→ ASYMPTOTIC NOTATIONS

→ Asymptotic Notations give us an idea about how good a given algorithm is compared to some other algorithm.

→ Let us see the mathematical definition of 'order of' now.

→ Primarily there are three types of widely used asymptotic notations.

(1) Big OH Notation (O)
(2) Big OMEGA Notation (Ω)
(3) Big THETA Notation (θ) → widely used one!

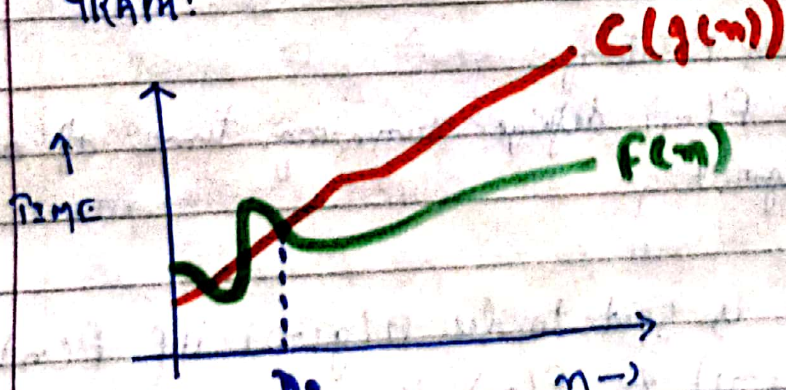
(1) Big OH Notation :
→ Big oh notation is used to describe asymptotic upper bound.
→ Mathematically, if $f(n)$ describes running time of an algorithm; $f(n)$ is $O(g(n))$ iff there exist positive constants C & n_0 such that :

$$\boxed{0 \leq f(n) \leq c(g(n)) \quad \text{for all } n \geq n_0}$$

⇓
used to give upper bound on a function.

→ If a function is $O(n)$, it is automatically $O(n^2)$ as well!

GRAPH:



(2) Big Omega Notation :
→ Just like O notation provides an asymptotic upper bound. Ω notation

provides asymptotic lower bound. Let $F(n)$ define running time of an algorithm;

→ $F(x)$ is said to be $\Omega(g(n))$ if there exists positive constants C & n_0 such that:

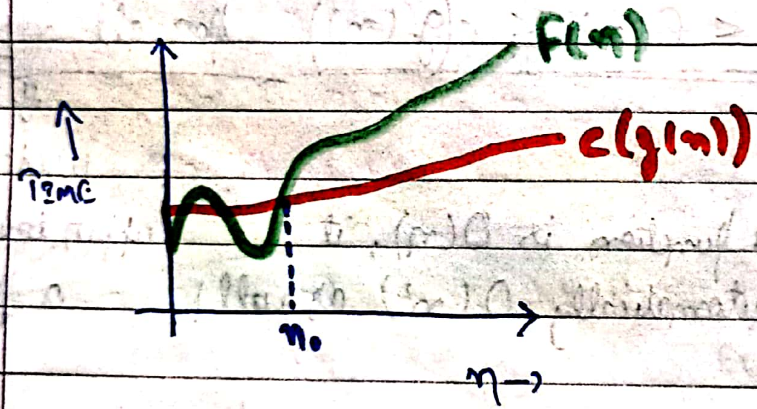
$$\boxed{0 \le c(g(n)) \le F(n) \quad \text{for all } n \ge n_0}$$

⇓

used to give lower bound on a function.

→ If a function is $\Omega(n^2)$ it is automatically $\Omega(n)$ as well.

GRAPH:



(3) Big THETA Notation:

→ Let $F(n)$ define running time of an algorithm.

→ $F(n)$ is said to be $\Theta(g(n))$ iff $F(n)$ is $O(g(n))$ & $F(n)$ is $\Omega(g(n))$.

