SNS COLLEGE OF TECHNOLOGY

Coimbatore-35
An Autonomous Institution

Accredited by NBA – AICTE and Accredited by NAAC – UGC with 'A+' Grade
Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai

DEPARTMENT OF INFORMATION TECHNOLOGY

19ITB201 – DESIGN AND ANALYSIS OF ALGORITHMS

II YEAR IV SEM

UNIT-I-Introduction

TOPIC: Fundamentals of the Analysis of Algorithm Efficiency – Analysis Framework

Prepared by
C.PARKAVI,AP/AIML

FUNDAMENTALS OF THE ANALYSIS OF ALGORITHM EFFICIENCY – ANALYSIS FRAMEWORK

Subject :Design and Analysis of Algorithm
Unit :I

Connection-to identify the topic from given image

Answer: Analysis Framework

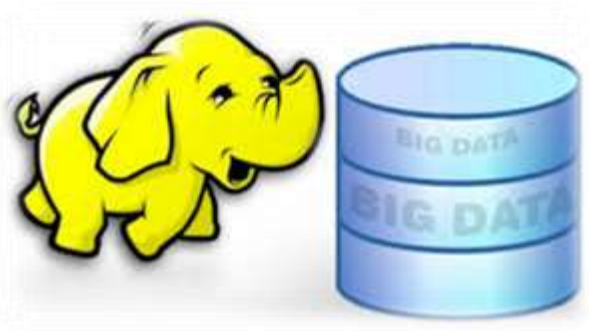
Fundamentals of the Analysis of Algorithm Efficiency

- Analysis Framework
- Asymptotic Notations and its properties
- Mathematical analysis for Recursive algorithms.
- Mathematical analysis for Non recursive algorithms.

Analysis Framework

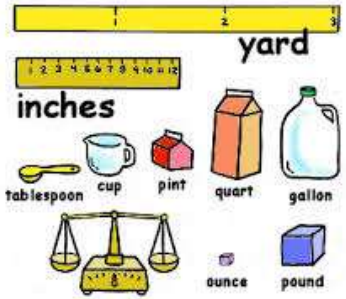
There are two kinds of efficiencies to analyze the efficiency of any algorithm. They are:

- ***Time efficiency***, indicating how fast the algorithm runs, and
- ***Space efficiency***, indicating how much extra memory it uses



The algorithm analysis framework consists of the following:

- Measuring an Input's Size
- Units for Measuring Running Time
- Orders of Growth
- Worst-Case, Best-Case, and Average-Case Efficiencies



Orders of Growth

- A **difference in running times on small inputs** is not what really distinguishes efficient algorithms from inefficient ones.
- For example, the greatest common divisor of two small numbers, it is not immediately clear how much more efficient Euclid's algorithm is compared to the other algorithms, the difference in **algorithm efficiencies becomes clear** for larger numbers only.

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

Consider Sequential Search algorithm some search key *K* **ALGORITHM** *SequentialSearch(A*[0..*n* - 1], *K)*

//Searches for a given value in a given 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

$i \leftarrow 0$

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

$i \leftarrow i + 1$

if $i < n$ **return** i

else return -1

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

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

Worst-case efficiency:-

•The ***worst-case efficiency*** of an algorithm is its efficiency for the worst case input of size *n*.

•The algorithm runs the longest among all possible inputs of that size.

•For the input of size *n*, the running time is ***Cworst(n) = n.***

Best case efficiency

•The ***best-case efficiency*** of an algorithm is its efficiency for the best case inp ut of size *n*.

•The algorithm runs the fastest among all possible inputs of that size n.

•In sequential search, If we search a first element in list of size *n. (i.e.* first ele ment equal to a search key), then the running time is

***Cbest(n)* = 1.**

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

Average case efficiency

- The Average case efficiency lies **between best case and worst case**.
- To analyze the algorithm's average case efficiency, we must make some assumptions about possible inputs of size *n*.

The standard assumptions are that

- The probability of a successful search is equal to p $(0 \leq p \leq 1)$ and
- The probability of the first match occurring in the *i*th position of the list is the same for every *i*.

Assessment

1.Two main measures for the efficiency of an algorithm are

A. Processor and memory

B. Complexity and capacity

C. Time and space

D. Data and space

2. Which of the following case does not exist in complexity theory?

A. Best case

B. Worst case

C. Average case

D. Null case

Thank you!