Asymptotic Notation :

→ Comparision of non-negative functions $f(n)$ and $g(n)$ for Large n value.

→ Asymptotic is a greek word = meaul = Too large

→ Growth rate comparision of $f(n)$, $g(n)$ for Large input is called asymptotic comparision of $f(n)$, $g(n)$

→ $f(n)$ and $g(n)$ Asymptotically equal if $\lim_{n \to \infty} \frac{f(n)}{g(n)} = \text{constant } (>0)$

eg. $f(n) = 10n^2 + 20$ $\quad g(n) = 5n^2 + 10n$

$$\lim_{n \to \infty} \frac{n^2 \left[10 + \frac{20}{n^2}\right]}{n^2 \left[5 + \frac{10}{n}\right]} = \frac{10}{5} = 2 \quad [\text{constant}]$$

Then we can say $f(n)$ $g(n)$ are asymptotically equal. Both are growing parallelly.

$f(n)$ and $g(n)$ Asymptotically not equal if $\lim_{n \to \infty} \frac{f(n)}{g(n)} = 0$ or ∞

$f(n) = 20n^2$ $\quad g(n) = 30n + 100$

$$\lim_{n \to \infty} \frac{20n^2}{n\left[30 + \frac{100}{n}\right]} = \frac{\infty}{30} = \infty$$

$f(n)$ and $g(n)$ are not equal.

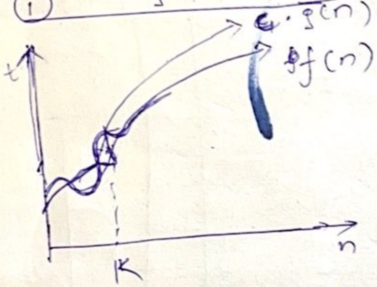
If $\lim_{n \to \infty} \frac{f(n)}{g(n)} = \infty$. $f(n)$ asymptotically bigger than $g(n)$. $f(n)$ grows faster than $g(n)$.

If $\lim_{n \to \infty} \frac{f(n)}{g(n)} = 0$ $g(n)$ asymptotically bigger than $f(n)$. $g(n)$ grows faster than $f(n)$.

... time comp... time complexity. Iteration, frequency

Mathematical way of representing time complexity of an algorithm we are analyzing without execution of an algorithm executed or calls itself again, all theses can be represented by Notations or no. of times function executed or calls itself again,

1 Big-oh (O)



(or) little o only <

→ worst case → Upper Bound (At most)
→ at max. → least upper bound.

n = input value, t is time

let say to solve a problem f(n) amount of time required. Then if we want to write f(n) in terms of O(?) then

$f(n) = O(g(n))$
$f(n) \le c \cdot g(n)$
$c > 0$
$n \ge K,\ K \ge 0$

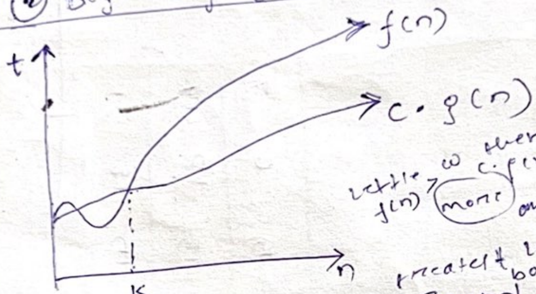
eg: $f(n) = 2n^2 + n$
$f(n) = O(?)$
$2n^2 + n \le c \cdot g(\ )$
quadratic (dominating) linear

Now upper bound of n^2 can be $n^2, n^3, n^4, 2^n, n^n$ etc. but choose least upper bound or closest value

$2n^2 + n \le 2 \cdot n^2$ $[\because c = 2]$
$2n^2 + n \le 3n^2$ $[n \le n^2$ or $1 \le n^2$

n^2
n

2 Big-Omega (Ω)



little ω then $f(n) >$ (more) $c \cdot f(n)$ only >

→ Best case → lower Bound (At least), minimum time, greatest lower bound.

