Aayush Shah
19BCE245
28 July 2021

Design and Analysis of Algorithms

Tutorial 1

1. **What do you mean by an algorithm? Explain the importance of an algorithm by suitable example.**

A set of well-defined instructions to solve a particular problem is an Algorithm in computer programming terms. It takes a set of input and produces a desired output.

An algorithm is important in many ways so that we can Optimise a computer program according to the available resources.

- Example :

For an E-Commerce website like amazon, flipkart or Ebay, They use algorithms in their sites and backends for their requirements. For an example, They have to make a efficient search engine for all the available products, recommended products section should be relevant to the user, Products shown as per user's nearby location, and in many things.

2. **What is the smallest value of n such that an algorithm whose running time is $100n^2$ runs faster than an algorithm whose running time is 2^n on the same machine?**

Let us assume that there is an Algorithm 1 which has running time of $100n^2$, and another Algorithm 2 which has running time of 2^n.

If Algorithm 1 wants to be faster than Algorithm 2 than Algorithm 1's running time must be smaller than Algorithm 2.

So that,

Running time of Algorithm 1 < Running time of Algorithm 2

$100n^2 < 2^n$

By trial and error (putting values from n=1, 2, 3…),

We came to know that at n=15,

$100n^2$ becomes 22500 and 2^n becomes 32,768 and after increasing the value of n, we will always satisfy the condition.

∴ So, the smallest value of n such that an algorithm whose running time is $100n^2$ runs faster than an algorithm whose running time is 2^n.

3. Describe the method for analysing an algorithm. What do you mean by best case, average case and worst case time complexity of an algorithm?

There are four methods available for analysing an algorithm. Time complexity is one of them, which is defined by amount of time any instruction runs in specific amount of time.

- **Best case time complexity** : If an algorithm requires minimum number of operations or steps or tasks to deliver the correct output, then that algorithm's time complexity is known as best case time complexity of an algorithm. Here, the algorithm takes fastest time to complete with optimal input chosen.

  For example, the best case for a sorting algorithm would be data that's already sorted.

- **Average case time complexity** : If an algorithm requires arbitrary number of operations to deliver the correct output, then that algorithm's time complexity is know as average case time complexity of an algorithm.

- **Worst case time complexity** : If an algorithm requires maximum number of operations to deliver the correct output, then that algorithm's time complexity is known as worst case time complexity of an algorithm.

4. What is the difference between Apriori approach and Posteriori approach for solving any problem? Discuss with proper example.

The analysis of algorithms are done one the basis of two factors : Time Complexity and Space Complexity. Time Complexity of an algorithm can be calculated by using two methods : Posteriori Analysis and Priori Analysis.

Here is the difference between Apriori analysis and Posteriori analysis :

A Posteriori analysis	A priori analysis
Posteriori analysis is a relative analysis.	Piori analysis is an absolute analysis.

It is dependent on language of compiler and type of hardware.	It is independent of language of compiler and types of hardware.
It will give exact answer.	It will give approximate answer.
It doesn't use asymptotic notations to represent the time complexity of an algorithm.	It uses the asymptotic notations to represent how much time the algorithm will take in order to complete its execution.
The time complexity of an algorithm using a posteriori analysis differ from system to system.	The time complexity of an algorithm using a priori analysis is same for every system.
If the time taken by the algorithm is less, then the credit will go to compiler and hardware.	If the program running faster, credit goes to the programmer.

5. **Among Merge-Sort, Insertion-Sort and Bubble-Sort which sorting technique is the best in the worst case. Support your argument with an example and analysis.**

Here are time complexities of Merge Sort, Insertion Sort and bubble sort in the worst case :

Merge Sort : $O(n \log(n))$

Insertion Sort : $O(n^2)$

Bubble Sort : $O(n^2)$

Here the time complexity of Merge sort is the best among other sorting algorithms' time complexity.

For Example, to sort a 10^5 sized array, Merge Sort took 19 ms. Whereas Bubble Sort took 14392 ms and Insertion Sort took 1318 ms.