**NATIONAL INSTITUTE OF TECHNOLOGY**

**DELHI**

**ASSIGNMENT – 5**

**DESIGN AND ANALYSIS OF ALGORITHMS**

P and NP problems

Submitted by : - ANKIT GHOSH

Roll no. : - 181210065

Intractable problem

Decidable problem

terminology

Tractable problem

problem

Undecidable problem

Decidable problems :-

Problems for which algorithms exist to solve them.

Undecidable problems :-

Problems for which algorithms do not exist to solve them.

Tractable problem :-

Problems for which polynomial time algorithms exist to solve them.

Intractable problem :-

Problems for which algorithms exist but polynomial time algorithms do not exist.

**P and NP introduction :-**

P (Deterministic Polynomial Time):-

set of all decision problems for which there exist a polynomial time algorithm to solve them.

**Examples :-**

1. Minimum spanning tree problems using prim’s and Kruskal algorithms.
2. Fractional knapsack problem.

NP(Non-Deterministic Polynomial Time) :-

set of all decision problems for which there exist a polynomial time verification algorithm.

**Examples :-**

1. If we are able to solve a problem in polynomial time, we will surely be able to verify in polynomial time, so every P problem will also be a NP problem.
2. Travelling salesman problem :- we are not able to find polynomial time solution for this problem but we can verify this in polynomial time so this a NP problem but not a P problem.

**DIAGRAMATIC REPRESENTATION:-**

**DIFFERENCE BETWEEN P & NP :-**

|  |  |  |
| --- | --- | --- |
| S NO. | P PROBLEMS | NP PROBLEMS |
| 1. | These can be solved in polynomial time by deterministic algorithms. | These can be solved in polynomial time by non-deterministic algorithms. |
| 2. | Such problems can be solved and verified in polynomial time. | NP problems solution cannot be obtained in polynomial time but if solution is given it can be verified in polynomial time. |
| 3. | Till now we believe that P problems are subset of NP problems. | Till now we believe that NP problems are a superset of P problems. |
| 4. | e.g. Searching, Sorting, Addition, Multiplication, etc. | e.g. Sudoku Puzzle ,Travelling Salesman Problem, 0/1 knap sack etc. |

**Polynomial time reduction algorithm :-**

To prove P = NP we have to prove that every problem which lies in NP can be solved in polynomial time

There are millions of NP problems we can’t solve each problem to prove this, here comes reduction.

A problem ‘A’ is said to be polynomial time reducible to a problem ‘B’ if :-

1. Every instance ‘a’ of ‘A’ can be transformed to some instance ‘b’ of ‘B’ in polynomial time.
2. Answer of ‘a’ is ‘YES’ if and only if answer of ‘b’ is ‘YES’.

So

If A is reduced to B in polynomial time then :-

* If B is easy then A is also easy.
* If B is in P then A is also in P.
* If this is proven that A can’t be solved in polynomial time then B is also can’t be solved in polynomial time.
* If A is not in P then B is also not in P.

**NP hard and NP complete problems :-**

NP hard :-

If every problem in NP can be polynomial time reducible to a problem ‘A’ then ‘A’ is called a NP hard problem.

* If ‘A’ could be solved in polynomial time then every problem in NP can be solved in polynomial time so

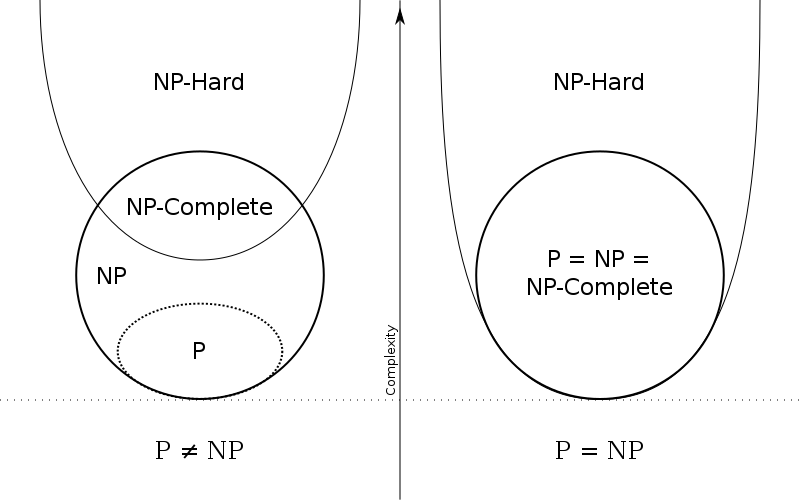
P = NP

NP complete :-

If a problem lies in NP and also a NP hard problem then this problem is called NP complete problem.

**Note :-**

1. If NP hard or NP complete problem is solved in polynomial time then NP = P.
2. If NP or NP complete problem is proven to be not solvable in polynomial time then P NP.
3. If ‘A’ is a NP hard problem and ‘A’ can be reduced in polynomial time in ‘B’ then be is also called a NP hard problem. If this ‘B’ is a NP problem then ‘B’ will be NP complete problem.



**Some examples of NP complete problems :-**

1. Circuit satisfiability problem
2. Satisfiability(SAT) problem
3. 3-CNF satisfiability problem
4. Subset sum problem
5. Clique problem
6. Vertex cover problem
7. Hamiltonian cycle problem
8. Travelling salesman problem