09/20/24 APPLIED ALGORITHMS Name- Venkata Madhumit

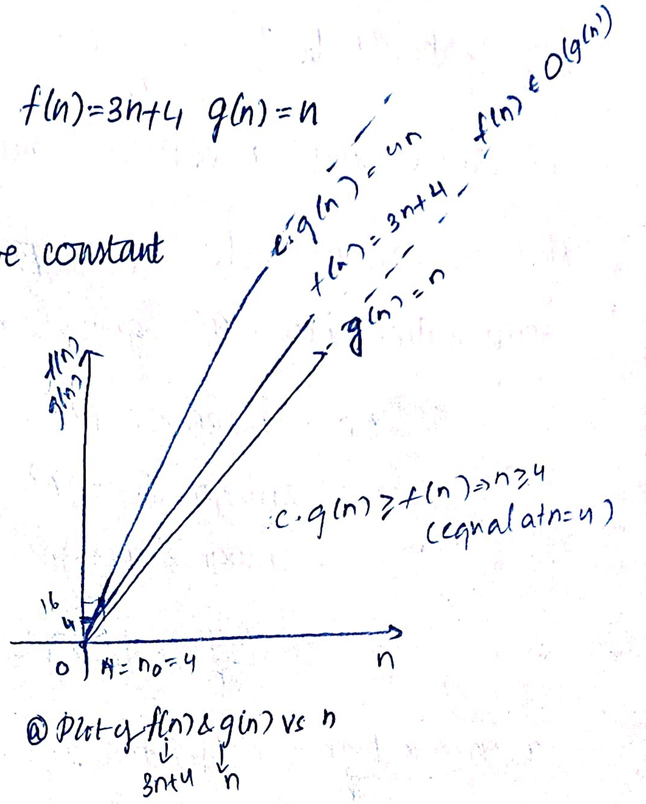
UID No - 2001456880

Question 11

a) Prove or disprove than $f(n) \in \Theta(g(n))$ where $f(n)=3n+4$ $g(n)=n$

Sol: To prove $f(n) \in O(g(n))$

$f(n) \le c \cdot g(n)$ for all $n \ge n_0$ 'c' is +ve constant



① Plot of f(n) & g(n) vs n (3n+4, n)

Given $f(n) = 3n+4$

$g(n) = n$

$3n+4 \le C(n)$

$4 \le cn - 3n$

$4 \le (c-3)\,n$

say $c=4$, $n_0=4$ then $4 \le (4-3)\,n$

$n \ge 4$

i.e $f(n) \in O(g(n))$

$\boxed{\text{i.e } 3n+4 \in O(g(n)) \text{ for } n \ge 4}$

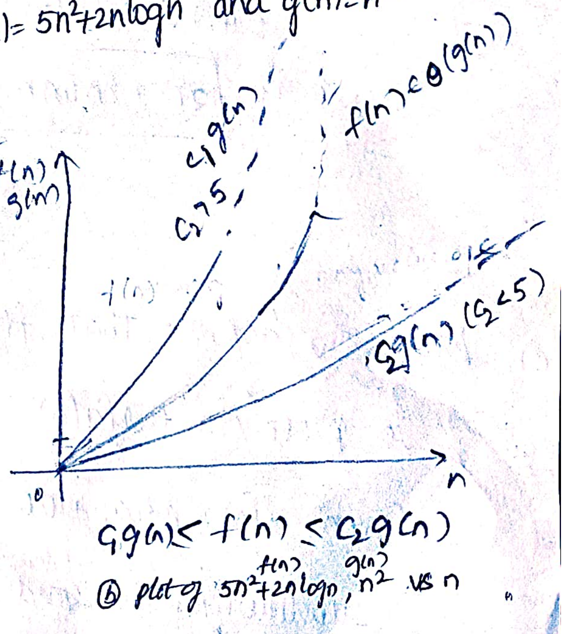
b) Prove or disprove that $f(n) \in \Theta(g(n))$ where $f(n) = 5n^2+2n\log n$ and $g(n)=n^2$