$f(n) = \Omega(g(n))$
$f(n) \ge C \cdot g(n)$
$2n^2 + n \ge C \cdot n^2$ (nearby value)
$2n^2 + n \ge 2 \cdot n^2 + n$ $[\because c = 2]$
$c = 2$
$2n^2 + n \ge$
$\boxed{n \ge 0}$

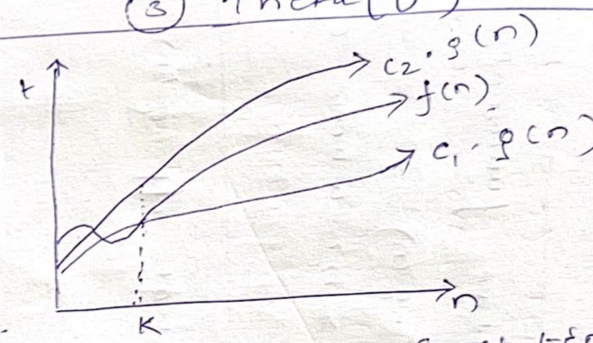
$2n^2 + n \ge 2n^2$

for $c = 5$ $55 \ge 50$

$\lim_{n \to \infty} \frac{f(n)}{g(n)} = \infty = f(n)$ grows faster than

← $\lim_{n \to \infty} \frac{f(n)}{g(n)} = 0$ $g(n)$ grows faster than $f(n)$

3 Theta (θ)



→ Average case → Exact time.

$c_1 \cdot g(n) \le f(n) \le c_2 \cdot g(n)$
$n^2 \le 2n^2 + n \le n^2$
$2n^2 \le 2n^2 + n \le 3n^2$
$2n^2 + n \ge cn^2$ $c = 2$
$2n^2 + n \ge \cancel{2n} c \cdot n^2$

choose carefully value of c.

c value can be two only or 1.

Big oh

$f(n)$ and $g(n)$ are non-negative functions.
$f(n) = O(g(n))$ if $f(n) <= c \cdot g(n)$
for all n value where $n \geq n_0$, where c, n_0 are constants.

$f(n) = O(g(n))$ means that iff $g(n)$ Asympt bigger or equal to $f(n)$.

eg

(1) $f(n) = 2n+3$
$g(n) = 3n$

n	0	1	2	3	4	5	6	7
$f(n) = 2n+3$	3	5	7	9	11	13	15	17
$c \cdot g(n) = 3 \cdot n$	0	3	6	9	12	15	18	21

growth rate bigger.

$f(n) = 2n+3$
$f(n) \leq g(n) \cdot c$
$2n+3 \leq c \cdot g(n)$
$\leq 3 \cdot g(n)$ ——— $n \geq 3$.

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

(2) $f(n) = 2n+3$
$g(n) = n^3$

n	0	1	2	3	4
f(n)	3	5	7	9	11
g.(n)	0	1	8	27	64

$2n+3 <= c.n^3$ for all n, $n \geq 2$

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

eg:

$\cancel{f(n) = 2n+3}$

$\cancel{g(n) = 2^{\log_{10} n}}$

order:

decreasing fun^c < constant < log function < polynomial < Exponential < exp. fun^c.

tion (imp.)

① $f(n) = n^{2.1}$ $\qquad g(n) = n^2 \log n$

which is true.

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

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

(c) a & b

(d) None.

$n^{2.1} > n^2 . \log n$

$n^2 : n^{0.1}$

② $f(n) = \begin{cases} 2^n & \text{for even } n \\ n & \text{,, odd } n \end{cases}$

$g(n) = \begin{cases} 2^n & \text{for odd } n \\ n & \text{for even } n \end{cases}$

which is true

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

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

(c). a & b.

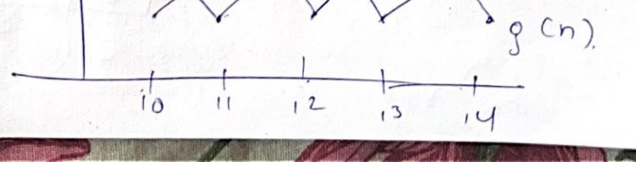
(d) None

soln

for even $f(n)$ is bigger (2^n)

,, ,, $g(n)$,, ,, (2^n)

so both $f(n)$ and $g(n)$ combinely they are non-comparable.



(12) $f(n) = n^{1+\sin n}$

$g(n) = n$

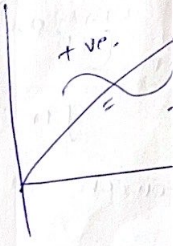
which is true.

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

(c) a & b (d) none.

