1

Nancy Jaiswal
F
64
2016863

Tutorial - 1

Q1. What do understand by asymptotic notation. Define different Asymptotic notation with examples.

⇒ Asymptotic notations are used to tell the complexity of an algorithm when input is very high.

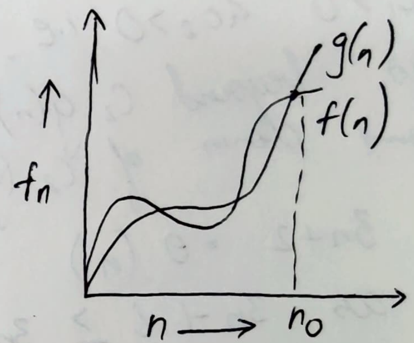
i) Big O(n)

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

if $f(n) \leq g(n) * C \quad \forall n \geq n_0$

for some constant, $C > 0$

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



eg. $f(n) = n^2 + n$

$g(n) = n^3$

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

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

ii) Big Omega (Ω)

When $f(n) = \Omega\, g(n)$ means $g(n)$ is "tight" lower bound of $f(n)$ i.e $f(n)$ can go beyond $g(n)$ i.e

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

iff $f(n) \geq c.g(n)$

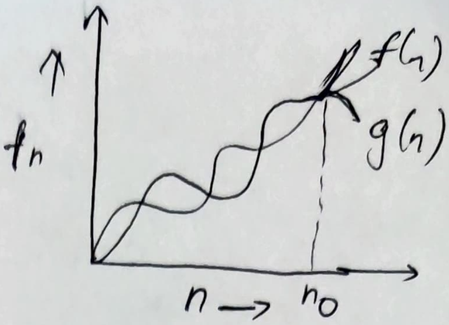
$\forall n_1 > n_0$ & c = constant > 0

eg. $f(n) = n^3 + 4n^2$

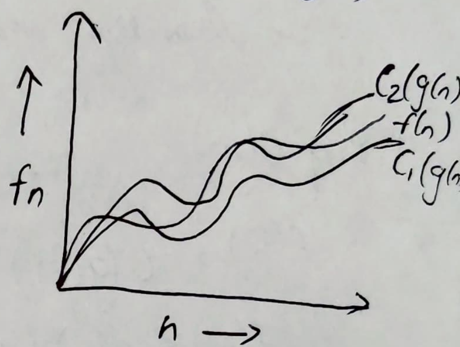
$g(n) = n^2$

i.e $f(n) \geq c * g(n)$

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



iii) Big Thetha (θ): When $f(n) = \theta(g(n))$ gives the tight upper bound & lower bound both i.e $f(n) = \theta(g(n))$ iff $c_1 * g(n_1) \leq f(n) \leq c_2 * g(n)$ $\forall$ $n \geq \max(n_1, n_2)$, some constant $c_1 > 0$ & $c_2 > 0$ i.e $f(n)$ can never go beyond $c_2\, g(n)$ & will never come down of $c_2(g(n))$



e.g $3n+2 = \theta(n)$

as $3n + i \geq 3n$ & $3n+2 \leq 4n$ for

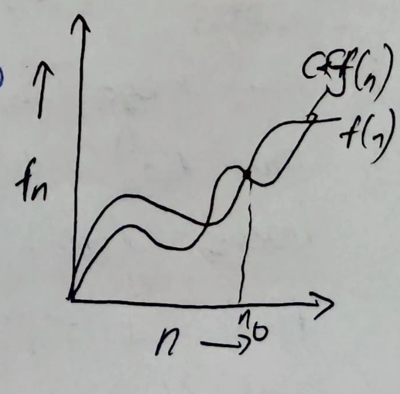
$n, c_1 = 3, \quad c_2 = 4$ & $n_0 = 2$

iv) Small oh(o):

