Aditi Banerji
38 - F

Tutorial - 1

①

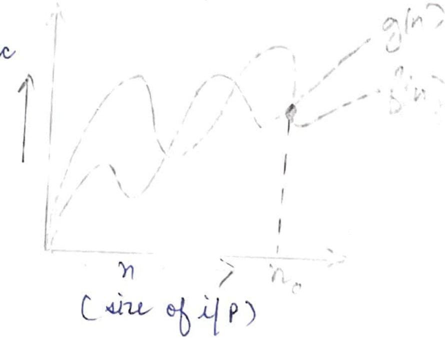
Q1. What do you understand by Asymptotic notation, define different asymptotic notation with example.

i) Big O (n)

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

if

$f(n) \leq g(n) \times C \;\forall\; n \geq n_0$



for some constant, $c > 0$

$g(n)$ is "tight" upper bound of $f(n)$

eg: $f(n) \Rightarrow n^2 + n$
$g(n) \Rightarrow n^3$

$n^2 + n \leq c * n^3$

$n^2 + n = O(n^3)$

ii) Big Omega (Ω)

Aditi

when $f(n) = \Omega(g(n))$

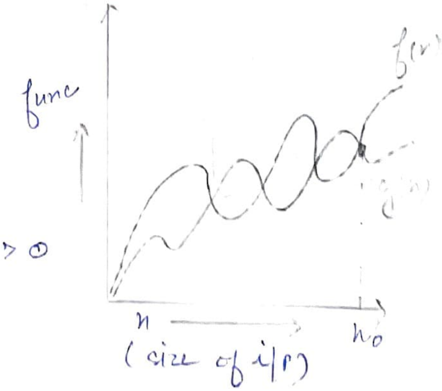
means $g(n)$ is "tight" lowerbound of $f(n)$ ie $f(n)$ can go beyond $g(n)$

ie. $f(n) = \Omega\, g(n)$

if and only if

$f(n) \geq c \cdot g(n)$

$\forall\; n \geq n_0$ and c = constant > 0



Ex: $f(n) \Rightarrow n^3 + 4n^2$

$g(n) \Rightarrow n^2$

ie $f(n) \geq c * g(n)$

$n^3 + 4n^2 = \Omega(n^2)$

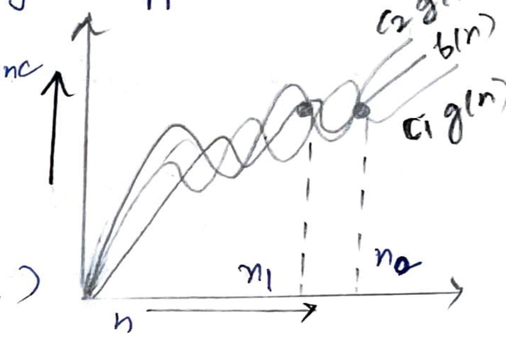
(iii) Big Theta (Θ)

when $f(n) = \Theta(g(n))$ gives the tight upperbound and lowerbound both.

ie $f(n) = \Theta(g(n))$

if and only if

$c_1 * g(n_1) \leq f(n) \leq c_2 * g(n_2)$



for all $n \geq \max(n_1, n_2)$, some constant $c_1 > 0$ & $c_2 > 0$

ie $f(n)$ can never go beyond $c_2 g(n)$ and will never come down of $c_1 g(n)$

Ex: $3n+2 = \Theta(n)$ as $3n+2 \geq 3n$ &

$3n+2 \leq 4n$ for n, $c_1 = 3$, $c_2 = 4$ & $n_0 = 2$

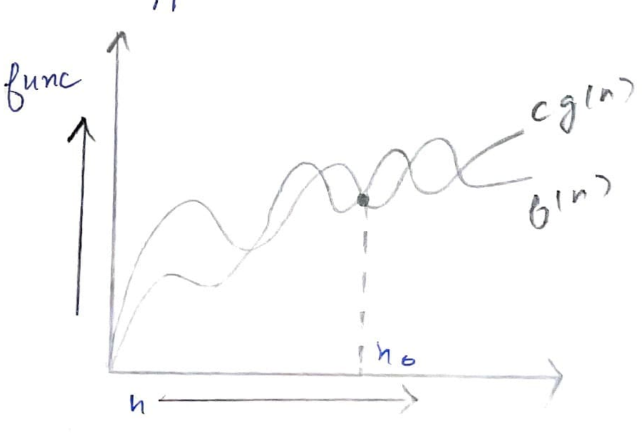
iv) Small o (o)

when $f(n) = o\, g(n)$ gives the upper-bound.

i.e $f(n) = o\, g(n)$

if and only if $f(n) < c * g(n)$

$\forall\ n > n_0$ & $c > 0$



Ex: $f(n) = n^2$, $g(n) = n^3$

$f(n) < c * g(n)$

$n^2 = o(n^3)$

v) Small omega (ω)

