Asymptotic Notations

Overview:

The efficiency of an algorithm depends on the amount of time, storage and other resources required to execute the algorithm. The efficiency is measured with the help of asymptotic notations.

An algorithm may not have the same performance for different types of inputs. With the increase in the input size, the performance will change.

The study of change in performance of the algorithm with the change in the order of the input size is defined as asymptotic analysis.

What are Asymptotic notations?

Asymptotic notations are the mathematical notations used to describe the running time of an algorithm when the input tends towards a particular value or a limiting value.

For example: In bubble sort, when the input array is already sorted, the time taken by the algorithm is linear i.e. the best case.

But, when the input array is in reverse condition, the algorithm takes the maximum time (quadratic) to sort the elements i.e. the worst case.

When the input array is neither sorted nor in reverse order, then it takes average time. These durations are denoted using asymptotic notations.

There are mainly three asymptotic notations:

Big-O notation

Omega notation

Theta notation

Big-O Notation (O-notation)

Big O notation is an asymptotic notation that measures the performance of an algorithm by simply providing the order of growth of the function.

This notation provides an upper bound on a function which ensures that the function never grows faster than the upper bound. So, it gives the least upper bound on a function so that the function never grows faster than this upper bound.

It is the formal way to express the upper boundary of an algorithm running time. It measures the worst case of time complexity or the algorithm's longest amount of time to complete its operation.

Omega Notation (Ω-notation)

It basically describes the best-case scenario which is opposite to the big o notation.

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

It determines what is the fastest time that an algorithm can run.

If we required that an algorithm takes at least certain amount of time without using an upper bound, we use big- Ω notation i.e. the Greek letter "omega". It is used to bound the growth of running time for large input size.

Theta Notation (Θ-notation)

The theta notation mainly describes the average case scenarios.

It represents the realistic time complexity of an algorithm. Every time, an algorithm does not perform worst or best, in real-world problems, algorithms mainly fluctuate between the worst-case and best-case, and this gives us the average case of the algorithm.

Big theta is mainly used when the value of worst-case and the best-case is same.

It is the formal way to express both the upper bound and lower bound of an algorithm running time.

Little o Notation (o Notation)

There are some other notations present except the Big-Oh, Big-Omega and Big-Theta notations. The little o notation is one of them.

Little o notation is used to describe an upper bound that cannot be tight. In other words, loose upper bound of f(n).

Little omega (ω Notation)

Another asymptotic notation is little omega notation. it is denoted by (ω).

Little omega (ω) notation is used to describe a loose lower bound of f(n).