$$

Big Omega Notation (Ω)

For function g(n) , we define the big Omega of n the set ,

$\Omega(g(n)) = \{\ f(n) : \exists\ positive\ constants\ c\ and\ n_0\ such\ that\ \forall n \geq n_0,$

$$0\ \leq c(g(n)) \leq f(n)$$

Examples:

1. $let\ f(n)\ = \log_{10} n + c_0\ and\ g(n)\ =\ \log_9 n$

$$f(n)\ \epsilon\ \Omega((g(n))$$

2. $let\ f(n)\ = 2n^2 + 1\ and\ g(n)\ =\ n^2$

$$2n^2 + 1 \geq cn^2 \;\; for\ c = \ 1\ and\ all\ n \geq \ 1$$

$$f(n)\ \epsilon\ \Omega((g(n))$$

Little Omega Notation($\boldsymbol{\omega}$)

$f(n) is\ \omega\big(g(n)\big)\ if or\ any\ real\ constant\ c > 0\ \exists\ an\ integer\ constant\ n \geq 1\ such\ that$

$f(n) > cf(n)\ \ \forall n \geq n_0$

Definition using limits : $f(n)\ \epsilon\ \boldsymbol{\omega}((g(n))\ if\ \ \lim_{n\to\infty} \frac{f(x)}{g(x)} > \ 0$

$$let\ f(n)\ = n^3\ and\ g(n)\ =\ 2n^2 + 1$$

$$\lim_{n\to\infty} \frac{n^3}{2n^2 + 1} = \lim_{n\to\infty} \frac{n}{2 + \frac{1}{n^2}} = \frac{n}{2} > 0$$

$$f(n)\ \epsilon\ \boldsymbol{\omega}((g(n))$$

Examples:

QUESTION 2

Big-O Notation	Comparison Notation	Limit Definition	Description
o(g)	$f\ <\ g$	$\lim_{n\to\infty} \frac{f(x)}{g(x)} =\ 0$	Upper bound of growth of a function which is not tight
O(g)	$f\ \leq\ g$	$\lim_{n\to\infty} \frac{f(x)}{g(x)} <\ \infty$	Tight upper bound of growth of a function
$\Theta(g)$	$f\ =\ g$	$\lim_{n\to\infty} \frac{f(x)}{g(x)} \in\ \mathbb{R} > 0$	Tight bound of growth of a function
$\omega(g)$	$f\ >\ g$	$\lim_{n\to\infty} \frac{f(x)}{g(x)} >\ 0$	Lower bound of growth of a function which is not tight
Ω(g)	$f\ \geq\ g$	$\lim_{n\to\infty} \frac{f(x)}{g(x)} =\ \infty$	Tight lower bound of growth of a function

- Relationship between O(g) and o(g)

$$f(n)\ \epsilon\ o(g(n)) \Rightarrow\ f(n)\ \epsilon\ O(g(n))$$

but

$$f(n)\ \epsilon\ O(g(n)) \nRightarrow f(n)\ \epsilon\ o(g(n))$$

- Relationship between $\omega(g(n))$and $\epsilon\ \Omega(g(n))$

$$f(n)\ \epsilon\ \omega(g(n)) \Rightarrow\ f(n)\ \epsilon\ \Omega(g(n))$$

but

$$f(n)\ \epsilon\ \Omega(g(n)) \nRightarrow f(n)\ \epsilon\ \omega(g(n))$$

- Relationship between O(g) , $\Omega(g)$ and $\Theta(g)$

$$f(n)\ =\ \Theta(g(n))\ \ \text{iff}\ \ \text{f(n)} = O(g(n))\ \ \text{and f(n)} = \Omega(g(n))$$

QUESTION 3- PART (I)

Code	cost	Times(worst case)
Bubble Sort A		
for j = A.length to 2 do	c_1	n
swapped = false	c_2	(n-1)
for i = 2 to j do	c_3	$\sum_{j=2}^{n} j = \frac{n(n+1)}{2} - 1$
swapped = false	c_4	$\sum_{j=2}^{n}(j-1) = \frac{n(n-1)}{2}$
if (A|i — 1|> A[i]) then	c_5	$\sum_{j=2}^{n}(j-1) = \frac{n(n-1)}{2}$
temp = A|i|	c_6	$\sum_{j=2}^{n}(j-1) = \frac{n(n-1)}{2}$
A|i| = A|i — 1|	c_7	$\sum_{j=2}^{n}(j-1) = \frac{n(n-1)}{2}$

A|i — 1| = temp	c_8	$\sum_{j=2}^{n}(j-1) = \frac{n(n-1)}{2}$
swapped = true	c_9	$\sum_{j=2}^{n}(j-1) = \frac{n(n-1)}{2}$
if (! swapped) then	c_{10}	(n-1)
break;	c_{11}	0

T(n) = $c_1 n + (c_2 + c_{10})(n-1) + c_3(\frac{n(n+1)}{2} - 1) + (c_4 + c_5 + c_6 + c_7 + c_8 + c_9)\frac{n(n-1)}{2}$

Code	cost	Times(for the worst case)
Bubble Sort B		
n = A.length	c_1	1
do	c_2	n
swapped = false	c_3	(n)
for i = 2 to n do	c_4	$(n)(n-1)$
if (A|i — 1|> A[i]) then	c_5	$(n-1)$(n-1)
temp = A|i|	c_6	$(n-1)$(n-1)
A|i| = A|i — 1|	c_7	$(n-1)$(n-1)
A|i — 1| = temp	c_8	$(n-1)$(n-1)
swapped = true	c_9	$(n-1)$(n-1)
newLimit = i-1	c_{10}	$(n-1)$(n-1)
n = newLimit	c_{11}	n
while swapped	c_{12}	n

T(n) = $c_1 + (c_2 + c_3 + c_{11} + c_{12})(n) + (c_4 + c_5 + c_6 + c_7 + c_8 + c_9 + c_{10})(n-1)(n-1)$

QUESTION 3- PART(II)

- The time complexities for the worst case for the given two algorithms are not the same and the second algorithm takes more time than that of first for the worst case.

Question 3- part (iii)

- The worst case time complexity alone can be analyzed by considering the particular sorting algorithm for a reversely sorted array as the input array