ALGORITHMIC EFFICIENCY

Prof. K. Adisesha

BE, M.Sc., M.Th., NET, (Ph.D.)

Learning objectives

- Introduction
- Algorithm Definition
- What is computational Complexity
- Estimating Complexity of Algorithm
- Asymptotic Notations
- Complexity of an Algorithm

Algorithm

What is an Algorithm?

- An algorithm is a step-by-step procedure for solving a problem in a finite amount of time.
- The word algorithm comes from the name of a Persian Mathematician Abu Ja'far Mohammed ibn-I Musa al Khowarizmi.
- For a given problem:
  - There can be more than one solution (more than one algorithm) .
  - An algorithm can be implemented using different programming languages on different platforms.

Algorithm

Designing of an Algorithm

- Design of an algorithm is an area of computer science which minimizes the cost.
- Always design algorithms which minimize the cost.

Analysis of Algorithm

- Analysis of Algorithms is the area of computer science that provides tools to analyze the efficiency of different methods of solutions.
- In short predict the cost of an algorithm in terms of resources and performance is called analysis of Algorithm.

Properties of Algorithm

Donald Ervin Knuth has given a list of five properties for an algorithm, these properties are:

1. FINITENESS
2. DEFINITENESS
3. INPUT
4. OUTPUT
5. EFFECTIVENESS

Properties of Algorithm

FINITENESS

- An algorithm must always terminate after a finite number of steps.
- It means after every step one reach closer to solution of the problem and after a finite number of steps algorithm reaches to an end point.

DEFINITENESS

- Each step of an algorithm must be precisely defined.
- It is done by well thought actions to be performed at each step of the algorithm.
- Also the actions are defined unambiguously for each activity in the algorithm.

INPUT

- Any operation you perform need some beginning value/ quantities associated with different activities in the operation.
- So the value/quantities are given to the algorithm before it begins.

Properties of Algorithm

OUTPUT

- One always expects output/result (expected value/quantities) in terms of output from an algorithm.
- The result may be obtained at different stages of the algorithm.
- Result is obtained from the intermediate stage of the operation then it is known as intermediate result
- Result obtained at the end of algorithm is known as end result.
- The output is expected value/quantities always have a specified relation to the inputs.

EFFECTIVENESS

- Algorithms to be developed/written using basic operations.
- Algorithms operations should be done exactly and in a finite amount of time by a person, by using paper and pencil only.

Efficiency of Algorithm

Performance of an algorithm depends on many factors:

- **Internal Factors:** Specify algorithm's efficiency in terms of
  - Time required to run
  - Space (Memory)required to run

- **External Factors**: affect the algorithm's performance
  - Size of the input to the algorithm
  - Speed of computer on which it is run
  - Quality of the Computer

Internal Factors

There are two aspects of algorithmic performance or efficiency:

- **TIME COMPLEXITY:** It is essentially efficiency, or how long a program function takes to process a given input.
  - Instructions take time.
  - How fast does the algorithm perform?
  - What affects its runtime?

- **SPACE COMPLEXITY:** of an algorithm is total space taken by the algorithm with respect to the input size. Space complexity includes both Auxiliary space and space used by input.
  - Data structures take space
  - What kind of data structures can be used?
  - How does choice of data structure affect the runtime?

Asymptotic Analysis

Asymptotic Analysis:

- In Asymptotic Analysis, we evaluate the performance of an algorithm in terms of input size
- Asymptotic analysis of an algorithm refers to defining the mathematical framing of its run-time performance.
- Usually, the 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 Analysis

