Types of time functions / complexity

- $O(1)$ — constant $\left.\begin{array}{l} f(n)=2 \\ f(n)=5 \end{array}\right\} O(1)$ → order of 1.
- $O(\log n)$ — logarithmic
- $O(n)$ — linear

$$\left.\begin{array}{l} f(n) = 6n+3 \\ f(n) = 700n + 200 \\ f(n) = \frac{n}{300} + 4 \end{array}\right\} O(n)$$

degree of polynomial $= 1$

- $O(n^2)$ — quadratic
- $O(n^3)$ — cubic
- $O(2^n)$ — Exponential $O(3^n)$ $O(n^n)$

These are all classes of functions.

Classes of functions in increasing order of weightage.

$$1 < \log n < \sqrt{n} < n < n\log n < n^2 < n^3 < \text{---}$$

$$2^n < 3^n \text{---} < n^n$$

→ Comparison of classes of functions.

Scanned By Scanner Go

Ex

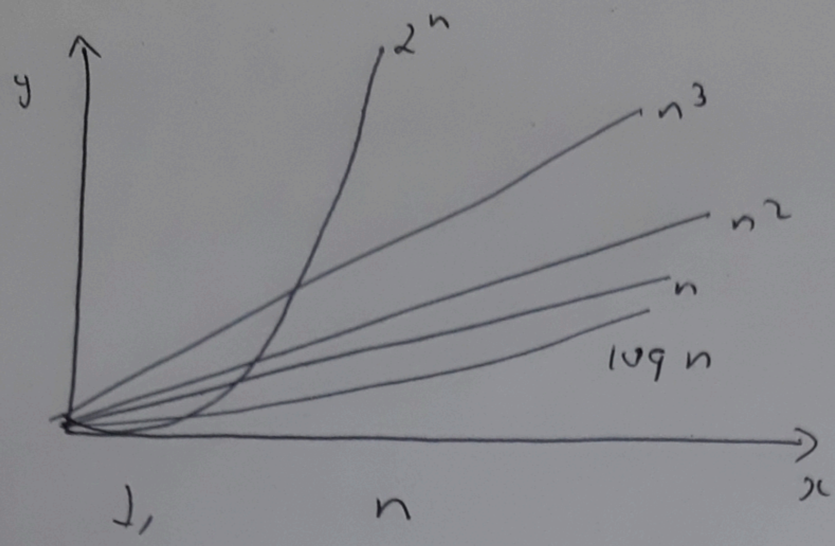
	n value	log n	n	n^2	2^n
1	1	0	1	1	2
2	2	$\log_2 2 = 1$	2	4	4
3	4	2	4	16	16
4	8	$\log_2 8 = 3$	8	64	256
5	9	$\log_2 9 = 3.1$	9	81	512

- As we go on increasing n log n growth < n < n^2 < 2^n.
- Also until pt. no 3 values were similar. From pt. 4 2^n is moving at faster pace
- Even $n^{100} < 2^n$

  $n^k < 2^n$.

  Don't assume n as small no.

  $n \longrightarrow \infty$. For larger values of n, at some point $n^{100} < 2^n$. And from that point 2^n will be greater always.



Initially smaller for some values of n but eventually it will be larger.

Explanation

Ex

$f(n) = 2n + 3$ — find $O(f(n))$ – Big Oh.

$2n+3 \leq$ ______ something should be written which is greater than $f(n)$.

We can write anything on RHS. But take care that the func is having 2 terms on LHS. Don't write multiple terms on RHS. Write only single term. It can have any ~~coefficient~~ coefficient that satisfies condition of notations.

So $\underset{f(n)}{2n+3} \leq \underset{c}{10}\,\underset{g(n)}{n}$ True $n \geq 1$

$\therefore f(n) = O(n)$

$2n+3 \leq \cancel{5n} \text{ or } 7n$ All true.

Soln to get value → make all terms equal (to highest exp. value of n)

so $2n+3 \Rightarrow 2n+3n = 5n$

so $2n+3 \leq 5n$

$n \geq 1$ $\therefore f(n) = O(n)$

Also

$2n+3 \le 2n^2 + 3n^2$

$2n+3 \le 5n^2$ Also true

so $f(n) = O(n^2)$

$1 < \log n < \sqrt{n} < n < n\log n < n^2 < n^3 < \cdots$
$2^n < 3^n < \cdots < n^n$.

- Seeing above f(n) belongs to class of functions "n". So all the functions including n and to RHS will be upper bound of f(n).
- All to LHS including n will become lower bound of f(n).
- n → will be avg. bound.
- So for $f(n) = 2n+3$.

$O(n)$
$O(n^2)$ } all true
$O(2^n)$

But $f(n) = O(\log n)$ false.

- Try to write the closest function. Though you can write higher value func. also but not useful.

Omega notation — lower bound.

The function $f(n) = \Omega g(n)$ iff $\exists$ +ve constant c and n_0

s.t. $f(n) \geq c * g(n) \; \forall \; n \geq n_0$.

So $f(n) = 2n+3 \geq \underset{\downarrow c}{1}\,\underset{\downarrow g(n)}{n} \; \forall \; n \geq 1$

$\therefore$ $f(n) = \Omega(n)$

Also $2n+3 \geq 1 * \log n \; \forall \; n \geq 1$

$f(n) = \Omega(\log n)$

But $f(n) = \Omega(n^2)$ X wrong.

Nearest func. is useful.

