"बेटी बचाओ, बेटी पढ़ाओ"

JAYOTI VIDYAPEETH WOMEN'S UNIVERSITY, JAIPUR

Faculty of Education & Methodology

Faculty Name	-	JV'n Nisha kumari
		(Asst. Prof./ Asso. Prof./ Professor)
Program	-	B-tech/Vth Semester / Year
Course Name	-	Design and Analysis of Algorithms
Session No. & Name	-	1. & Design and Analysis of Algorithms

Academic Day starts with –

- Greeting with saying **'Namaste'** by joining Hands together following by 2-3 Minutes Happy session, Celebrating birthday of any student of respective class and **National Anthem**.

Lecture starts with- quotations' answer writing

Review of previous Session - **Discussion Computer Language**

- Topic to be discussed today- Today We will discuss about **…… Active database………..**
- Lesson deliverance (ICT, Diagrams & Live Example)-

➢ PPT (10 Slides)

➢ Diagrams

Introduction & Brief Discussion about the Topic

What is meant by Algorithm Analysis?

Algorithm analysis is an important part of computational complexity theory, which provides theoretical estimation for the required resources of an algorithm to solve a specific computational problem. Analysis of algorithms is the determination of the amount of time and space resources required to execute it.

Why Analysis of Algorithms is important?

- To predict the behavior of an algorithm without implementing it on a specific computer.
- It is much more convenient to have simple measures for the efficiency of an algorithm than to implement the algorithm and test the efficiency every time a certain parameter in the underlying computer system changes.
- It is impossible to predict the exact behavior of an algorithm. There are too many influencing factors.
- The analysis is thus only an approximation; it is not perfect.
- More importantly, by analyzing different algorithms, we can compare them to determine the best one for our purpose

Average Case Time = \sum_{i=1}^{n}\frac{\theta (i)}{(n+1)} = \frac{\theta (\frac{(n+1)*(n+2)}{2})}{(n+1)} = \theta (n)

History:

- The word algorithm comes from the name of a Persian author, Abu Jaffar Mohammed ibn Musa al Khwarizmi (c. 825 A.D.), who wrote a textbook on mathematics.
- He is credited with providing the step-by-step rules for adding, subtracting, multiplying, and dividing ordinary decimal numbers.

- When written in Latin, the name became Algorisms, from which algorithm originated.
- This word has taken on a special significance in computer science, where "algorithm" has come to refer to a method that can be used by a computer for the solution of a problem.
- Between 400 and 300 B.C., the great Greek mathematician Euclid invented an algorithm.
- Finding the greatest common divisor (GCD) of two positive integers.
- The GCD of X and Y is the largest integer that exactly divides both X and Y.
- For example, the GCD of 80 and 32 is 16.
- The Euclidian algorithm, as it is called, is the first non-trivial algorithm ever devised.

The history of algorithm analysis can be traced again to the early days of computing when the first digital computer systems were developed. In the 1940s and 1950s, computer scientists commenced to develop algorithms for solving mathematical issues, including calculating the value of pi or solving linear equations. These early algorithms had been frequently simple and easier, and their performance was not a major challenge. As computers have become extra powerful and have been used to resolve increasingly more complicated problems, the need for efficient algorithms has become more critical. In the 1960s and 1970s, computer scientists began to increase techniques for reading the time and area complexity of algorithms, such as the use of big O notation to explicit the growth price of an algorithm's time or space necessities.

During the 1980s and 1990s, algorithm analysis became a crucial mode of research in computer technology, with many researchers working on developing new algorithms and reading their efficiency. This period saw the development of several critical algorithmic techniques, including divide and conquer algorithms, dynamic programming, and greedy algorithms.

Today, algorithm analysis has a crucial place of studies in computer science, with researchers operating on developing new algorithms and optimizing existing ones. Advances in algorithmic evaluation have played a key function in enabling many current technologies, inclusive of machine learning, information analytics, and high-performance computing.

Types of Algorithm Analysis:

There are numerous types of algorithm analysis which can be generally used to measure the performance and efficiency of algorithms:

1. **Time complexity evaluation:** This kind of analysis measures the running time of an algorithm as a characteristic of the input length. It typically entails counting the quantity of primary operations completed by way of the algorithm, such as comparisons, mathematics operations, and reminiscence accesses.
2. **Space complexity evaluation:** This form of evaluation measures the amount of memory required via an algorithm as a characteristic of the enter size. It typically includes counting the variety of variables and information systems utilized by the algorithm, as well as the size of each of these records structures.
3. **Worst-case evaluation:** This type of analysis measures the worst-case running time or space utilization of an algorithm, assuming the enter is the maximum toughest viable for the algorithm to deal with.
4. **Average-case analysis:** This kind of evaluation measures the predicted walking time or area usage of an algorithm, assuming a probabilistic distribution of inputs.

5. **Best-case evaluation:** This form of analysis measures the nice case running time or area utilization of an algorithm, assuming the input is the easiest possible for the algorithm to address.
6. **Asymptotic analysis:** This sort of analysis measures the overall performance of an algorithm as the enter size methods infinity. It normally includes the usage of mathematical notation to describe the boom fee of the algorithm's strolling time or area usage, including O(n), Ω(n), or Θ(n).

These sets of algorithm analysis are all useful for information and evaluating the overall performance of various algorithms, and for predicting how properly an algorithm will scale to large problem sizes.

Advantages of design and analysis of algorithm:

There are numerous blessings of designing and studying algorithms:

1. **Improved efficiency:** A properly designed algorithm can notably improve the performance of a program, leading to quicker execution instances and reduced resource utilization. By studying algorithms and identifying regions of inefficiency, developers can optimize the algorithm to lessen its time and space complexity.
2. **Better scalability:** As the size of the input information will increase, poorly designed algorithms can quickly turn out to be unmanageable, leading to slow execution times and crashes. By designing algorithms that scale well with increasing input sizes, developers can make certain that their packages stay usable while the facts they take care of grows.
3. **Improved code exceptional:** A nicely designed algorithm can result in better code first-rate standard, because it encourages developers to think seriously about their application's shape and organization. By breaking

down complicated issues into smaller, extra manageable sub problems, builders can create code that is simpler to recognize and maintain.

4. **Increased innovation:** By knowing how algorithms work and how they can be optimized, developers can create new and progressive solutions to complex problems. This can lead to new merchandise, services, and technologies which can have a considerable impact on the arena.
5. **Competitive benefit:** In industries where pace and performances are vital, having properly designed algorithms can provide an extensive competitive advantage. By optimizing algorithms to lessen expenses and enhance performance, groups can gain a facet over their competitors.

Overall, designing and analyzing algorithms is a vital part of software program improvement, and can have huge advantages for developers, businesses, and quit customers alike.