To prove $f(n) \in \Theta(g(n))$

Sol: $c_1 g(n) \le f(n) \le c_2 g(n)$

i.e $f(n) \ge c_1 g(n)$ i.e $f(n) \in \Omega(g(n))$

$f(n) \le c_2 g(n)$ i.e $f(n) \in O(g(m))$



$c_1 g(n) \le f(n) \le c_2 g(n)$

② plot of $5n^2+2n\log n$, n^2 vs n

Given $f(n) = 5n^2+2n\log n$, $g(n) = n^2$

$f(n) \ge c_1 g(n)$

$5n^2+2n\log n \ge c_1 (n)^2$

$2n\log n \ge (c_1-5)\,n^2$

$n\log n \ge \left(\frac{c_1-5}{2}\right) n^2$ (divide by n^2)

$\frac{\log n}{n} \ge \left(\frac{c_1-5}{2}\right) \rightarrow$ eq①

for Large values of n i.e $n \to \infty$, $\frac{\log n}{n} \to 0$ (tends to zero), for eq ① to satisfy above condition $\frac{\log n}{n} \geq \frac{c_1-5}{2}$ (as $\frac{\log n}{n} \to 0$)

c_1 should be less than '5' $c_1 \leq 5$, then $\frac{c_1-5}{2} \leq 0$ (true for Large n value)

i.e there exists a constant such $f(n) \in \Omega(g(n))$ i.e $f(n) \geq c_1 g(n)$ (for $c_1 \leq 5$).

Now we should check for upper bound $f(n) \leq c_2 g(n)$

subsituting $f(n)$ & $g(n)$ gives us

$$5n^2 + 2n\log n \leq c_2 \cdot n^2$$

$$2n\log n \leq (c_2-5)n^2$$

$$n\log n \leq \left(\frac{c_2-5}{2}\right)n^2 \quad \text{dividing by } n^2 \text{ (positive no)}$$

$$\frac{\log n}{n} \leq \left(\frac{c_2-5}{2}\right) \to eq(2)$$

as $n \to \infty$, $\frac{\log n}{n} \to 0$ (tends to zero), for large values of n, so to hold eqn (2)

c_2 should be greater than 5 i.e $c_2 > 5$ for all $n \geq n_0$ i.e $f(n) \in O(g(n))$

$\therefore$ we can conclude that $\underbrace{c_1 g(n) \leq f(n)}_{f(n)} \leq c_2 g(n)$, satisfies both conditions

lies in between upper & lower bound of function g(n)

$f(n) \in \Theta(g(n))$ for $c_2 > 5$, $c_1 < 5$

i.e $\boxed{5n^2 + 2n\log n \in \Theta(n^2)}$ for large values of n' $n \geq n_0$

c) Prove or disprove that $f(n) \in \Omega(g(n))$ where $f(n) = n\log n + 100$, $g(n) = n$

Sol: To prove $f(n) \in \Omega(g(n))$, it should satisfy condition $f(n) \geq c \cdot g(n)$

Given $f(n) = n\log n + 100$, $g(n) = n$

substituting in inequality condition

$$n\log n + 100 \geq c \cdot n \quad \text{for } n > 0$$

$$\log n + \frac{100}{n} \geq c \quad \text{divide by 'n' on both sides}$$

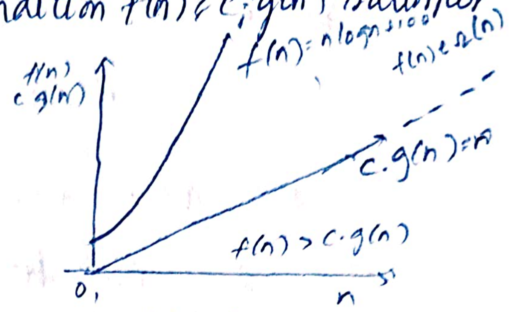
So for large values of 'n' $n \to \infty$, $\frac{100}{n} \to 0$, $\log n \to \infty$ (increases and more than c always.)

so the inequality $\log n + \frac{100}{n} \leq c$ holds for any value of c.

