Tutorial -1

Name → ~~Raj~~ Rajarshi Bhattacharjee
Section → AI&DS
Roll No → 15
Univ Roll No → 2017662

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

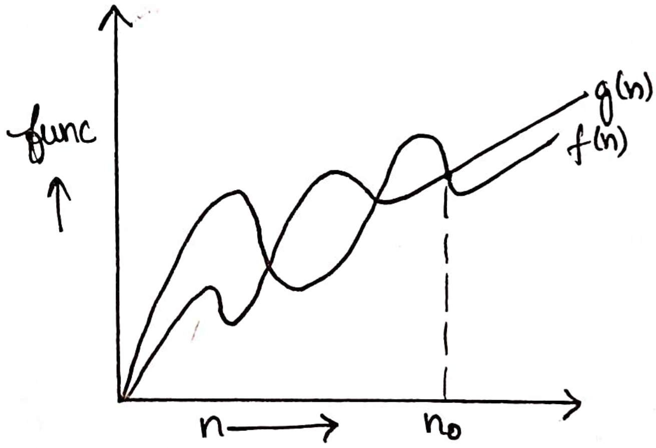
(i) Big O(n)

$f(n) = O(g(n))$

if $f(n) \leq g(n) \times c \quad \forall n \geq n_0$

for some constant, $c > 0$.

g(n) is 'tight' upper bound of f(n)



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

$g(n) \Rightarrow n^3$

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

$n^2 + n \cancel{\leq} = O(n^3)$

(ii) Big Omega (Ω)

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

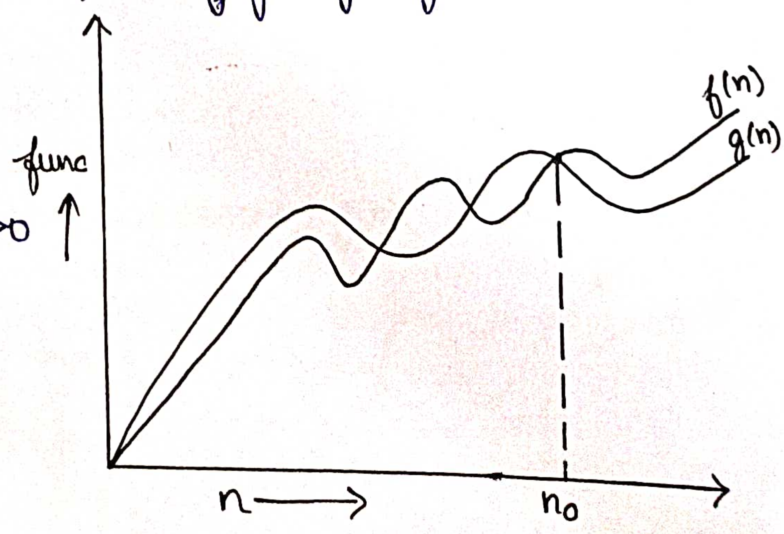
means g(n) is "tight" lowerbound of f(n) ie f(n) can ~~go~~ go beyond g(n).

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

if and only if

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

$\forall n_2 > n_0$ and c = constant > 0



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

$g(n) \Rightarrow g n^2$

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

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

(iii) Big Theta (Θ)

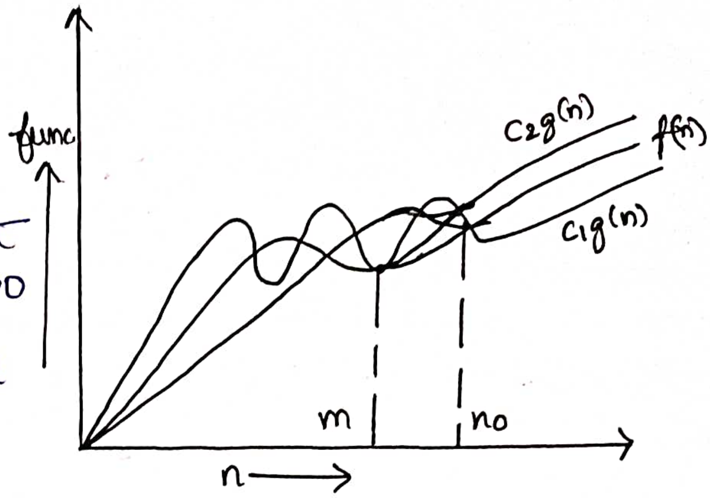
When $f(n) = \Theta(g(n))$ gives the tight upperbound and lowerbound both

