Time Complexity Analysis

Prepared By,
Saima Siddique Tashfia
Lecturer, Department of CSE
University of Information Technology & Sciences

Understanding Time Complexity

Time complexity is a concept in computer science that deals with the quantification of the amount of time taken by a set of code or algorithm to process or run as a function of the amount of input. In other words, the ***time complexity is how long a program takes to process a given input.*** The efficiency of an algorithm depends on two parameters:

- Time Complexity
- Space Complexity

Time Complexity: It is defined as the number of times a particular instruction set is executed rather than the total time taken. It is because the total time taken also depends on some external factors like the compiler used, the processor's speed, etc.

Space Complexity: It is the total memory space required by the program for its execution.

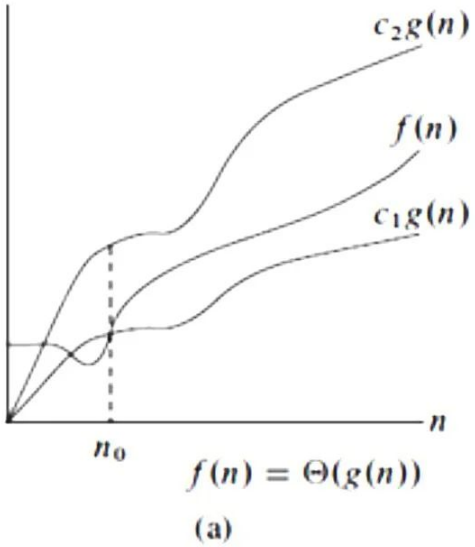
Asymptotic Analysis

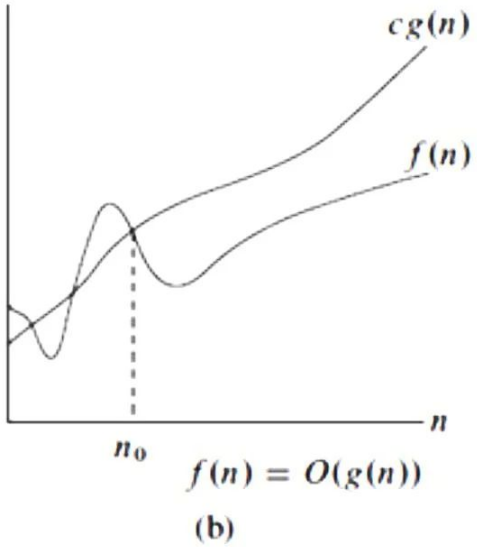
Asymptotic notations are the mathematical notations used to describe the running time of an algorithm when the input tends towards a particular value or a limiting value.

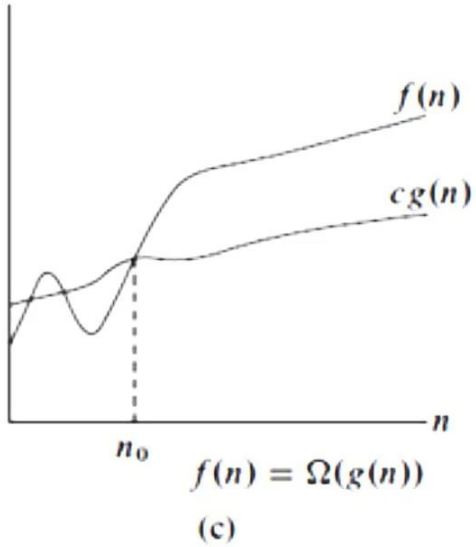
Three asymptotic notations

Notation	Meaning	Example
Big O (O)	It represents the upper bound or the worst case scenario of an algorithm. It tells us how much time an algorithm can take at most for any input size.	**The worst case time of bubble sort is O(n^2).**
Big Omega (Ω)	It represents the lower bound or the best case scenario of an algorithm. It tells us how much time an algorithm can take at least for any input size.	**The best case time of binary search is Ω(log n).**
Big Theta (Θ)	It represents the tight bound or the average case scenario of an algorithm. It tells us how much time an algorithm can take on average for any input size.	**The average case time of quick sort is Θ(n log n).**

All Three Notations







Theta | **Big O** | **Omega**