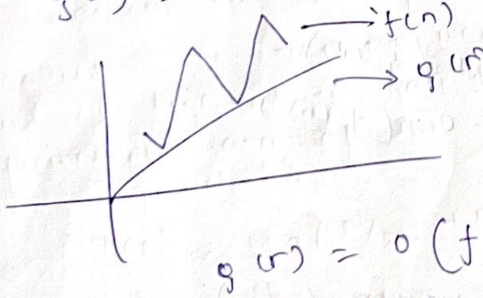
soln

	$n^{1+\sin n}$	n
0	n	n
90	$n^{1+1} = n^2 >$	n
180	$n^{1+0} = n$	n
270	$n^{1-1} = n^0 <$	n
360	n $=$	n



Here $f(n)$ and $g(n)$ are non-comparable. bco
sometimes $f(n)$ bigger other time $g(n)$. thou
$\sin n$ (it ranges between +ve to -ve)

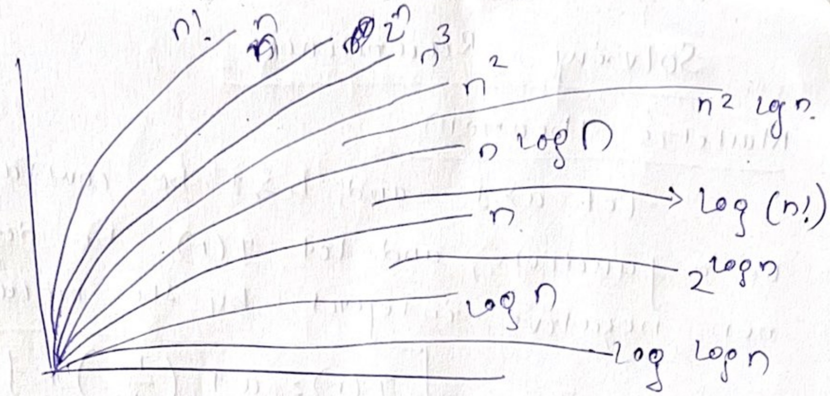
(13) $f(n) = n^{2+\sin n}$, $g(n) = n$ } Here $f(n)$ is bigger

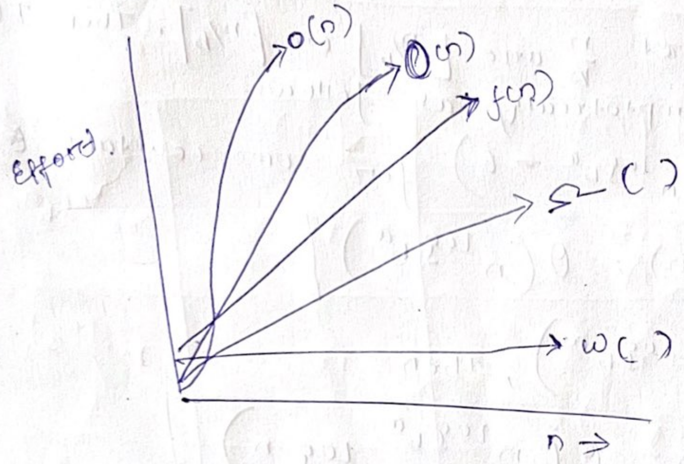


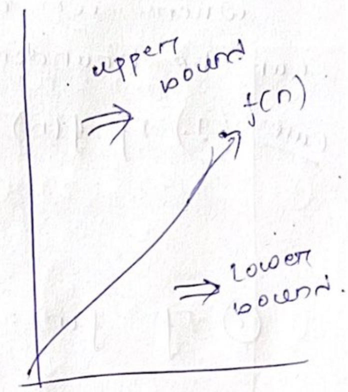
$g(n) = O(f(n))$

(14) $n!$, 2^n, n^n, $n^{\sqrt{n}}$ (all are exponential

$\boxed{\begin{array}{l} \text{I.M.P} \\ 2^n < 3^n \quad [\text{power same, base same}] \\ n^n > n^{\sqrt{n}} \end{array}}$







$n! > 4^n > 2^n > n^2 > n(\log n) > \log(n!) > n$

Solving Recurrences

Master's Theorem

Let $a \geq 1$ and $b \geq 1$ be constants, let $f(n)$ be a function, and let $T(n)$ be defined on the non negative integers by the recurrence

$$\boxed{T(n) = aT\left(\frac{n}{b}\right) + f(n)} \quad \text{—— (1)}$$

where we interpret $\frac{n}{b}$ are $\lfloor n/b \rfloor$ or $\lceil n/b \rceil$. Then $T(n)$ can be bounded asymptotically as follows-

case (1.) If $f(n) = O(n^{\log_b a - \epsilon})$ for some constant $\epsilon > 0$

$$\boxed{T(n) = \theta(n^{\log_b a})}$$

(2) If $f(n) = \theta(n^{\log_b a})$, Then

$$\boxed{T(n) = \theta(n^{\log_b a} \log n)}$$

(3) If $f(n) = \Omega(n^{\log_b a + \epsilon})$ for some constant $\epsilon > 0$ and if $f(n/b) \leq cf(n)$ for some constant $c < 1$ and all sufficiently large n, then

$$\boxed{T(n) = \theta(f(n))}$$

Here, one can see that $f(n)$ is compared with $n^{\log_b a}$ and solution to recurrence is obtained by selecting maximum of two as in case 1 and case 3. In case 2 the two functions are of the same size, so we multiply by logarithmic function.

ex (1) $T(n) = 4T(n/2) + n$

sol^n Given that;

$$T(n) = 4T(n/2) + n$$

compare the relation with equation 1.