→ Mathematically:

$$0 \le F(n) \le c_1(g(n)) \quad \forall\ n \ge n_0$$

&

$$0 \le C_2(g(n)) \le F(n) \quad \forall\ n \ge n_0$$

→ sufficiently large value of n.

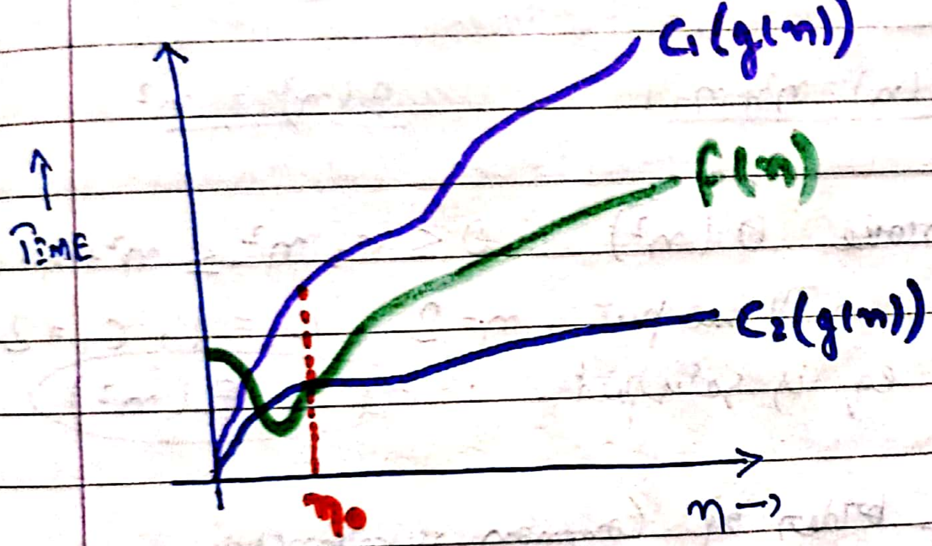
Date

→ Merging both the equations, we get:

$$\boxed{0 \leq C_2(g(n)) \leq F(n) \leq C_1(g(n)) \quad \forall \; n \geq n_0}$$

→ The equation simply means there exists positive constants C_1 & C_2. Such that $F(n)$ is sandwitched b/w $C_2(g(n))$ & $C_1(g(n))$.

GRAPH:



→ Which one of these to use?

→ Since Big theta gives a better picture of runtime for a given algorithm, most of the interviewers expect you to provide an answer in terms of Big theta when they say 'order of'.

→ Quick Quiz: Prove that n^2+n+1 is $O(n^3)$, $\Omega(n^2)$ & $\Theta(n^2)$ using respective definitions:

Page No.
Date

Ans → $f(n) = n^2 + n + 1$ $\qquad g(n) = n^3$

→ Prove $O(n^3)$: $0 \le n^2 + n + 1 \le c\, n^3$

→ If we put, $n = 2$ & $c = 1$,
equation is satisfied. ∴ It is $O(n^3)$

$f(n) = n^2 + n + 1$ $\qquad g(n) = n^2$

→ Prove $\Omega(n^2)$: $0 \le c\, n^2 \le n^2 + n + 1$

→ If we put, $n = 1$ & $c = 1$,
eq. is satisfied. ∴ It is $\Omega(n^2)$

$f(n) = n^2 + n + 1$ $\qquad g(n) = n^2$

→ Prove $\Theta(n^2)$: $0 \le c_2 n^2 \le n^2 + n + 1 \le c_1 n^2$

→ If we put, $n = 2$, $c_2 = 1$, $c_1 = 2$,
eq. is satisfied. ∴ It is $\Theta(n^2)$

→ ↑sing order of Common runtimes:

$1 < \log(n) < n < n \log n < n^2 < n^3 < 2^n < n^n$

BETTER - - - - ~~COMMON RUNTIMES~~ → WORSE

→ FROM BEST TO WORSE →

Scanned with CamScanner