# Design and Analysis of Algorithm

Date
Page

## History of Algorithm :-

A parsian Author Name "Khowarizmi" wrote a book about Mathematics in which he predicted a step by step approach to perform basic mathematical operations like addition, substraction, Multiplication, division.

His name when written in latin becomes "Algorismium" from which the today known word Algorithm is taken.

⇒ A digital world like today faces digital problem these problems have complexities like time and space.

example of an Algorithm cause - Euclid's Algorithm to find greatest common devisor which was first ever non-trivial algorithm device.

simple Algorithm.

- Take a pan and turn on the flame.
- take some water.
- then tealeaves.
- then sugar.
- then ginger.
- then Milk.
- leave it for sometime and boil it.

Date
Page

Analysis of Algorithm: Any Algorithm can be analyze two types of approaches, Namely, Priori, Postier:

Postier:-

Postier analysis is perform after the execution or coding for a particular Algorithm this type of analysis basically compares the performance based on Hardware.

Prior:- It is the process of finding out the computation of complexity of any algorithm this can be measure through calculating the time it takes and the space it occupy. (Means any Algorithm).

Aim of any Analysis is to compare various algorithms used to solve the same problem.

⇒ There are three of Analysis:-

1) Best-Case Analysis- This type of Analysis gives the user the lower bound on the running time of a particular algorithm for any given of inputs. its takes that how much the time an algorithm takes atleast. it is also donated by big omega or big (Ω).

Date
Page

2) Worst-case analysis - The worst case analysis provides the upper bounds of the execution time for any algorithm (Legitimate inputs.) it takes the maximum time taken by an algorithm to solve a particular problem.

☆ Frequency of worst case usage is more than best case usage.

3) Average-case Analysis: This Analysis takes the sum of the running-time in all the Legitimate cases. and then calculates the average. also known as theta Notation.

Example:- Linear Search:- searching an element x in a given Array.

```
int search (int arr[], int l, int target)
{
  for (i=0; i<l; i++)
    if (i== target)
    {
      print arr[i]
    }
  else
    print -1.
}
```

Best case - O(1)
Worst case - O(n)
Avg case - O(n/2).

Date
Page

# Asymptotic Notations:

## (1) Big oh (O):-

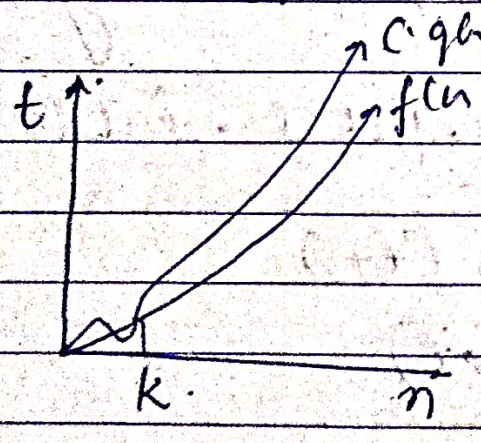
$f(n) \leq c \cdot g(n)$ | $\therefore$ C is A.C
$f(n) = O(n)$ | $\therefore c \geq k$
Let $f(n) = 2n^2 + n$ | $\therefore n \geq 0$
$2n^2 + n \leq c \cdot g(n)$
$2n^2 + n \leq c \cdot n^2$
$2n^2 + n \leq 3n^2$

* $n^2$ is Quadratic variable whereas $n$ is polynomial ; so the dominating element is $n^2$.

$\therefore 2n^2 + n \leq c \cdot n^2$

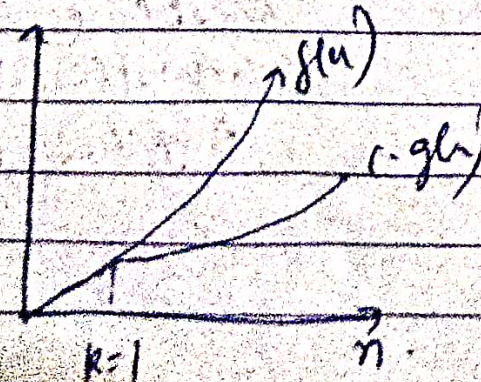
* C is an Arbitrary Constant which should satisfy the above condition.