Since there is a constant $c>0, n \geq n_0 \Rightarrow$ the condition $f(n) \geq c \cdot g(n)$ satisfies

so we can say that $f(n) \in \Omega(g(n))$

$\boxed{\text{i.e } n\log n + 100 \in \Omega(n)}$



© plot of $f(n), g(n)$ vs n
$n\log n + 100$, n

d) Prove or disprove than $f(n) \in \Theta(g(n))$ where $f(n) = n^{\log n}$ and $g(n) = 2^n$

Sol: Given $f(n) = n^{\log n}$, $g(n) = 2^n$

for $f(n) \in \Theta(g(n))$ ~~f(n)~~ should satifies inequality $c_1 g(n) \leq f(n) \leq c_2 g(n)$

$f(n) \leq c_2 \cdot g(n)$ i.e $f(n) \in O(g(n))$

$n^{\log n} \leq c_2 2^n$ apply log on both sides

$\log n \log n \leq \log(c_2 2^n)$

$(\log n)^2 \leq \log c_2 + \log_2 2^n$

$(\log n)^2 \leq \underbrace{\log c_2}_{\text{constant}} + n \rightarrow$ eq① $f(n) \in O(g(n))$

n grows ~~slower~~ ~~faster~~ faster than $\log n$, than same with $(\log n)^2$, for large values (enough 'n' value to satisfy condition) 'n', the inequality always satisfies, can be for any value of c

i.e $n^{\log n} \leq c_2 \cdot 2^n$ for large values 'n'

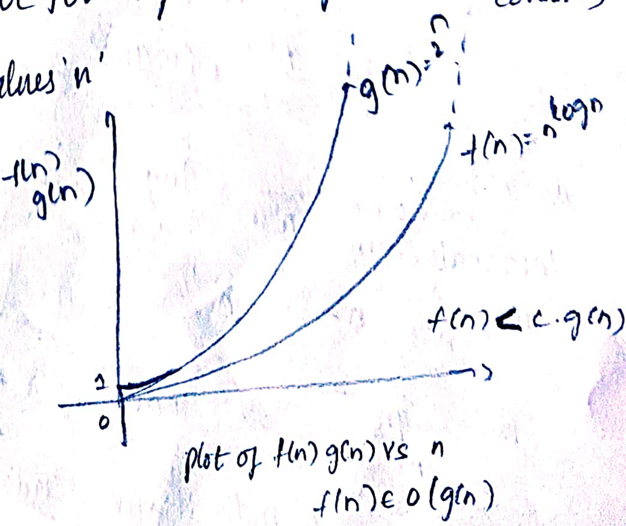
$f(n) \geq c_1 g(n)$

$n^{\log n} \geq c_1 2^n$

$\log n \log n \geq \log(c_1 \cdot 2^n)$

$(\log n)^2 \geq \log c_1 + \log_2 2^n$

$(\log n)^2 \geq \log c_1 + n \rightarrow$ eq②



plot of $f(n)$ $g(n)$ vs n
$f(n) \in O(g(n))$

as $n \to \infty$ 'n' grows faster than $(\log n)^2$ i.e for large values of 'n', this inequality donot satisfy, no constant 'c' can satisfy ~~[illegible]~~ to prove this i.e $f(n) \not\geq c_1 g(n)$ i.e $f(n) \notin \Omega(g(n))$

Since $f(n)$ doesnot hold lower bound condition $f(n) \ngtr c_1 g(n)$ and $f(n)$ is always ~~greater~~ less than $c_1 g(n)$ i.e $f(n) < c_2 g(n)$, we can say that

$f(n) \notin \Theta(g(n))$ grows slower than $g(n)$

i.e $n \log n$ always will grow slower than 2^n

so it can't hold the Theta notation

$\boxed{n \log n \notin \Theta(2^n)}$ it can be represented as upper bound condition