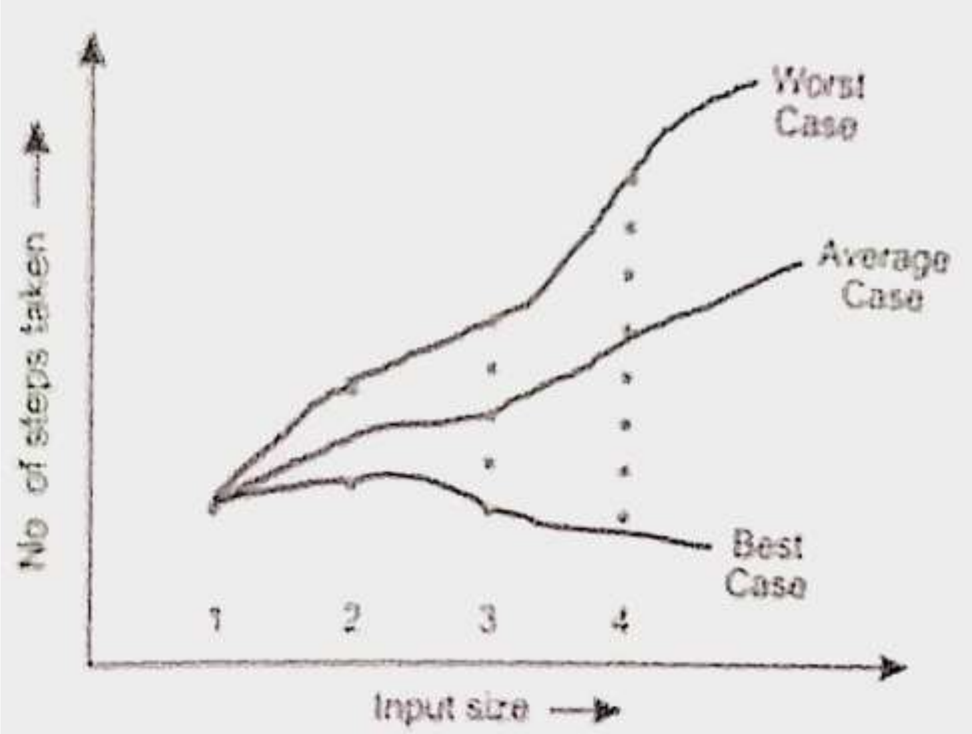
Asymptotic Analysis:

- Following are the commonly used asymptotic notations to calculate the running time complexity of an algorithm.

Asymptotic Analysis

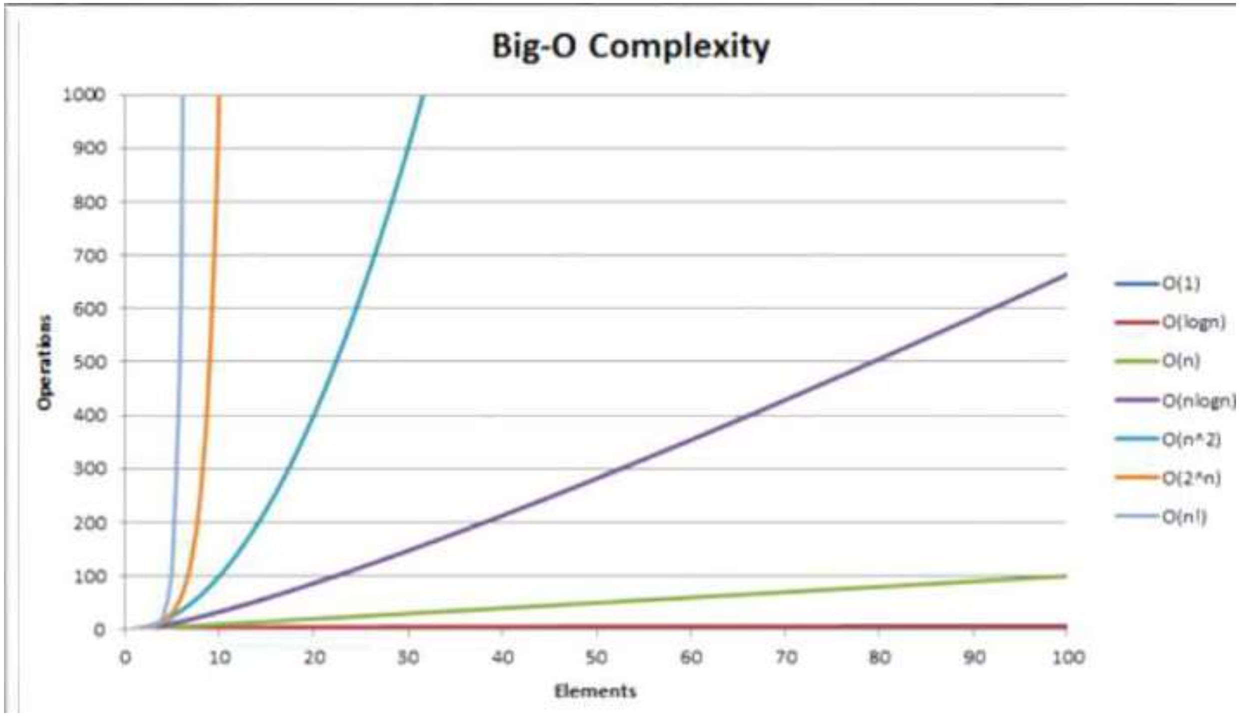
Asymptotic Analysis:

