TCS 402

Assignment-01

1.) These notations are used to tell complexity of an algorithm, when input is very large.
Some examples of Asymptotic notations are:

- Big O notation: This notation represents the upper bound of an algorithm's growth rate. It represents the worst case scenario for the algorithm's performance. Ex: $O(\log n)$, $O(n)$, etc.

- Big Omega (Ω) notation: This notation describes the lower bound of an algorithm's growth rate. Represents the best case scenario for the algorithms performance. Ex: $\Omega(n)$, $\Omega(\log n)$, etc.

- Theta notation (Θ): This notation describes both the upper & lower bounds of an algorithm's growth rate, providing a tight bound on the growth rate.

  Ex: $\Theta(n)$, $\Theta(\log n)$, etc.

2.) $r = 2$

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

$n = 2^{k-1}$

$2^{k-1} = \log_2 n$

$\log_2$ on both sides,

$\log_2 2^{k-1} = \log_2 n$

$\Rightarrow k = \log n \Rightarrow O(\log n)$

$i =$ 1 2 4 8 16 32 n

3.)

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

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

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

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

On every step it get called by 3 times,

So, $T(n) = \underline{3^n}$.

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

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

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

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

$= 2^k T(n-k) - (2^{k-1} + 2^{k-2} + \ldots + 1)$

$T(0) = 1$;

$T(n) = 2^n - (2^{n-1} + 2^{n-2} + \ldots + 1)$.

$T(n) = 2^n - \frac{2(1-2^n)}{1-2^{-1}} = 2^n - 2 + 2^{1-n}$.

5) 1 3 6 10 15

i	s
1	1
2	3
3	6
4	10
5	15
:	:
n	n

$O(n) \rightarrow O(n)$

$O(n) = O(n)$.

6.)

$i = 1\ 2\ 3 \ldots n$

$i * i <= n;$

So, $O(\sqrt{n})$.

7.) $O(n \log^2 n)$ $\quad \underbrace{i++}_{n};\ \underbrace{j*2}_{\log_2 n};\ \underbrace{k*2}_{\log_2 n};$

8.) $O(n^2)$.

9.) $O(n \log n)$.

10) $n \log(c) = k \log(n)$

$\frac{\log(c)}{k} = \frac{\log(n)}{n}$

$$n^{\log c} = k^{\log n}$$

Since bases are same, exponents must be equal,

$\log c = k$ or for base = 10,

$c = 10^k$.