Theta, Omega Complexities

By

Arun Cyril Jose

Omega notation (Ω)

- f(n) = Ω(g(n)) iff there exists a positive constant ‘c’ and ‘n_0’ such that, $0 <= c\ g(n) <= f(n)$ for all $n >= n_0$

Running Time

- These Bounds are for **algorithms** rather than **programs.**
  - Programs are just implementations of an algorithm, and almost all always the details of the program do not affect the bounds.

- These Bounds are for **algorithms** rather than **problems.**
  - A problem can be solved with several algorithms, some more efficient than others.

Running Time

$O(Log(N))$	10^{-7} seconds
$O(N)$	10^{-6} seconds
$O(N*Log(N))$	10^{-5} seconds
$O(N^2)$	10^{-4} seconds
$O(N^6)$	3 minutes
$O(2^{(N)})$	10^{14} years.
$O(N!)$	10^{142} years.

Algorithm Analysis

- Worst Case
  - Upper bound on running time.
  - No matter what the inputs are, algorithm would not run longer than this.
- Best Case
  - Lower bound on running time.
  - Input is the one for which the algorithm runs the fastest.

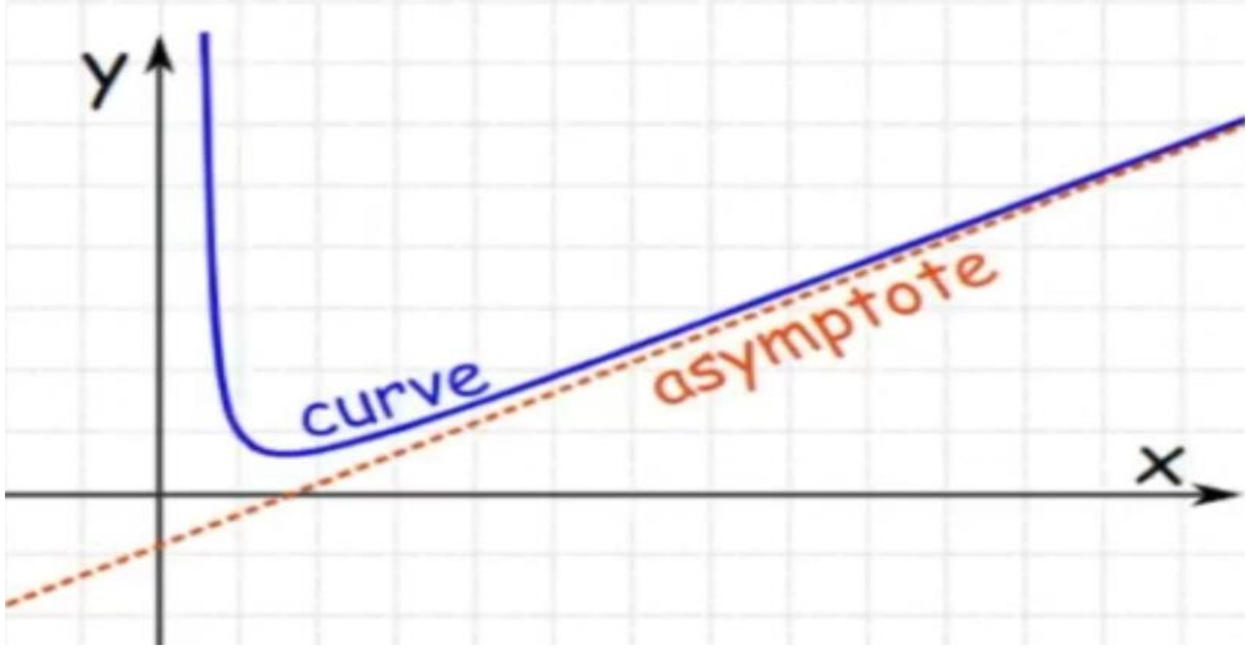
$$\boxed{\textit{Lower Bound} \leq \textit{Running Time} \leq \textit{Upper Bound}}$$

- Average Case
  - A prediction about the running time.
  - Assuming the input to be random.

