AA Tutorial 7

Krishnaraj P. Thadesar
PA20 A1

Q.1. Define the class of P, NP, NP complete & NP Hard problems with suitable examples

→ (1) P :→ P stands for polynomial time. It refers to a class of decision problems that can be solved by a deterministic turing machine in polynomial time. In other words, these problems can be solved efficiently.

Examples: Sorting an array
Searching an array
Matrix multiplication.

(2) NP : Stands for Non-deterministic polynomial time. Here, a solution can be verified in polynomial time by a deterministic turing machine; However, finding the solution may not be computationally efficient.

Ex. Travelling Salesman Problem
Boolean satisfyability and graph coloring.

(3) NP-Complete : It refers to the class of decision problems that are both in NP, and are as hard as the hardest problems in NP.

A problem is considered NP complete if any problem is any problem is NP can be converted to it, is polynomial time.

eg. Satisfyability problem

(4.) NP. Hard : stands for Non-deterministic polynomial time hard. It refers to the class of problems that are atleast as hard ~~as~~ as the hardest problem is NP but may not necessarily be is NP themselves. They are computationally challenging, and no efficient solution has been found for them.

eg. Hamiltonian cycle problem.

Q.2. Explain Algorithm for non deteerministic searching.

→ Non-deterministic searching is a concept used in theoritical computer science to analyse problems in the NP class. In non deterministic searching, the algorithm explores different possible paths or solutions simultaincously without knowing which one will lead to the correct solution.

→ This algorithm assumes the existance of a magical oracle that can provide the correct

solution at each step

Algorithm:

1. Enumerate all possible solutions that can be explored

2. For each possible solution or path check if it satisfies the given constraint or requirement.

3. If any possible solutions satisfy the constraints, accept the solution.

4. If none of the ~~problems~~ possible solutions are satisfiable, reject the problem instance

This only allows for a theoritical analysis of problems and not to find the actual solution.

Q.3. Define randomized algorithms. Explain Randomized Quick sort.

→ Randomized algorithms : They are algorithms that use a random component during their execution. They make use of random choices to improve efficiency or ensure the algorithm's correctness probabilistically. Randomness is introduced either in input or during the algorithm's intend operations.

Randomized quicksort can be described as follows

1. Choose a random element from the array as a pivot
2. Partition the array into 2 subarrays, one with elements smaller than the pivot and one with elements greater that the pivot.
3. Recursively apply the randomized quicksort algorithm to the subarrays generated in the previous step.
4. ~~Combine~~ Combine the sorted subarrays to obtain the final sorted array.

By introducing randomness in pivot selection, randomized quicksort achieves an average-case time complexity of $O(n \log n)$ with a high probability, even for inputs that can cause worst-case time complexity.