Suryansh Goel
CST SPL02
44

Page No.
Date :

Design and Analysis Of Algorithms

TUTORIAL-01

Q1 Asymptotic notations are the mathematical notation used to describe the running time of an algorithm when the input tends towards a particular value or a limiting value.

There are mainly three asymptotic notations:-

- Big-O Notation - Gives the worst-case complexity
- Omega Notation - Gives the best-case complexity
- Theta Notation. - Gives the average case complexity.

For example:- In bubble sort, when the input array is already sorted, the time taken by the algorithm is linear, i.e, best-case.
But, when the same array is in reverse order, the time taken is quadratic, i.e, worst-case.

Q2 for (i=1 to n)
{ i=i*2; } // O(1)

$i = \overbrace{1, 2, 4, 8 \ldots\ldots n}^{K \text{ terms}} \Rightarrow G.P$

$a_k = ar^{(k-1)}$ $a=1$ $r=2$

$a_k = 1.2^{k-1}$ $a_k = n$

$n = 2^{k-1}$

taking log on both sides

$\log_2 n = k-1 \Rightarrow k = \log_2 n + 1$

$\therefore T(n) = O(\log_2 n + 1)$

removing constant terms

Ans $\Rightarrow T(n) = O(\log_2 n)$

Q3 $T(n) = \{3T(n-1)$ if $n>0$, otherwise $1\}$

$T(n) = 3T(n-1)$ — ① $T(0) = 1$ — ②

put $n=n-1$ in eq ①

$T(n-1) = 3T(n-1-1) = 3T(n-2)$ — ③

put value $T(n-1)$ from ③ to ①

$T(n) = 3[3T(n-2)] = 3^2 T(n-2)$ — ④

put $n=n-2$ in eq ①

$T(n-2) = 3T(n-2-1) = 3T(n-3)$ — ⑤

put value of $T(n-2)$ from ⑤ in ④

$T(n) = 3^2[3T(n-3)] = 3^3 T(n-3)$

Generalize,

$T(n) = 3^k T(n-k)$

Let $n-k=0$

$n=k$

$T(n) = 3^n T(0)$

as $T(0) = 1$ (from ②)

$T(n) = 3^n$

Page No.
Date :

$\boxed{T(n) = O(3^n)}$

Q4. $T(n) = \{ 2T(n-1)-1$ if $n>0$, otherwise $1 \}$

$T(n) = 2T(n-1) - 1$ — ① $T(0) = 1$ — ②

put $n = n-1$ in eq ①

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

put value $T(n-1)$ from ③ to ①

$T(n) = 2[2T(n-2) - 1] - 1 = 2^2(T(n-2)) - 2 - 1$ — ④

put value $n = n-2$ in eq ①

$T(n-2) = 2T(n-2-1) - 1 = 2T(n-3) - 1$ — ⑤

put value $T(n-2)$ from ⑤ to ①

$T(n) = 2^2[2T(n-3) - 1] - 2 - 1 = 2^3T(n-3) - 2^2 - 2^1 - 2^0$ — ⑥

Generalizing,

$T(n) = 2^k T(n-k) - 2^{k-1} - 2^{k-2} - 2^{k-3} - \dots - 2^0$

let $n-k = 0$

$k = n$

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

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

as $T(0) = 1$ from ②,

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

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

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

$T(n) = 1$

$\boxed{T(n) = O(1)}$

Q5

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

$i=1 \quad s=1$

$i=2 \quad s=3 \quad s=1+2$

$i=3 \quad s=6 \quad s=1+2+3$

$i=4 \quad s=10 \quad s=1+2+3+4$

$$S=1+2+3+4+----+k = \frac{k(k+1)}{2} > n \quad \{as\ s<=n\}$$

$$\frac{k^2+k}{2} > n$$

$$k > \sqrt{n}$$

Time complexity of the above fn : $\boxed{O(\sqrt{n}) = T(n)}$

Q6. void function (int n){
int i, count = 0;
for (i=1; i*i <= n; i++)
count++;   // O(1).

~~i = 1, 4, 4~~, $i = 1, 2, 3, \ldots n$

$i^2 = 1, 4, 9, \ldots$

so $i^2 <= n$ or $i <= \sqrt{n}$.

now $a_k = a + (k-1)d$

$a = 1, d = 1$

$a_k <= \sqrt{n}$

$\sqrt{n} = 1 + (k-1) \cdot 1$

$\sqrt{n} = k$

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

Q7

```
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++; }    //O(1)
      }
    }
}
```

i	j	k
$n/2$	$\log_2 n$	$(\log_2 n)^2$
$\frac{n}{2}+1$	$\log_2 n$	$(\log_2 n)^2$
⋮	⋮	⋮
n	$\log_2 n$	$(\log_2 n)^2$

$\left(\frac{n}{2}+1\right)$ times

$O(i*k) = O\left(\left(\frac{n}{2}+1\right)*(\log_2 n)^2\right)$ removing constants

$$\boxed{T(n) = O(n(\log_2 n)^2)}$$

Q8

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

$O(n^2)$

$T(n) = T(n-3) + n^2$ — ①

$T(1) = 1$

put $n = n-3$ in eq ①

$T(n-3) = T(n-6) + (n-3)^2$ — ②

put value of $T(n-3)$ from ② in ①

$T(n) = T(n-6) + n^2 + (n-3)^2$ — ③

put $n = n-6$ in eq ①

$T(n-6) = T(n-9) + (n-6)^2$ — ④

put value of $T(n-6)$ from ④ in ③

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

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

put $n-3k = 1$

$n = 1+3k \Rightarrow k = (n-1)/3$

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

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

$T(n) = cn^2 + k$ where c, k are constants

$\boxed{T(n) = O(n^2)}$ removing constants.

Q9 Time Complexity of :-

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

Ans for $i=1$, $j \rightarrow n$ times

for $i=2$, $j = 1+3+5+ \cdots n$

Using A.P,

$a=1, d=2, a_k = n \qquad a_k = a + (k-1)d$

$n = 1 + (k-1)\cdot 2$

$\frac{n-1}{2} = k-1 \Rightarrow k = \left(\frac{n+1}{2}\right)$ times

Similarly

for $i=3$, $j = 1, 4, 7, \ldots, n \Rightarrow \left(\frac{n+2}{3}\right)$ times

Generalising,

Time Complexity is :-

$$T(n) = n + \left(\frac{n+1}{2}\right) + \left(\frac{n+2}{3}\right) + \cdots + \left(\frac{n+k-1}{k}\right)$$

$$\sum_{k=1}^{n} \frac{n+k-1}{k} \Rightarrow \frac{n^2 + k(k+1)/2 - n}{k}$$

$$\Rightarrow \frac{n^2 + \frac{k^2 + k}{2} - n}{k}$$

removing constants & lower order terms

$$\boxed{T(n) = O(n^2)}$$