Algorithm Analysis framework

→ Random access machine (RAM) is a model of Computation where instructions are executed one after another with no Concurrent operations.

- RAM model Contain instructions Commonly found in real Computer : arithmetic (add, Subtract, multiply, divide, remainder, floor, ceiling), data movement (load, Store, Copy) and Control (Conditional, un Conditional branch, Subroutine Call and return)
- Each instruction takes a Constant amount of time.

(Last Page) Heading

→ There are two kind of efficiency (Complexity)

① Space Complexity

- It is the amount of memory it needs to run to Completion ie from start of execution to its termination.

- Space needed by any algorithm is the Sum of following Components

1) Fixed Component

this is independent of the Characteristic of the inputs and outputs. It includes Instruction Space, Space of Simple variables, fixed Size Component variables and Constant variables.

2) Variable Component

- This Consist of the Space needed by Component variables whose Size is dependent on the particular problem instances (I/P) being Solved, the Space needed by reference variables and the recursion Stack Space.
- Data Structure Components like linked List, heap, trees graphs.

Space (A) = Fixed Components (A) + Variable Components (A)

(AJAY RAWAT)

(2) Time Complexity

- It is the amount of Computer Time it needs to run to Completion.
- Time Taken by a program is the Sum of the Compile time and the run/execution time.
- the Compile time does not depend on the instances Characteristics

→ **Input Size**

- One of the instance characteristics for run time Complexity of an algorithm is input size.
- Longer input size make algorithm to run longer time.
- the input size for the problem of summing an array with 'n' elements is n+1 (n for listing the 'n' elements and 1 for 'n' value).

→ **Unit of Measuring Running time**

- The drawback of using time (second, millisecond) as a standard unit.
  - Speed of Computer
  - Quality of program implementing the algorithm
  - Compiler used in generating machine code.
  - Difficult in Clocking the actual running time of an algo.
- So have to look for metric that does not depend on these extraneous factor.
- Identify the most important operation (Basic operation) of algorithm.

- Basic operation is nothing but core operation, generally resides in inner loop.
  eg - in Sorting algo.
  - Comparing the elements.
  - Placing them in appropriate position.

→ Worst-Case, Best Case, and Average Case Efficiencies

→ Worst Case efficiency of an algorithm is its efficiency for worst case input of size n
- Algo runs the longest among all possible input of that size.
- It provides very important information about an algorithm's efficiency by bounding its running time from above.

→ Best Case efficiency of an algorithm is its efficiency for Best Case input of size n.

- Algorithm runs the fastest among all possible inputs of that size.

- Best Case does not mean the smallest input, it means the input of size n for which the algorithm runs the fastest.

- eg for Sorting algo (already sorted array) (Bounding from below mean Lower Bound)

→ Average Case
- Take all possible inputs and Calculate computing time for all of the inputs

- Sum all the Calculated values and divide the sum by

total no of inputs.

- eg linear Search.

- Must Know (predict) the mathematical distribution of all possible inputs.

Graphical Representation

Analysis of Algorithms

1 - Measuring time Complexity
2 - Measuring Space Complexity
3 - Measuring input Size
4 - Measuring running time.
5 - Computing best Case, worst Case, average Case efficiency.
6 - Computing order of growth of algorithms.