Now we get. $a = 4, \; b = 2, \; f(n) = n$

Is $f(n) = n = O(n^{\log_2 4 - \epsilon})$

$n = O(n^{2-\epsilon}) = ?$ (Ans Yes)

Yes.

then case 1 is applied.

$$T(n) = \theta\left(n^{\log_b a}\right) = \theta\left(n^{\log_b a}\right)$$

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

Exp.2 $T(n) = 4T(n/2) + n^2$

Soln: $a = 4$, $b = 2$, $f(n) = n^2$

Is $n^2 = O\left(n^{\log_2 4 - \epsilon}\right)$ NO, if $\epsilon > 0$, but it is true only if $\epsilon = 0$

Is $n^2 \neq O(n^{2-\epsilon})$

$n^2 =$

then case 2 of Master's theorem is applied.

Hence, $T(n) = \theta\left(n^{\log_b a} \cdot \log n\right)$

$$\boxed{T(n) = \theta(n^2 \cdot \log n)}$$

exmp.3

$T(n) = 7T(n/2) + n^2$

Soln $a = 7$, $b = 2$, $f(n) = n^2$

$n^2 = O\left(n^{\log_b a - \epsilon}\right)$

Is $n^2 = O\left(n^{\log_2 7 - \epsilon}\right)$ Yes becau $\log_2 7 = 2.81$.

$n^2 = O\left(n^{2.81 - \epsilon}\right)$

Yes so case 1 is applied.

$$T(n) = \theta\left(n^{2.81}\right)$$

exm.4

$8T(n/2) + n^2$

$f(n) = n^2 = O\left(n^{\log_2 8 - \epsilon}\right)$

$n^2 = O\left(n^{3-\epsilon}\right)$

Yes so

ex. 5

$f(n) = 3T(n/2) + n^2$

$\log_b a = \log_2 3 = 1.62$

Soln

$n^2 = \Omega(n^{1.62+\epsilon})$

So case-3

$\Omega(n^{1.62+\epsilon})$

Then $T(n) = \Theta(n^2)$

	$T(n) = aT(n/b) + f(n)$	$f(n)$	$n^{\log_b a}$
1	$T(n) = 2T(n/2) + \sqrt{n}$ Soln $= \Theta(n^1)$	$\sqrt{n}$ (0.5)	$n^{\log_b a} = n$
2	$T(n) = T(n/2) + \sqrt{n}$ Soln $= \Theta(\sqrt{n})$	$\sqrt{n}$	$n^{\log_2 1} = n$
3	$T(n) = 8T(n/4) + n^4$ $\Theta(n^4)$	n^4	$n^{\log_4 8} = n^3$
4	$T(n) = 8T(n/2) + n^2$ $\Theta(n^3)$	n^2	$n^{\log_2 8} = n^3$
5	$T(n) = 8T(n/2) + n^3$ $\Theta(n^3 \log n)$	n^3	n^3
6	$T(n) = T(n/2) + 1$ $\Theta(\log_b n)$	n^0	$n^{\log_2 1} = n^0$
7	$T(n) = 26T(n/3) + n^3$ Soln $\Theta(n^3)$	n^3	$n^{\log_3 26} \simeq n^{2.9}$
8	$T(n) = 26T(n/3) + n^2$ $\Theta(n^{\log_3 26})$	n^2	$n^{2.9}$
9	$T(n) = 2T(n/2) + n \log n$	$n \log n$	$n^{\log_2 2} = n^1$

Here $f(n)$ and $n^{\log_b a}$ are asymptotically not equal. so case 2 not applicable. but polynomially equal. so case 1 and case 3 not applicable. so it cannot be solved by master method. Use substitution method

(10)

$T(n) = 2T(n/2) + \frac{n}{\log n}$ — sol^n $\theta(n \log n \log n)$

$T(n) = 4T(n/2) + n^2 (\log n)^2$

$T(n) = 4T(n/2) + n^2 \log n$

$\theta(n^2 \log^2 n)$

I.M.P Some books also gives the master theorem in the following format.

$$T(n) = aT(n/b) + n^{\log_b a} \cdot \log_b^K n$$

then $\boxed{\theta(n^{\log_b a} \cdot \log^{K+1} n)}$