Algorithm Analysis

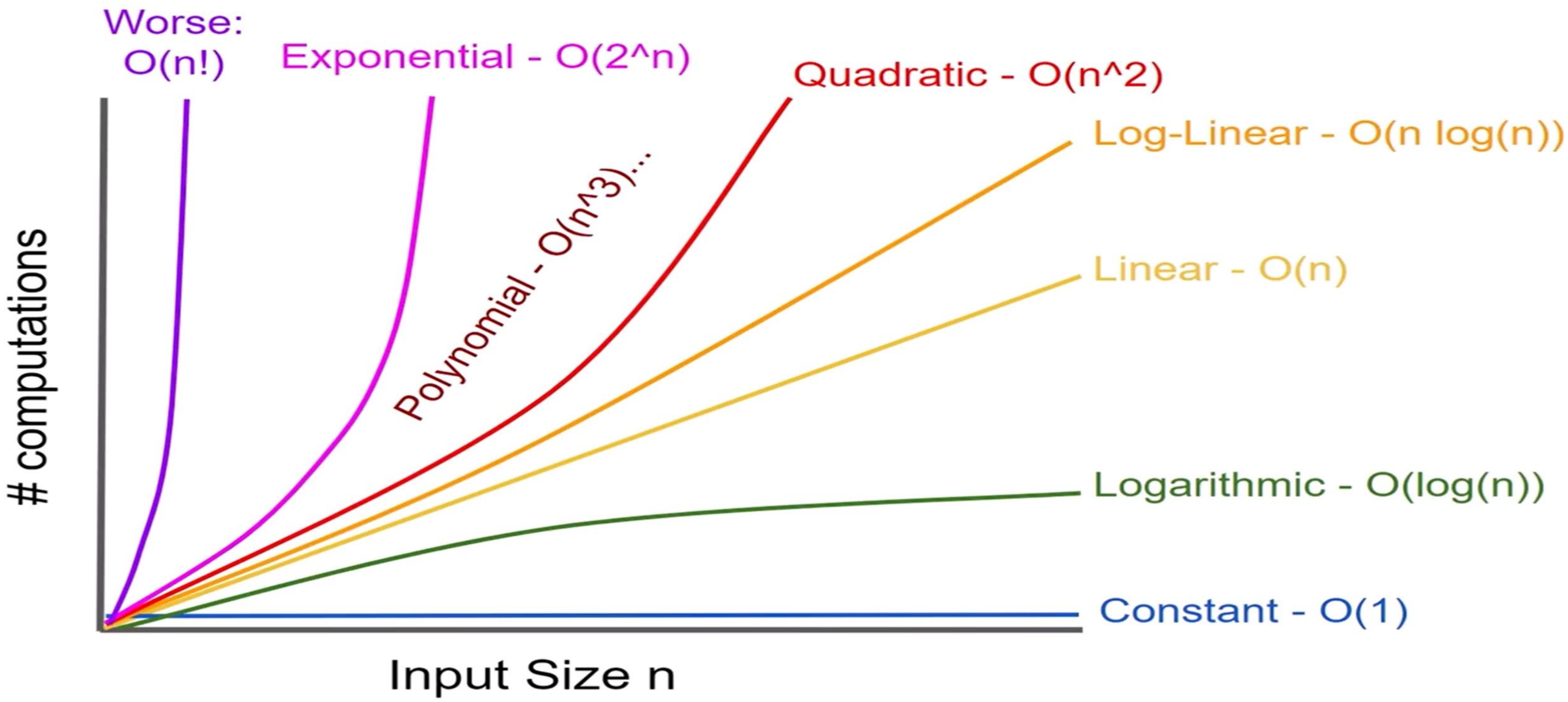
- Express running time as a function of input size *n* (i.e. f(n)).
- Compare different functions corresponding to running time.
- Such an analysis is independent of machine's processing power, programming language etc.

Asymptotic Analysis



- **Asymptote** is a line that a curve approaches as it heads towards infinity.