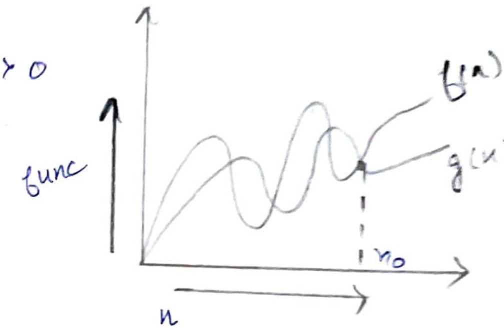
small omega gives the 'lower bound' ie

$f(n) = \omega(g(n))$

where $g(n)$ is lower bound of func $f(n)$

if and only if $f(n) > c * g(n)$

$\forall\ n > n_0$ & some constant, $c > 0$



Ex:-

Q2. what should be time complexity of:

Aditi

```
for ( int i = 1 to n)
{
      i = i * 2;          → O(1)
}
```

for i → 1, 2, 4, 8 . . . n times

ie series is a G.P

so $a = 1$, $r = \frac{2}{1}$

k^{th} value of GP :

$$t_k = ar^{k-1}$$

$$t_k = 1(2)^{k-1}$$

$$2n = 2^k$$

$$\log_2(2n) = k \log 2$$

$$\log_2 2 + \log_2 n = k$$

$$\log_2 n + 1 = k \quad \text{(Neglecting '1')}$$

So time Complexity $T(n) \Rightarrow O(\log_2 n)$

Q3. $T(n) \Rightarrow \{ \; 3T(n-1)$ if $n > 0$

otherwise 1

}

ie $T(n) \Rightarrow 3T(n-1)$ — (1)

$T(n) \Rightarrow 1$

put $n \Rightarrow n-1$ in (1)

$T(n-1) \Rightarrow 3T(n-2)$ — (2)

put (2) in (1)

$T(n) \Rightarrow 3 \times 3T(n-2)$

$T(n) \Rightarrow 9T(n-2)$ — (3)

put $n \Rightarrow n-2$ in (1)

$T(n-2) = 3T(n-3)$

put in (3)

Adita

$$T(n) = 27T(n-3) \quad — (4)$$

generalizing series,

$$T(n) = 3^k \, T(n-k) \quad — (5)$$

for k^{th} terms, let $n-k=1$ (Base case)

$$k = n-1$$

put in (5)

$$T(n) = 3^{n-1} \, T(1)$$

$$T(n) = 3^{n-1} \quad \text{(neglecting } 3^1\text{)}$$

$$T(n) = O(3^n)$$

Q4. $T(n) \Rightarrow \{$

$2T(n-1) - 1$ if $n > 0$,

otherwise 1

$\}$

$$T(n) = 2T(n-1) - 1 \quad — (1)$$

put $n = n-1$

$$T(n-1) = 2T(n-2) - 1 \quad — (2)$$

put in (1)

$$T(n) = 2 \times (2T(n-2) - 1) - 1 \quad —$$

$$= 4T(n-2) - 2 - 1 \quad — (3)$$

Aditi

put $n = n-2$ in (1)

$$T(n-2) = 2T(n-3) - 1$$

put in (1)

$$T(n) = 8T(n-3) - 4 - 2 - 1 \quad - (4)$$

6

generalising series,

$$T(n) = 2^k T(n-k) - 2^{k-1} - 2^{k-2} \dots 2^0$$

* kth term

Let $n - k = 1$

$k = n - 1$

$$T(n) = 2^{n-1} T(1) - 2^k \left(\frac{1}{2} + \frac{1}{2^2} + \dots \frac{1}{2^k} \right)$$

$$= 2^{n-1} - 2^{n-1} \left(\frac{1}{2} + \frac{1}{2^2} \dots \frac{1}{2^{n-1}} \right)$$

ie series is in G.P

$a = 1/2$, $r = 1/2$

So,

$$T(n) = 2^{n-1} \left(1 - \left(\frac{1}{2} \left(\frac{1 - (1/2)^{n-1}}{1 - 1/2} \right) \right) \right)$$

$$= 2^{n-1} \left(1 - 1 + (1/2)^{n-1} \right)$$

$$= \frac{2^{n-1}}{2^{n-1}}$$

$T(n) = O(1)$ Ans,

Q5. What should be time complexity of

```
int i=1, s=1;
while ( s<=n)
{
    i++;
    s=s+i;
    printf("#");
}
```

Aditi

here, $i = 1, 2, 3, 4, 5, 6 \ldots$ n times

$S = 1 + 3 + 6 + 10 + 15 \ldots + n$ times — (1)

∵ kth terms,

$T_k = 1 + 2 + 3 + 4 + \ldots$ k times

$= \frac{1}{2} k(k+1)$

for k iterations,

$\frac{k(k+1)}{2} <= n$

$\frac{k^2 + k}{2} <= n$ neglecting $+k/2$

$O(k^2) <= n$

$k = O(\sqrt{n})$

∴ Time Complexity $T(n) \Rightarrow O(\sqrt{n})$.

