

DATA STRUCTURES - ASYMPTOTIC ANALYSIS

http://www.tutorialspoint.com/data_structures_algorithms/asymptotic_analysis.htm Copyright © tutorialspoint.com

Asymptotic analysis of an algorithm, refers to defining the mathematical boundation/framing of its run-time performance. Using asymptotic analysis, we can very well conclude the best case, average case and worst case scenario of an algorithm.

Asymptotic analysis are input bound i.e., if there's no input to the algorithm it is concluded to work in a constant time. Other than the "input" all other factors are considered constant.

Asymptotic analysis refers to computing the running time of any operation in mathematical units of computation. For example, running time of one operation is computed as $f(n)$ and may be for another operation it is computed as $g(n^2)$. Which means first operation running time will increase linearly with the increase in n and running time of second operation will increase exponentially when n increases. Similarly the running time of both operations will be nearly same if n is significantly small.

Usually, time required by an algorithm falls under three types –

- **Best Case** – Minimum time required for program execution.
- **Average Case** – Average time required for program execution.
- **Worst Case** – Maximum time required for program execution.

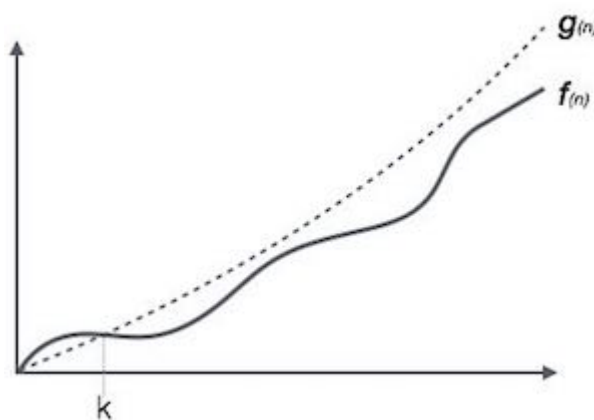
Asymptotic Notations

Following are commonly used asymptotic notations used in calculating running time complexity of an algorithm.

- O Notation
- Ω Notation
- θ Notation

Big Oh Notation, O

The $O(n)$ is the formal way to express the upper bound of an algorithm's running time. It measures the worst case time complexity or longest amount of time an algorithm can possibly take to complete.

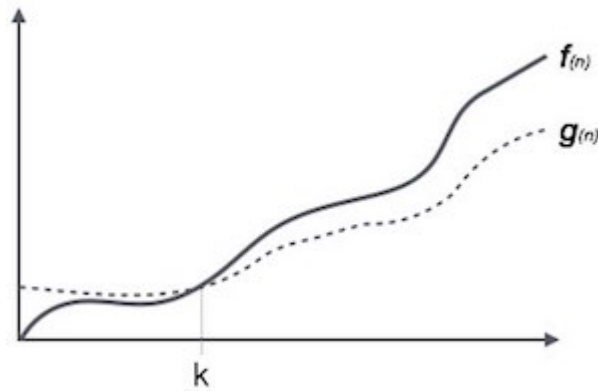


For example, for a function $f(n)$

$$O(f(n)) = \{ g(n) : \text{there exists } c > 0 \text{ and } n_0 \text{ such that } g(n) \leq c \cdot f(n) \text{ for all } n > n_0. \}$$

Omega Notation, Ω

The Ω_n is the formal way to express the lower bound of an algorithm's running time. It measures the best case time complexity or best amount of time an algorithm can possibly take to complete.

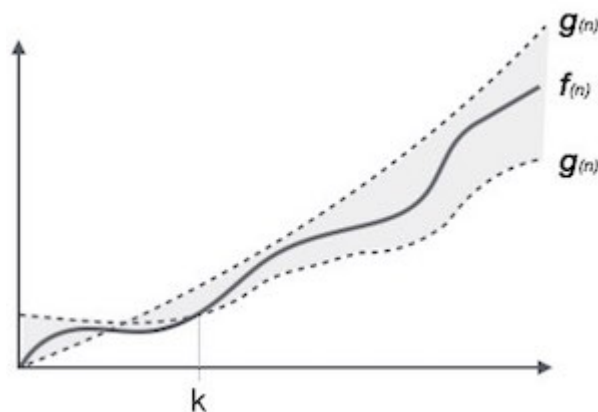


For example, for a function f_n

$$\Omega(f(n)) \geq \{ g(n) : \text{there exists } c > 0 \text{ and } n_0 \text{ such that } g(n) \leq c \cdot f(n) \text{ for all } n > n_0. \}$$

Theta Notation, θ

The θ_n is the formal way to express both the lower bound and upper bound of an algorithm's running time. It is represented as following –



$$\theta(f(n)) = \{ g(n) \text{ if and only if } g(n) = O(f(n)) \text{ and } g(n) = \Omega(f(n)) \text{ for all } n > n_0. \}$$

Common Asymptotic Notations

constant	–	$O(1)$
logarithmic	–	$O(\log n)$
linear	–	$O(n)$
n log n	–	$O(n \log n)$
quadratic	–	$O(n^2)$
cubic	–	$O(n^3)$
polynomial	–	$n^{O(1)}$
exponential	–	$2^{O(n)}$