$n \log n \in O(2^n)$ or $o(2^n)$ (big oh or little-oh) can be used

e) Prove or disprove $f(n) \in o(g(n))$ where $f(n) = n^{1.5} + \log^2 n$ and $g(n) = n^2$

Sol: $f(n) \in o(g(n))$, if it has strictly ~~greater~~ less than that of $c\,g(n)$

i.e $\boxed{f(n) < c \cdot g(n)}$ f(n) should grow very slow compared to g(n)

Given

$f(n) = n^{1.5} + \log^2 n$

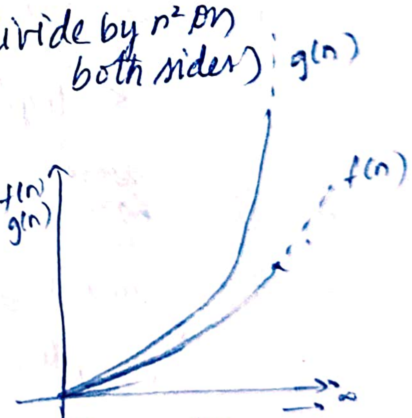
$g(n) = n^2$

$= n^{1.5} + (\log n)^2 < c \cdot n^2$ (n^2 is positive divide by n^2 on both sides)

$= \dfrac{n^{1.5} + (\log n)^2}{n^2} < c$

$= \dfrac{n^{1.5}}{n^2} + \dfrac{(\log n)^2}{n^2} < c$

i.e $\lim_{n \to \infty} \dfrac{f(n)}{g(n)} = 0$



$f(n) \in o(g(n))$, $g(n) > f(n)$

g(n) grows way faster than f(n)

plot $f(n), g(n)$ vs n

$\boxed{n^{1.5} + \log^2 n \quad n^2}$

as 'n' value increases / large values of n: $\dfrac{1}{n^{1/2}} + \dfrac{(\log n)^2}{n^2} < c$

for $\dfrac{1}{n^{1/2}}$ $n \to \infty$, $n^{-1/2} \to 0$ i.e $\dfrac{1}{n^{1/2}} \to 0$ (tends to zero)

for $\dfrac{(\log n)^2}{n^2}$, $n \to \infty$, value of $\dfrac{(\log n)^2}{n^2} \to 0$ (tends to zero)

i.e both terms tends to zero then $\dfrac{1}{n^{1/2}} + \dfrac{(\log n)^2}{n^2} \to 0$ (as $n \to \infty$)

i.e $c > 0$ for large values of n

i.e $\lim_{n \to \infty} \dfrac{f(n)}{g(n)} = 0$, as $n \to \infty$ $\dfrac{1}{n^{1/2}} + \dfrac{(\log n)^2}{n^2} \to 0 \Rightarrow \boxed{n^{1.5} + (\log n)^2 \in o(n^2)}$

$\boxed{\therefore f(n) \in o(g(n))}$ i.e $f(n) < c \cdot g(n)$, the condition holds for all positive values of 'c' and large values of n

09/19/24 APPLIED ALGORITHMS

Name - Venkata Madhumitha
UID No :- 2001456880

Question 2

Time complexities of following functions

b) def function2(n): By frequency count method

```
i=n
while i>0:
    i= i//3
return
```

→ loop runs till $i>0$

iteration 1st $i=n \rightarrow i=n/3$

2st $i=n/3 \rightarrow i=n/9$

3rd $i=n/9 \rightarrow i=n/27$

⋮

kth $i=n/3^{k+1} \rightarrow i=n/3^k$

i.e the loop runs till $i>0$, breaks if $i\leq 0$

$\frac{n}{3^k}\leq 0$

$3^k \geq n$

$k \geq \log_3 n$

can be written as order of $(\log n)$ → $\boxed{O(\log n) = \text{Time complexity for function2}}$

c) def function 3(n): By frequent counting.

```
i=1
while i*i<n:
    i+=1
return
```

