Introduction to Algorithms

By: Yehia M. Abu Eita

Outlines

- Introduction
- Algorithms advantages and disadvantages
- Space and time complexities
- Some algorithms applications
- Algorithm lifecycle

Introduction

- Algorithms are a **finite** series of well-defined instructions to **solve** a **problem** or to **make computations**.
- In order for some instructions to be an algorithm, it must be **clear**, have **well-defined inputs** and **outputs**, **finite**, **feasible**, and **language independent**.
- **Searching** and **sorting** algorithms are most widely used in many applications.

Algorithms advantages and disadvantages

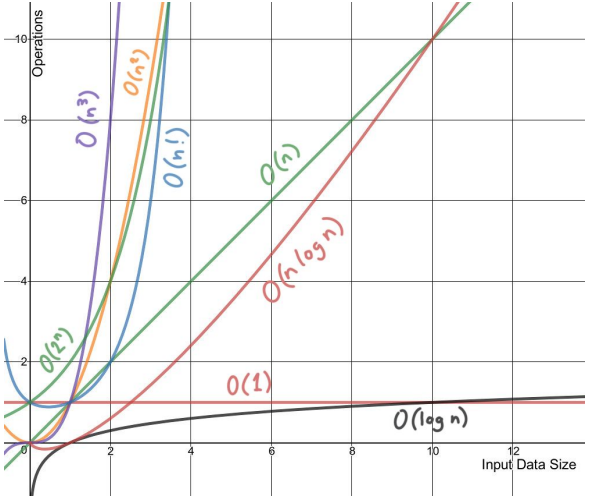
- **Advantages**:
  - It is easy to understand.
  - Algorithm is a step-wise representation of a solution to a given problem.
  - In Algorithm the problem is broken down into smaller pieces or steps hence, it is easier for the programmer to convert it into an actual program.
- **Disadvantages**:
  - Writing an algorithm takes a long time so it is time-consuming.
  - Understanding complex logic through algorithms can be very difficult.
  - Branching and Looping statements are difficult to show in algorithms.

Space and time complexities

- **Space complexity**:
  - It means **how much space of memory** is consumed during the **run-time**.
  - Space complexity is **increased** as long as you define **more variables** that allocates more memory.
- **Time complexity**:
  - It means **how much time** does it take **to finish** the needed operation.
  - Time complexity is **increased** as long as you make **more iterations/steps** to reach your goal.

Space and time complexities

- Complexity is measured in Big-O notation.
- Time complexity is CPU operations Vs. data size.
  - **O(1):**
    - This means there is **no dependency** on the input data size (**The best**).
  - **O(log n):**
    - This means there a **logarithmic increase** in operations.
  - **O(n):**
    - This means that number of operations is the **same** as the data size.
  - **O(n log n):**
    - This means that there is **more increase** in operations.
  - **O(n^2):**
    - This means that number of operations is **increasing rapidly** with any small change in data size (**Very bad**).



Space and time complexities

Algorithm	Time Complexity (Worst)	Space complexity (Worst)
Linear search	O(N)	O(1)
Binary search	O(logN)	O(1) or O(logN)
Jump search	O(sqrt(n))	O(1)
Bubble sort	$O(n^2)$	O(1)
Insertion sort	$O(n^2)$	O(1)

Some algorithms applications

- The Internet.
- In a transportation firm such as a trucking or railroad company, may have financial interest in finding shortest path through a road or rail network.
- A routing node on the Internet may need to find the shortest path through the network in order to route a message quickly.
- Searching and sorting products in eCommerce applications.
- Banking systems and payment gateways.

Algorithm lifecycle

- **Preparing the prerequisites**:
  - The **problem** that is to be solved by this algorithm i.e. clear problem definition.
  - The **constraints** of the problem that must be considered while solving the problem.
  - The **input** to be taken to solve the problem.
  - The **output** to be expected when the problem is solved.
  - The **solution** to this problem, within the given constraints.
- **Implement** the algorithm, using any language.
- **Test** the algorithm.

Summary

- Now you have good understanding about algorithms.
- It’s clear that algorithms are involved in many applications.
- Choosing suitable algorithm results in fast applications.
- Time and space complexities are very important for algorithms.
- Algorithms are well defined finite instructions to solve a problem.