$2n^2 + n \leq 3n^2$




## (2) Big-Omega ($\Omega$):

$f(n) \geq c \cdot g(n)$ | $\therefore$ C is A.C
$2n^2 + n \geq c \cdot g(n)$ | $\therefore c \geq k$
$2n^2 + n \geq c \cdot n^2$ | $\therefore n \geq 0$
$2n^2 + n \geq 2n^2$
$n \geq 0$



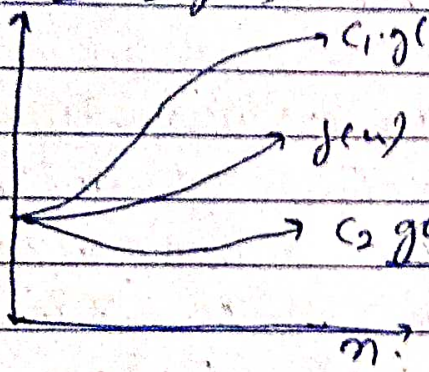
Date
Page

3) Theta ($\Theta$)

$c_1 B \geq \Theta \geq c_2 B$

$3n^2 \geq \Theta \geq 2n^2$

$c_1 \cdot g(n) \leq \Theta \leq c_2 \cdot g(n)$




(4) Small oh ($o$) -

$f(n) < c \cdot g(n)$

(5) small Omega ($\omega$) -

$f(n) > c \cdot g(n)$

Properties of Asymptotic Notations :-

| | | Reflexive | Transitive | Symmetric |
|---|---|---|---|---|
| $f(n) \leq c \cdot g(n)$ | $O$ | ✓ | ✓ | ✗ |
| $f(n) \geq c \cdot g(n)$ | $\Omega$ | ✓ | ✓ | ✗ |
| $c_1 g(n) \leq \Theta \leq c_2 g(n)$ | $\Theta$ | ✓ | ✓ | ✓ |
| Small oh ($o$) | | ✗ | ✓ | ✗ |
| Small ($\omega$) | | ✗ | ✓ | ✗ |

$\omega$

Date
Page

## Comparison of complexities:-

$$O(c) < O(\log(\log n)) < O(\log(n)) < O(n^{1/2}) < O(n)$$
$$< O(n \log n) < O(n^2) < O(2^n) < O(n^n) < O(2^{2^n})$$

Date 5/02/24
Page

## Measurments for the performance of an Algorithm

Performance Measurment of any algorithm is very useful as it helps to optimise the precision of Algorithm.

1) Time Complexity :- One of the most common ways to measure the performance of an algorithm is time complexity, which is the amount of time it takes for an algorithm to complete its tasks as a function of its input size.

Time complexity is usually express as a function or notation known as asymptotic notation (Big O, Big Ω tweta Θ).
Time complexity helps to estimate how an Algorithm scales with larger inputs compared to other algorithms.

* The comparison is on the same input size with respect to various Algorithms.

2) Space complexity :- The another way to measure performance of an Algorithm is to compare the memory it uses for any given input size. space complexities can also be expressed using Asymptotic Notation.

Date
Page

for example:- The space complexity with the O(1) uses less storage space than. $O(n^2)$. space complexity helps to evaluate how an Algorithm manages the resources. by which it impacts the overall performa

(3) Runtime analysis:- A more practical way to measure algorithm performance is. runtime analysis. it is an actual measurem of how long an algorithm takes to run on a specific machine or environment. measurment tools:- Timers, Profilers, Benchmarks,

(4) Asymptotic Notation.

(5) Euperical (form) Analysis:- It is more of an experimental way to measure the performance of an algorithm. in this type of analysis the observation and the evaluation of how an algorithm works is put into various scenerios. It is also known as case-base performance measure.

(6) Algorithm Design technique:- One of the best ways to improve the performance of an algorithm is to understand different strategies.

Date
Page

and principle have been choosen to create an efficient Algorithm. These designing technique can be divided into two categories-

① Problem Reduction.

② Solution Construction.