→ k times loop runs till $i*i<n$, breaks $i*i \geq n$

iteration 1st $i=1*1 \rightarrow i=2$

2nd $i=2*2<n \rightarrow i=3$

3rd $i=3*3<n \rightarrow i=4$

⋮

kth times $i=k*k<n \rightarrow i=k+1$

The loop breaks if $k^2 \geq n$

$k \geq \sqrt{n}$

i.e Time complexity of function3 is order of $\sqrt{n}$

$\boxed{\text{function}_{(3)} \rightarrow O(\sqrt{n})}$

d) def function4(n):

By frequency count method

```
i=n
while i>0:                  -> logn (as i is dividing by 2 each time
    for j in range(0,n):    -> say k times
        for k in range(0,j):   -> j times each iteration
            x=k
    i=i//2
return
```

for $j=0 \rightarrow k=0$ do not go through loop

$j=1 \rightarrow k=0 \rightarrow 1$

$j=2 \rightarrow k=0,1 \rightarrow 2$ times

$j=3 \rightarrow k=0,1,2 \rightarrow 3$ times

$j=4 \rightarrow k=0,1,2,3 \rightarrow 4$ times

⋮

$j=n-1 \rightarrow k=0,1,\ldots \rightarrow n-1$ times

i.e $\log n * \sum_{j=0}^{n-1} j$

$\log n * \frac{(n-1)(n-1+1)}{2}$

$\log n * \frac{n(n-1)}{2} \rightarrow \frac{(n^2-n)\log n}{2} \rightarrow$ order of $n^2 \log n$

∴ Time complexity of $\boxed{\text{function4 is } O(n^2 \log n)}$

e) def function5(n)

By frequency count method

```
i=2           -> 1
while i<n:    -> k times
    i=i*i
return
```

$i=2 \xrightarrow{2<n}$ $i=4 \rightarrow 2^2 \rightarrow 2^{2^1}$

$i=4 \xrightarrow{4<n}$ $i=16 \rightarrow 2^4 \rightarrow 2^{2^2}$

$i=16 \xrightarrow{16<n}$ $i=196 \rightarrow 2^{16} \rightarrow 2^{2^4}$

⋮

$i=k \xrightarrow{k<n}$ $i=2^{2^k}$

$2^{2^k}<n$, it continues

loop breaks if $2^{2^k} \geq n$

apply log on both side

$\log_2 2^{2^k} \geq \log_2 n$

$2^k \geq \log_2 n$ apply log again

$k \geq \log_2(\log_2(n))$

Overall time complexity of $\boxed{\text{function5} \rightarrow \text{order of } \log(\log(n)) \text{ i.e } O(\log(\log(n)))}$

1) def function1(n):

```
for i in range(0,n):      → This runs for n+1 times
    for j in range(0,i):  → i times
        k = j             → assignment
        while k<n:        → n-k times
            k+=1          → increment operation.
return
```

$i=0 \quad j=0 \quad k=0 \to k=1 \to n-1$

$i=1 \quad j=0,1 \quad k=0 \to n-1$ times

$k=1 \ldots n-1$ times

$i=2 \quad j=0,1,2 \quad k=0 - n-1$

$k=1 - n-1 \to n-1$ times

$k=2 - n-2 \to n-2$ times

outer loop runs for n+1 times

Inner for loop → i times → same as in function 4 (prev)

i.e $\sum_{i=1}^{n-1} i = \frac{(n-1)(n-1+1)}{2} \to \frac{n(n-1)}{2} \to$ order of n^2

~~Inner~~ while loop runs for n-k times

i.e ~~k=0~~, The loop breaks when $k \geq n$ where $k = n-j$

$$\sum_{j=0}^{i-1} (n-j) = ni - \frac{(i-1)(i)}{2}$$

$$= \frac{ni - i^2 + i}{2} \to \frac{i(n-1) - i^2}{2} \to \text{order of } n^2$$

The overall time complexity → order of n^2

$\boxed{\therefore \text{Function1} = O(n^2)}$