06. Time Complexity of

```
void fn(int n)
{
    int i, count = 0;
    for (i=1; i*i<=n; i++)
    count++;
}
```

Since, $i^2 <= n$ ie $i <= \sqrt{n}$

$i = 1, 2, 3, 4 \ldots \sqrt{n}$

By Summation Method,

$\sum_{i=1} 1 + 2 + 3 + \ldots + \sqrt{n}$

Aditi

$$T(n) = \frac{\sqrt{n} \times (\sqrt{n}+1)}{2}$$

$$T(n) = \frac{n \times \sqrt{n}}{2}$$

$$T(n) = O(n)$$

Q7. Time complexity of

```
void function (int n)
{
    int i, j, k, count = 0;
    for (int i = n/2 ; i<=n; i++ )
    for( j= 1 ; j<=n; j=j*2)
    for (k=1; k<=n; k=k*2)
        Count++;
}
```

Since, for $k = k^2$

$k = 1, 2, 4, 8 \ldots n$

Since series is in G.P

So, $a=1$, $r=2$

k^{th} term: $\frac{a(r^k-1)}{r-1}$

$$= \frac{1(2^k-1)}{1}$$

$$n = 2^k - 1$$

$$n+1 = 2^k$$

$$\log_2(n) = k$$

Aditi

i	j	k
1	log(n)	log(n) * log(n)
2	log(n)	log n * log n
⋮	⋮	⋮
n	log(n)	log n * log(n)

T.C ⇒ $O(n * \log n * \log n)$

⇒ $O(n \log^2(n))$ Ans.

Q8. Time complexity of

```
void function (int n)
{
    if (n == 1) return;
    for (i = 1 to n) {
    for (j = 1 to n) {
        printf("*");
    }
    }
    function (n-3);
}
```

for (i = 1 to n)

we get j = n times every turn

∴ $i * j = n^2$

* kth

Now,

$T(n) = n^2 + T(n-3)$;

$T(n-3) = (n^2 3)^2 + T(n-6)$;

$T(n-6) = (n^3 6)^2 + T(n-9)$;

and $T(1) = 1$;

Aditi

Now, subsitute each value in T(n)

$T(n) = n^2 + (n-3)^2 + (n-6)^2 + \dots + 1$

Let

$n - 3k = 1$

$k = (n-1)/3$

total terms $= k+1$

$T(n) = n^2 + (n-3)^2 + (n-6)^2 + \dots + 1$

$T(n) \leq kn^2$

$T(n) \leq \frac{(n-1)}{3} \times n^2$

So, $T(n) = O(n^3)$ Ans.

Q9. Time Complexity of :-

```
void function (int n)
{
    for (int i=1 to n) {
        for (int j=1 ; j<=n; j=j+i) {
            printf ("*");
        }
    }
}
```

for: $i = 1$ $\quad j = 1+2+ \dots \quad (n \geq j+i)$

$i = 2$ $\quad j = 1+3+5 \dots \quad (n \geq j+i)$

$i = 3$ $\quad j = 1+4+7 \dots \quad (n \geq j+i)$

Aditi

nth term of A.P is

$T(m) = a + d \times m$

$T(m) = 1 + d \times m$

$(n-1)/d = m$

for $i = 1$ $\quad (n-1)/1$ times

$i = 2$ $\quad (n-1)/2$ times

$\vdots$

$i = n-1$

we get,

$$T(n) = i_1 j_1 + i_2 j_2 + \cdots i_{n-1} j_{n-1}$$

$$= \frac{(n-1)}{1} + \frac{(n-2)}{2} + \frac{(n-3)}{3} + \cdots 1$$

$$= n + \frac{n}{2} + \frac{n}{3} + \cdots \frac{n}{n-1} - n \times 1$$

$$= n\left[1 + 1/2 + 1/3 + \cdots 1/n-1\right] - n \times 1$$

$$= n \times \log n - n + 1$$

Since $\int 1/x = \log x$

$T(n) = O(n \log n)$ — Ans.

Q10. For the functions n^k & c^n, what is the asymptotic Relationship b/w these functions?
Assume that $k >= 1$ & $c > 1$ are constants. find out the value of c & no. of which relationship holds.

As given n^k and c^n

Relationship b/w n^k & c^n is

$n^k = O(c^n)$

$n^k \leq a(c^n)$

$\forall\ n \geq n_0$ & constant, $a > 0$

for $n_0 = 1$, $c = 2$

Aditi

$\Rightarrow 1^k < a^2$

$\Rightarrow n_0 = 1$ & $c = 2$ Ans

Aditi