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

$f(n) = o\, g(n)$ iff

$f(n) < c * g(n)$ $\forall$ $n > n_0$ & $n > 0$



eg $f(n) = n^2$, $g(n) = n^3$

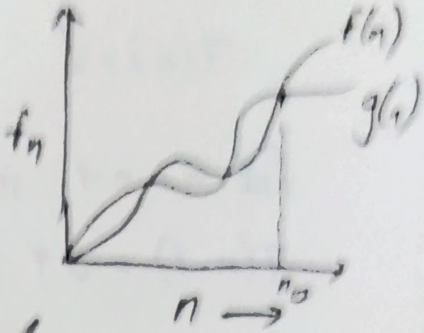
$f(n) < c * g(n)$

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

ii) Small Omega (ω):

It gives the 'lower bound' i.e $f(n) = \omega(g(n))$ where $g(n)$ is lower bound of $f(n)$

iff $f(n) > c * g(n)$ $\forall$ $n > n_0$ $\forall c > 0$



Q2. What should be time complexity of -

for (i=1 to n) {i = i * 2;}

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

for i = 1, 2, 4 ... n times
i.e series is a GP
So $a = 1$, $r = 2$

Kth value of GP

$T_k = a r^{k-1}$

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

$2n = 2^k$

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

$\log_2 2 + \log_2 n = k$

$\log_2 n + 1 = k$

So, time complexity $T(n) = O(\log_2 n)$

3. $T(n) = 3T(n-1)$ if $n > 0$ otherwise 1

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

$T(n) = 1$

Put $n = n-1$ in eq 1

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

Put value of $T(n-1)$ in eq. 1

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

$T(n) = 9T(n-2)$ - 3

Put $n = n-2$ in eq. 1

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

Put value of $T(n-2)$ in eq 3

$T(n) = 9(3T(n-3))$

$T(n) = 27T(n-3)$

⋮

So, $T(n) = 3^k T(n-k)$ - 5

for k^{th} term, let $n-k=1$ [base case]

$k = n-1$

Put $k = n-1$ in eq 5

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

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

$= 3^{n-1}$

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

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

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

$= 4T(n-2) - 2 - 1$

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

Put $n = n-2$ in eq. (1)

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

Put in eq (3)

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

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

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

Kth term => put $n-k = 1$

$k = n-1$

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

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

$\because$ a G.P

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

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

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

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

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

5. What should be time complexity of -

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

$i = 1, 2, 3, 4, 5 \ldots$

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

Sum of $s = 1 + 3 + 6 + 10 \ldots n$ - ①

Also $s = 1 + 3 + 6 + 10 + \ldots T_{n-1} + T_n$ - ②

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

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

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

for k, $1 + 2 + 3 + \ldots k <= n$

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

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

$O(k^2) \le n$

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

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

6. Time complexity of -

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

As $i^2 = n$

$i = \sqrt{n}$

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

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

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

$T(n) = O(n)$

7.
```
void function (int n){
  int i, j, k , count=0;
  for(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 $n = k^2$

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

$\therefore$ series is in G.P

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

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

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

$n = 2^k - 1$

$n + 1 = 2^k$

$\log_2(n) = k$

i	j	k
1	$\log(n)$	$\log(n) * \log(n)$
2	$\log(n)$	$\log(n) * \log(n)$
$\vdots$	$\vdots$	$\vdots$
n	$\log(n)$	$\log(n) * \log(n)$

$T.C = O(n * \log n * \log n)$

$= O(n \log^2(n))$

8. T.C of -

```
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 time

$\therefore i * j = n^2$

kth, now,

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

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

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

and $T(1) = 1$

now, put these values in $T(n)$

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

$$n - 3k = 1$$

$k = (n-1)/3$ total terms $= k+1$

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

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

$$T(n) \simeq (k-1)/3 \, n^2$$

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

9. T.C of -

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

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

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

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

n^{th} term of AP is:

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

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

$(n-1)/d = n$

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

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

we get,

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

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

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

$= n[1 + 1/2 + 1/3 \dots \frac{1}{n-1}] - n \times 1$

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

Since $f \frac{1}{n} = \log 4$

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

10. For the functions, n^k and c^n, what is the asymptotic relationship b/w these functions?
Assume that $k >= 1$ and $c > 1$ are constants. Find out the value of c and n_0 for which relation holds

As given n^k & 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 $n_0 = 1$; $c = 2$

$\Rightarrow 1^k = a^2$

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