Asymptotic notation is a mathematical notation used in computer science to describe the behavior of functions as the input size approaches infinity.

It provides a way to classify and compare the growth rates of functions and algorithms, focusing on the dominant terms that affect the overall complexity.

The most commonly used asymptotic notations are:

1. Big O notation (O):

It represents the upper bound or worst-case scenario of the growth rate of a function. For a given function f(n), O(g(n)) represents an upper bound on f(n), indicating that f(n) grows no faster than g(n) asymptotically. It provides an upper limit on the running time or space requirements of an algorithm.

2. Omega notation (Ω):

It represents the lower bound or best-case scenario of the growth rate of a function. For a given function f(n), Ω(g(n)) represents a lower bound on f(n), indicating that f(n) grows at least as fast as g(n) asymptotically. It provides a lower limit on the running time or space requirements of an algorithm.

3. Theta notation (Θ):

It represents both the upper and lower bounds of the growth rate of a function. For a given function f(n), Θ(g(n)) represents a tight bound on f(n), indicating that f(n) grows at the same rate as g(n) asymptotically. It provides a precise estimate of the running time or space requirements of an algorithm.

These notations allow us to classify and compare the efficiency or complexity of algorithms by focusing on their growth rates rather than the exact running times, which can vary depending on hardware and specific input instances.

For example, if we have an algorithm with a running time of $O(n^2)$, it means that the algorithm's worst-case time complexity grows quadratically with the input size. If another algorithm has a running time of O(n log n), it grows more efficiently than the previous algorithm since n log n has a lower growth rate than n^2.

Asymptotic notation helps in making informed decisions about algorithm selection, analyzing algorithmic efficiency, and understanding the scalability of algorithms with large input sizes. It provides a standardized and abstract way to express and compare algorithmic complexity.