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 $n \geq max(n1, n2)$, some constant $c_1 > 0$ and $c_2 > 0$



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)

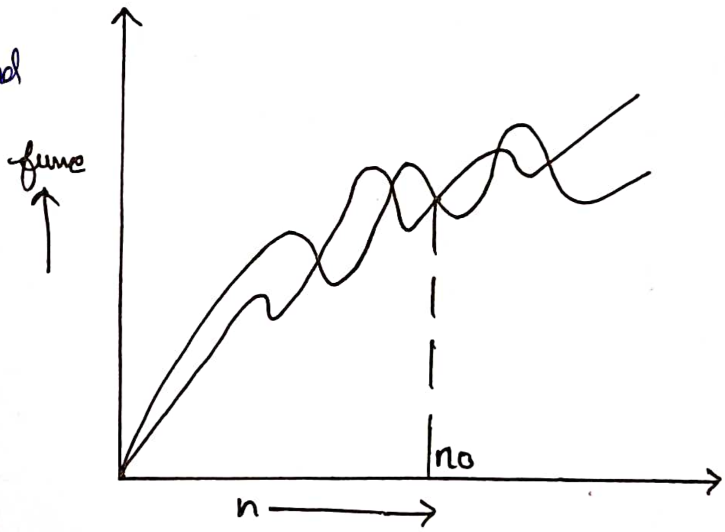
When $f(n) = o g(n)$ gives the upper bound

ie $f(n) = o g(n)$

if and only if

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

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



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

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

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

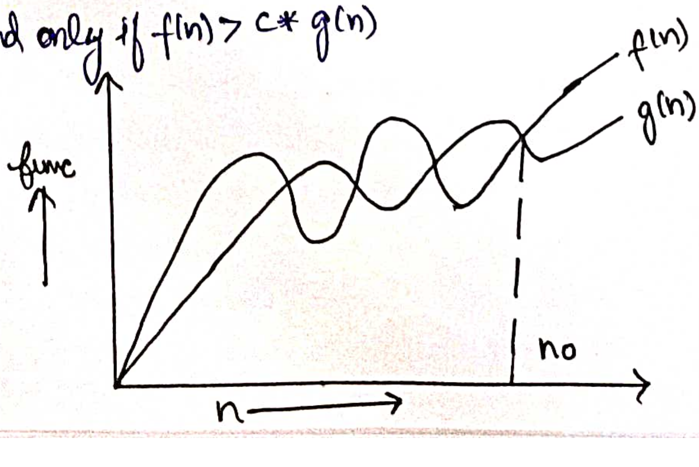
(v) Small Omega (ω)

It gives the 'lower bound' ie,

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

where g(n) is lower bound of f(n) if and only if $f(n) > c * g(n)$

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



Q.2 What should be time complexity of :

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

↳ for $i \Rightarrow 1, 2, 4, 6, 8, \ldots\ldots$ n times

ie series is a GP

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

K^{th} value of GP: $t_K = ar^{K-1}$

$t_K = 1(2)^{K-1}$

$2^{*}n = 2^K$

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

$\log_2 2 + \log_2 n = K$

$\log_2 n + 1 = K$ (Neglecting '1')

So, Time Complexity $T(n) \Rightarrow O(\log_2 n)$ Ans

Q3. $T(n) = \{ 3T(n-1)$ 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)