- Following graph is commonly used to calculate the running time complexity of an algorithm.



O Notation (Big Oh Notation)

- Big O specifically describes the worst-case scenario, and can be used to describe the execution time required or the space used (e.g. in memory or on disk) by an algorithm.
- Big O complexity can be visualized with this graph:



Omega Notation - Ω

- Big Ω describes the set of all algorithms that run no better than a certain speed (it's a lower bound)
- It measures the best case time complexity or the best amount of time an algorithm can possibly take to complete.
- Best case performance of an algorithm given function g(n), we denote by Ω(g(n)) the set of functions.

  $$\Omega(g(n)) = \{f(n): \text{there exist positive constants } c \text{ and } n0 \text{ such that } 0 <= c*g(n) <= f(n) \text{ for all } n >= n0\}.$$

- Best case performance of an algorithm is generally not useful, the Omega notation is the least used notation among all three.

Theta Notation - θ

θ notation

- You can use the big-Theta notation to describe the average-case complexity.
- The θ notation describes asymptotic tight bounds

DEF. Big Theta. $f(n)$ is $\Theta(g(n))$ iff $\exists$ positive real constants C_1 and C_2 and a positive integer n_0, such that

$$C_1 g(n) \leq f(n) \leq C_2 g(n) \quad \forall\, n \geq n_0$$

- If an algorithm has the average-case time complexity of,

  say, 3*n^2 - 5n + 13 ,

  then it is true that its average-case time complexity is

  Theta(n^2) , O(n^2) , and O(n^3)

TIME COMPLEXITY

TIME COMPLEXITY OF SORTING ALGORITHMS

Algorithm	Time Complexity		
	Best	Average	Worst
Selection Sort	Ω(n^2)	θ(n^2)	O(n^2)
Bubble Sort	Ω(n)	θ(n^2)	O(n^2)
Insertion Sort	Ω(n)	θ(n^2)	O(n^2)
Heap Sort	Ω(n log(n))	θ(n log(n))	O(n log(n))
Quick Sort	Ω(n log(n))	θ(n log(n))	O(n^2)
Merge Sort	Ω(n log(n))	θ(n log(n))	O(n log(n))
Bucket Sort	Ω(n+k)	θ(n+k)	O(n^2)
Radix Sort	Ω(nk)	θ(nk)	O(nk)

TIME COMPLEXITY OF SEARCHING ALGORITHMS

Algorithm	Best case	Average	Worst case
Linear search	O(1)	O(N)	O(N)
Binary search	O(1)	O(log N)	O(log N)

SPACE COMPLEXITY

SPACE COMPLEXITY OF SEARCHING AND SORTING ALGORITHMS

Algorithm	Space complexity
Selection sort	O(1)
Merge sort	O(N)
Linear search	O(1)
Binary search	O(1)

Conclusion!

- We learned about the Algorithm Definition
- What is computational Complexity
- Estimating Complexity of Algorithm
- Asymptotic Notations
- Complexity of an Algorithm

Thank you