So best is $f(n) =$ ~~Ωg~~ $\Omega(n)$

Theta notation – avg case.

- The func. $f(n) = \theta g(n)$ iff $\exists$ +ve constant c_1, c_2 and n_0

s.t. $c_1 g(n) \leq f(n) \leq c_2 * g(n)$

eg $f(n) = 2n+3$

$$\underbrace{1}_{c_1} * n \leq 2n+3 \leq \underbrace{5}_{c_2} * n$$ (Already done)

$\therefore f(n) = \theta(n)$

- $f(n) = \theta(n^2)$ X wrong
- $f(n) = \theta(\log n)$ X wrong.
- we have to write similar on LHS and RHS as θ notation gives exact time complexity. (mostly recommended). If it is not possible to find exact notation for class of func, then use Big Oh or Omega notation.

Ex

$1 < \log n < \sqrt{n} < n < n\log n < n^2 < n^3 \ldots$

$\ldots < 2^n < 3^n < n^n.$

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

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

$3n^2 + 3n + 2 \leq \underbrace{8}_{c}\,\underbrace{n^2}_{g(n)}$ $\quad n \geq 1$

$\therefore f(n) = O(n^2)$ $\quad$ f of n is Big Oh of n^2.

→

$3n^2 + 3n + 2 \geq 1 * n^2$

$\therefore f(n) = \Omega(n^2)$

→ $1 * n^2 \leq \cancel{2n^2 + 3n}\ 3n^2 + 3n + 2 \leq 8n^2$

$\therefore f(n) = \Theta(n^2)$

(2) $f(n) = n^2 \log n + n$

$1 * n^2 \log n \leq n^2 \log n + n \leq 10 n^2 \log n$

- (Both sides write $n^2 \log n$)

So $f(n) = O(n^2 \log n)$
$f(n) = \Omega(n^2 \log n)$
$f(n) = \theta(n^2 \log n)$

- $n^2 \log n$ comes in between n^2 and n^3.
Combination of n^2 and $\log n$.
This is also a class. $n^2 \log n$

(3) $f(n) = n!$

It is ~~eg~~ $n! = 1 * 2 * 3 * --- (n-2) * (n-1) * n$

As a practice we make both sides as n.

$$1\times1\times1\times1--1 \le 1\times2\times3---\times n \le n\times n\times n\times--\times n$$

$$1 \le n! \le n^n$$

$$f(n) = O(n^n)$$
$$f(n) = \Omega(1)$$

Here we cannot find tight bound or avg bound or Θ bound. exactly

$$n^{10} \le n! < n^{11}$$ { we cannot say this always }

So when we cannot mention Θ bound exactly. Then upper and lower bounds are useful.

Q $f(n) = \log n!$

$$\log(1\times1--\times1) \le \log(1\times2\times---n) \le \log(n\times n\times--\cdot n)$$

↓

0 so write 1

$$1 \le \log(1\times2\times--n) \le \log(n^n)$$
$$= n\log n)$$

- For factorial func again we define only upper bound and lower bound as it is difficult to find avg bound.

So $f(n) = O(n\log n)$

$f(n) = \Omega(1)$

Properties of Asymptotic notations

General properties –

(1) If $f(n) = O(g(n))$ then

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

eg: $f(n) = 3n^2 + 6$ is $O(n^2)$

then $7 \cdot f(n) = 7(3n^2 + 6)$

$= 21n^2 + 42$

$\in O(n^2)$

It is true for Ω also. Also for Θ notation.

(2) Reflexive property -

If $f(n)$ is given then $f(n)$ is $O(f(n))$

A function is upper bound of itself.

eg: $f(n) = n^2$ then $f(n) = O(n^2)$

It is applicable to Ω also.

(3) Transitive property -

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

and $g(n) = O(h(n))$

then $f(n) = O(h(n))$

eg If $f(n) = n$ $g(n) = n^2$

$h(n) = n^3$

then n is $O(n^2)$ n^2 is $O(n^3)$

then n is $O(n^3)$

Applicable to all 3 notations

④ **Symmetric property**

if $f(n) = \Theta(g(n))$

then $g(n) = \Theta(f(n))$

This property holds true only for Θ notation.

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

then $f(n) = \Theta(n^2)$

$g(n) = \Theta(n^2)$

⑤ **Transpose symmetric** — True for Ω and O notation.

if $f(n) = O(g(n))$ then $g(n) = \Omega(f(n))$

eg: if $f(n) = n$ $g(n) = n^2$

then n is $O(n^2)$ — upper bound.

and n^2 is $\Omega(n)$ — lower bound

⑥ If $f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$

when same func is both upper bound and lower bound then it is Θ bound.

ie if $g(n) \leq f(n) \leq g(n)$ then $f(n) = \Theta(g(n))$

⑦ If $f(n) = O(g(n))$
and $d(n) = O(e(n))$
then $f(n) + d(n) = O(\max(g(n), e(n)))$

eg if $f(n) = n$ $O(n)$
$d(n) = n^2$ $O(n^2)$

$f(n) + d(n) = n + n^2$
$= O(n^2)$

Also if $f(n) = n^2$ $O(n^2)$
$d(n) = n$ $O(n)$

then $f(n) + d(n) = n^2 + n = O(n^2)$

⑧ If $f(n) = O(g(n))$ and
$d(n) = O(e(n))$
then $f(n) * d(n) = O(g(n) * e(n))$

eg If $f(n) = O(n)$
$d(n) = O(n^2)$ then
$f(n) * d(n) = O(n^3)$