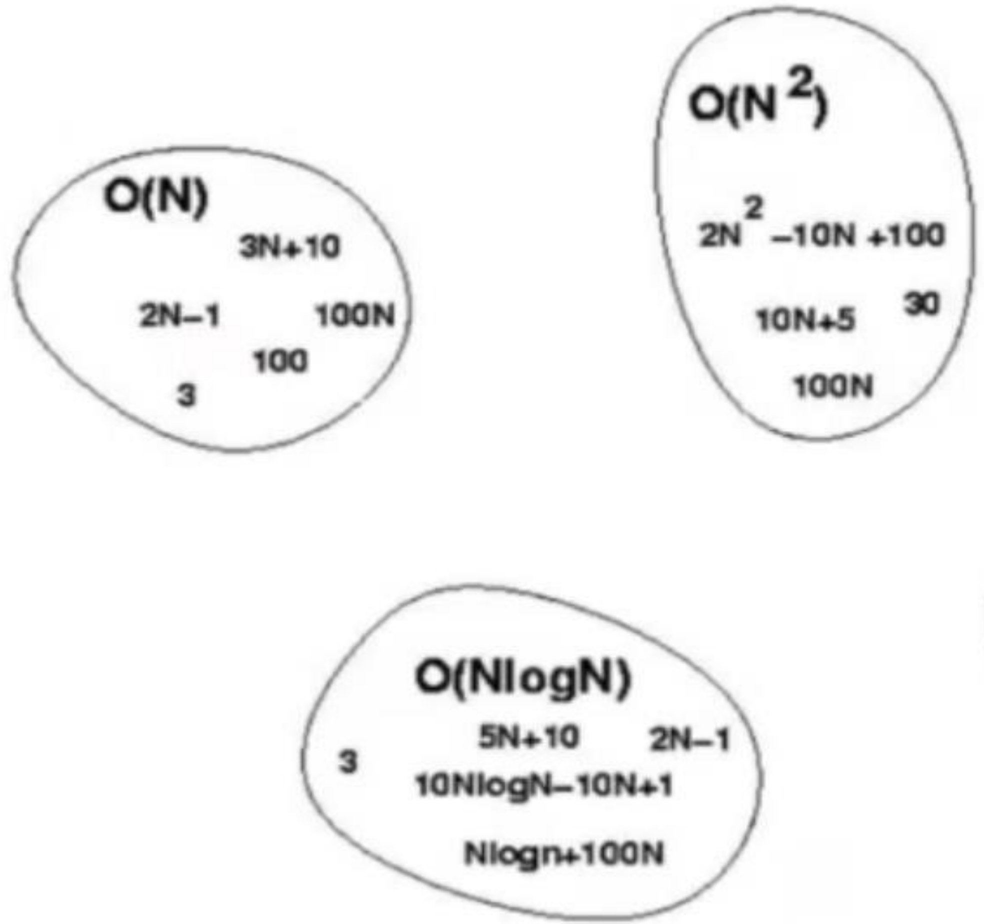
Running Time



Asymptotic Analysis

- Running time of an algorithm as input size approaches infinity is called the *asymptotic running time*.
- We look at **tight** bound, **upper** bounds and **lower** bounds.
- Growth rate for an algorithm is the rate at which the cost of the algorithm grows as the input size increases.

Asymptotic Analysis



$O(g(n))$ is the set of functions with smaller or same order of growth as $g(n)$

Examples

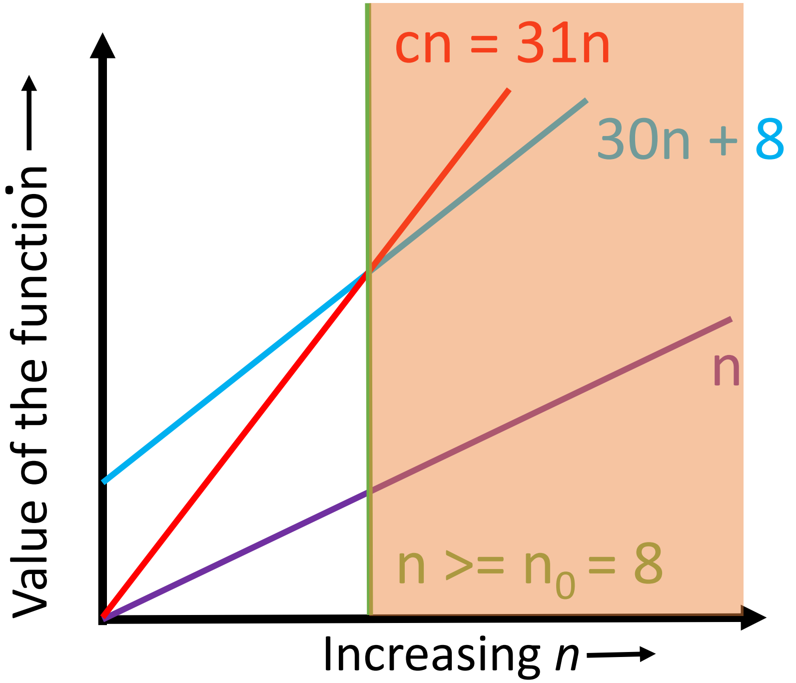
- $2n^2 = O(n^3)$: $2n^2 <= cn^3 \Rightarrow 2 <= cn \Rightarrow c = 1$ and $n_0 = 2$
- $n^2 = O(n^2)$: $n^2 <= cn^2 \Rightarrow 1 <= c \Rightarrow c > 1 \Rightarrow c = 1$ and $n_0 = 1$
- $1000n^2 + 1000n = O(n^2)$: $1000n^2 + 1000n <= 1001n^2$