$T(n) = 27T(n-3)$ —(4)

Generalising series

$$T(n) = 3^K T(n-K) \text{ — } (5)$$

for K^{th} term, 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. $T(n) = \{ 2T(n-1) - 1$ if $n > 0$,
otherwise 1
}

$$T(n) = 2T(n-1) - 1 \longrightarrow (1)$$

put $n = n-1$

$$T(n-1) = 2T(n-2) - 1 \longrightarrow (2)$$

put in (1)

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

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

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

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

Put in (1)

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

Generalising series

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

→ $\underline{K^{th}\text{ term}}$ let $n-K=1$

$$K = n-1$$

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

$= 2^{K-1} - 2^{K-1}\left(\frac{1}{2} + \frac{1}{2^2} + \cdots + \frac{1}{2^{K-1}}\right)$

ie series is in GP.

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

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

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

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

$= 1$

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

Q.5. What should be time complexity of

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

→ $i = 1 \quad 2 \quad 3 \quad 4 \quad 5 \quad 6 \ldots$

$s = 1 + 3 + 6 + 10 + 15 + \cdots$

sum of $s = 1 + 3 + 6 + 10 + \cdots + n \rightarrow (1)$

Also $S = 1 + 3 + 6 + 10 + \cdots T_{n-1} + T_n \rightarrow (2)$

$0 = 1 + 2 + 3 + 4 + \cdots n - T_n$

$T_K = 1 + 2 + 3 + 4 + \cdots + K$

$T_K = \frac{1}{2} K(K+1)$

for K iterations

$$1+2+3+\dots K <= n$$

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

$$\frac{K^2+K}{2} <= n$$

$$O(K^2) \le n$$

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

$$\boxed{Tn = O(\sqrt{n})}$$ Ans

Q.6. Time Complexity of

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

↳ As $i^2 = n$

$$i = \sqrt{n}$$

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

$$\sum_{i=1} 1+2+3+4+\dots+\sqrt{n}$$

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

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

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

Q.7 Time Complexity of

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

↳ since, for $K = K^2$

$K = 1, 2, 4, 8, \dots n$

∵ series is in GP.

So, $a=1, r=2$

$$\frac{a(r^n-1)}{r-1}$$

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

$n = 2^K - 1$

$\Rightarrow n+1 = 2^K$

$\Rightarrow \log_2(n) = K$

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

$TC \Rightarrow O(n * \log n * \log n)$

$\Rightarrow O(n \log^2(n)) \longrightarrow$ Ans

Q.8 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 ~~for~~ j=n ~~tm~~ times every ~~time~~ turn

$\therefore i * j = n^2$

K^{th}; 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 $T(1) = 1;$

Now, substitute each value in $T(n)$

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

let $K^n - 3K = 1$

~~K~~ $K = (n-1)/3$ total ~~the~~ terms $= K+1$

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

$T(n) \simeq K n^2$

$T(n) \simeq (K-1)/3 * n^2$ $\therefore$ So, $\underline{T(n) = O(n^3)}$ 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 $i=1$ $\quad j = 1+2+\cdots \; (n \geq j+1)$

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

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

n^{th} term of AP is

$$T(n) = a + d*m$$

$$T(m) = 1 + d*m$$

$$(n-1)/d = n$$

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

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

$i = n-1$

we get,

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

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

$$= n + n/2 + n/3 + \cdots n/n-1 - n \times 1$$

$$= n\left[1 + \frac{1}{2} + \frac{1}{3} + \cdots \frac{1}{n-1}\right] - n*1$$

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

Since, $\int \frac{1}{x} = \log x$

$$T(n) = O(n \log n)$$

Ans

Q.10. For the function $n^{-1}k$ & 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 and c^n is

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

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

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

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

$\Rightarrow 1^K < a^2$

$\Rightarrow \underline{n_0 = 1 \text{ \& } c = 2}$ → Ans