Article:-P, NP, NP-Hard & NP-complete problems.

* Basic Intoduction
* Type of Problems
* Introduction of P, NP, NP-Hard & NP-complete problems

Intoduction:-

Algorithm:-

Algorithm is a set of rules that must be followed when solving a particular problem.

Algorithm mean a finite number of roles which is used for solving a given problem.

Every problems are divided into classes known as **Complexity Classes.**

The common resources are time and space, meaning how much time the algorithm takes to solve a problem and the corresponding memory usage.  
The time complexity of an algorithm is used to describe the number of steps required to solve a problem, but it can also be used to describe how long it takes to verify the answer.

Types of Complexity Classes

This article discusses the following complexity classes:

P Class

NP Class

NP hard

NP complete

Problem:-

Types of Problems

• Trackable

• Intrackable

• Decision

• Optimization

Trackable : Problems that can be solvable in a reasonable(polynomial) time.

Intrackable : Some problems are *intractable,* as they grow large, we are unable to solve them in reasonable time.

Tractability

• What constitutes reasonable time?

– Standard working definition: *polynomial time*

– On an input of size *n* the worst-case running time is O(*nk*) for some

constant *k*

– O(n2), O(n3), O(1), O(n lg n), O(2*n*), O(*n*n), O(*n*!)

– Polynomial time: O(n2), O(n3), O(1), O(n lg n)

– Non polynomial time: O(2*n*), O(*n*n), O(*n*!)

Optimization/Decision Problems:-

• Optimization Problems lem is one which asks,“What is the optimal solution to problem X?”

– Examples:

• 0-1 Knapsack

• Fractional Knapsack

• Minimum Spanning Tree

• Decision Problems

Optimization/Decision Problems

• An optimization problem tries to find an optimal solution

• A decision problem tries to answer a yes/no question

• Many problems will have decision and optimization versions

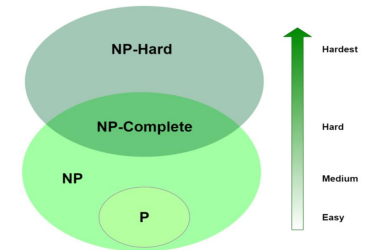
– Eg: Traveling salesman problem

• optimization: find hamiltonian cycle of minimum weight

• decision: is there a hamiltonian cycle of weight

Introduction of P, NP, NP-Hard & NP-complete problems:-

There are many type of problem like maintion in problem statement.



The Class P

P: the class of problems which is solve in polynomial-time with deterministic algorithms.

– That is, they are solvable in O(p(n)), where p(n) is a polynomial on n

– A deterministic algorithm is (essentially) one that always computes the correct answer

Sample Problems in P

• Sorting

• Searching

The class NP

NP: the class of decision problems that are not solvable in polynomial time but we can verify the answer in polynomial time.

– A nondeterministic computer is one that can “guess” the right answer or solution

• Think of a nondeterministic computer as a parallel machine that can freely spawn an infinite number of processes

• Thus NP can also be thought of as the class of problems “whose solutions can be verified in polynomial time”

• Note that NP stands for “Nondeterministic Polynomial-time”

Sample Problems in NP

• Fractional Knapsack

• MST

• Others?

– Traveling Salesman

– Graph Coloring

– Satisfiability (SAT)

• the problem of deciding whether a given

Boolean formula is satisfiable

Reduction

• A problem R can be reduced to another problem Q if any instance of R can be rephrased to an instance of Q, the solution to which provides a solution to the instance of R

– This rephrasing is called a transformation

• Intuitively: If R reduces in polynomial time to Q, R is “no harder to solve” than Q

In other word :-

Polynomial time

Reduction

Convert the exponential time problem to polynomial time problem are called reduction.

NP-hard class

• What does NP-hard mean?

– A lot of times you can solve a problem by reducing it to a different problem. I can reduce Problem B to Problem A if, given a solution to Problem A, I can easily construct a solution to Problem B. (In this case,"easily" means "in polynomial time.“).

• A problem is NP-hard if all problems in NP are polynomial time reducible to it, ...

Every problem in NP is reducible to HC in polynomial time. Ex:- TSP is reducible toHC.

Example: lcm(m, n) = m \* n / gcd(m, n), B A Ex:- Hamiltonian Cycle

Ex :- if we have a two problems problem A and problem B

If problem A solve with the help of problem B .

A solve/belongs to

**A** is a exponential problem and solve with non determination algorithm it become solve.

Solve by non determination algo.

Np problem after solve it become Np hard

NP-complete problems

• A problem is NP-complete if the problem is both

– NP-hard, and

– NP. Np complete

p

Np Np hard