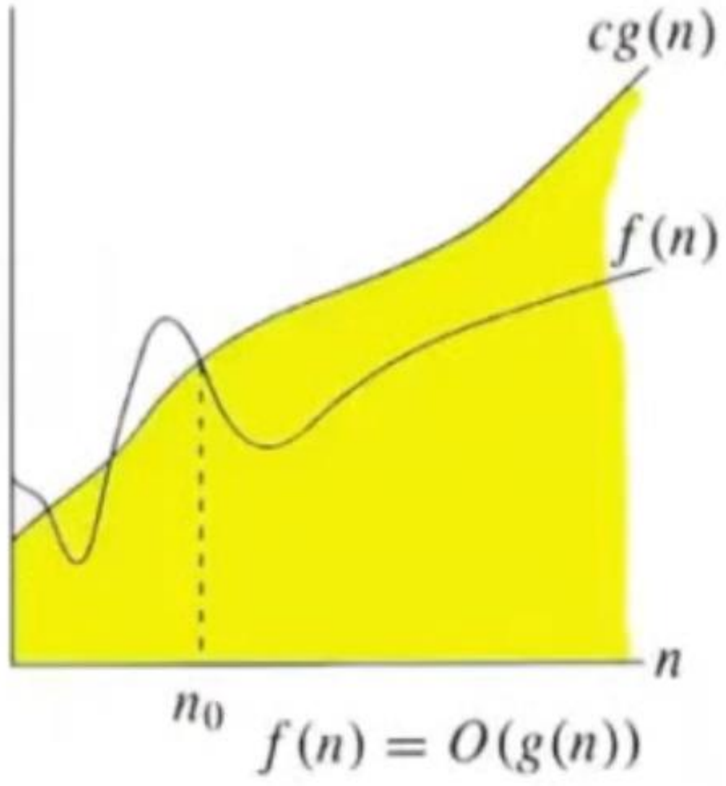
$\Rightarrow 1000n <= n^2 \Rightarrow 1000 <= n \Rightarrow c = 1001$ and $n_0 = 1000$

- $n = O(n^2)$: $n <= cn^2 \Rightarrow 1 <= cn \Rightarrow cn >= 1 \Rightarrow c = 1$ and $n_0 = 1$

Examples

- Show that 30n + 8 is O(n)
  - Show, $\exists c, n_0$: $30n + 8 \le cn \; \forall \, n > n_0$
- 30n + 8 is not less than n anywhere (n >0)
- It isn't even less than 31n everywhere.
- It is less than 31n everywhere to the right of n = 8.





Examples

- There are no unique values for c and n_0 in proving the asymptotic bounds.
- Prove that $100n + 5 = O(n^2)$

  $100n + 5 <= 101n^2$

  c = 101 and n_0 = 5 is one solution.
- $100n + 5 <= 105n^2$

  c = 105 and n_0 = 1 is one solution.
- Find **some** constant c and n_0 that satisfies the asymptotic notation relation.

Examples Omega notation (Ω)

- $5n^2 = \Omega(n)$

  $\exists c, n_0$ such that $0 <= cn <= 5n^2 \; \forall \; n > n_0$

  $c = 1$ and $n_0 = 1$

- **$10n + 5 \neq \Omega(n^2)$**

- n^3 **$= \Omega(n^2)$**
- n **$= \Omega(\log n)$**

Theta notation (Θ)

- $f(n) = \Theta(g(n))$ iff $f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$
  - Most closest Tight Bound.
- **iff there exists a positive constant ‘c_1’, ‘c_2’ and ‘n_0’ such that,**

  $c_1\, g(n) <= f(n) <= c_2\, g(n)$ for all $n >= n_0$: also, $c_1, c_2 > 0$ and $n_0 > 0$

- **$0 <= c_1\, g(n) <= f(n) <= c_2\, g(n)$**
- $f(n) = 2n + 3$
- $g_1(n) = 3n$
- $g_2(n) = n$