# Lecture Notes

on

# Analysis of an Algorithm


July 2020

**(Be safe and stay at home)**

# Analysis of an Algorithm

- To decide which one is best among diffent algorithm for the same problem.
- We have two algorithm (Algorithm A & Algorithm B) for the same problem.
- The anlysis of an algorithm is done based on :
  i) Time Complexity (CPU time)
  ii) Space Complexity (Main Memory)
- In front of CPU time, memory becomes cheaper day by day, so CPU time is taken first.
- When time is same then go for memory(space).

# Example

Problem:(any sorting)

| Algo A | Algo B | |
|---|---|---|
| $n^2$ | $n^3$ | A better, Diff. in time |
| $n^3$ | $n^2$ | B better, Diff. in time |
| $n^3$ | $n^3$ | Same time, diff. space, (B better) |
| $n^2$ | $n^2$ | Same time same space (equal performance) |

- $A = \theta(B)$ Same
- $A = O(B)$, B is worst
- $A = \omega(B)$ , B is best

# Time Complexity

- Time complexity is the sum of compiler time and running time.
- $T(A) = C(A) + R(A)$
  we can discard the compiler time as its negligable (each algorithm compile only one time in it's lifetime.)
  $T(A) = R(A)$
- Type of language (fast/slow) is used for compiler, decides **compiler time.**
- If we compare c & Java compiler, C compiler is fast.
  Java compiler is slow as it has lots of packages, linking and all takes more time.

# Types of Analysis

Two types of analysis:

| Apostiory Analysis | Apriori Analysis |
|---|---|
| Its plateform and hardware dependent | Its plateform and hardware independent |
| It gives exact answer | Approximate answer |
| Answer is changing from one system to another | remains same for all |

| | | | | |
|---|---|---|---|---|
| A | pascal compiler, pentium | more time | $100.n^2$ | $n^2$ |
| | Java compiler, pentium-2 | less than above | $10.n^2$ | |
| | C compiler, pentium-2 | less time | $n^2$ | |
| | C compiler, Super fast computer | very less time | $\frac{n^2}{100}$ | |

# Apriori Analysis

- It's a determination of **order of magnitude** of a statement.
- Number of times a statement is executed while running is called **order of magnitude.**

**Example 1:**

```
main()
{
x=y+z; —> one time —>O(1)
}
```

# Apriori Analysis Contd...

**Example 2:**

main()
{
x=y+z; —> one time —>O(1)
for(i=0;i<=n;i++)
{
x=y+z; —>O(n) }
}
$= O(1+n)(n)$

# Q & A?

Queries are welcome on slack channel for discussion