Need for Analysing Algorithm

Q) Why we need to Analyse Algorithm?

Ans: It is reasonable to measure an algorithm's efficiency as a function of a parameter indicating the size of the algorithm's input. But there are many algorithms for which running time depends not only on an input size but also on the specifics of a particular input.

Such for Sequential Search:

ALGORITHM: SEQUENTIAL SEARCH ($A[0..n-1], K$)

//Searches for a given value in each array by sequential Search

//Input: An array A[0..n-1] and a search key K

//Output: The index of the first element in A that matches K

// or -1 if there are no matching elements.

$1 \leftarrow 0$

while $i < n$ *and A[i]* $\neq K$ *do*

$i \leftarrow i + 1$

if $i < n$ *return* i

else return -1

Consider as an example, sequential search. This is a straight forward algorithm that searches for a given item (some search key K) in a list of n elements by checking successive elements of the list until either a match with the search key is found or the list is exhausted. In the above pseudocode, in which for simplicity, a list is implemented as an array. It also assumes that the second condition $A[i] \neq K$ *will not be checked if the first one , which checks that the array's index does not exceed its upper bound , fails.*

Clearly, the running time of this algorithm can be quite different for the same list size n.

<u>In conclusion:</u>

- *It is logical to assume that the efficiency of algorithms depends solely on the input size n.*

- *Sometimes, efficiency of an algorithm depends upon the distribution of input data as well.*

Therefore, to analyse the given algorithm , we need to know with which inputs the algorithm takes less time(performing well) and with which inputs the algorithm takes a long time .

That means we represents the algorithm with multiple expressions:

- *One for the case where it takes less time.*
- *Another for the case where it takes more time.*

In general, the first case is called ,

- *Best Case*

The second case is called ,

- *Worst Case*

And suppose the input is random and number of trials taken gives an average runtime , at that time we got a third case called,

- *Average Case*

To analyse an algorithm, we need some kind of syntax and that forms the base for Asymptotic Analysis/Notation.