P, NP, NP-Complete and NP-Hard

class P - Problems

- The class P consists of those problems that can be solved and verified in polynomial time.
- More specifically, they are problems that can be solved in time $O(n^k)$ for some constant k,
- For P Class problems, Deterministic algorithms can be written
- P Problems are subset of NP Problems
- If problems belongs to class P, it is easy to find the solution

Example

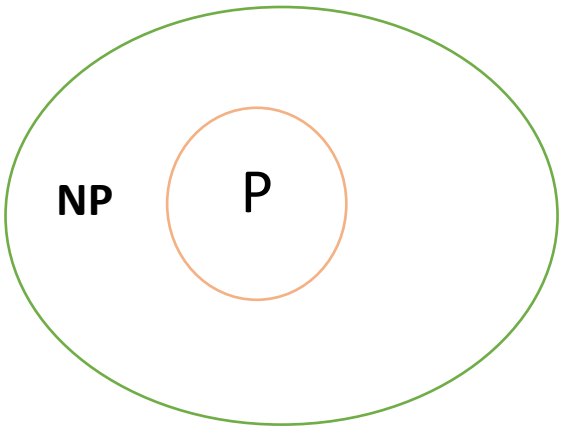
Linear search, Binary Search, matrix multiplication

NP - Non-Deterministic polynomial time

- Solution to NP Problems cannot be obtained in polynomial time, but if the solution is given it can be verified in polynomial time.
- It is the collection of decision problems that can be solved by a non-deterministic algorithm in polynomial time.
- NP problems are super set of P Problems

Example

Travelling Salesperson, knapsack problem



Deterministic Algorithms :

For the Deterministic algorithms, given a particular input it will always produce the same output. In Deterministic algorithm the path of execution for algorithm is same in every execution

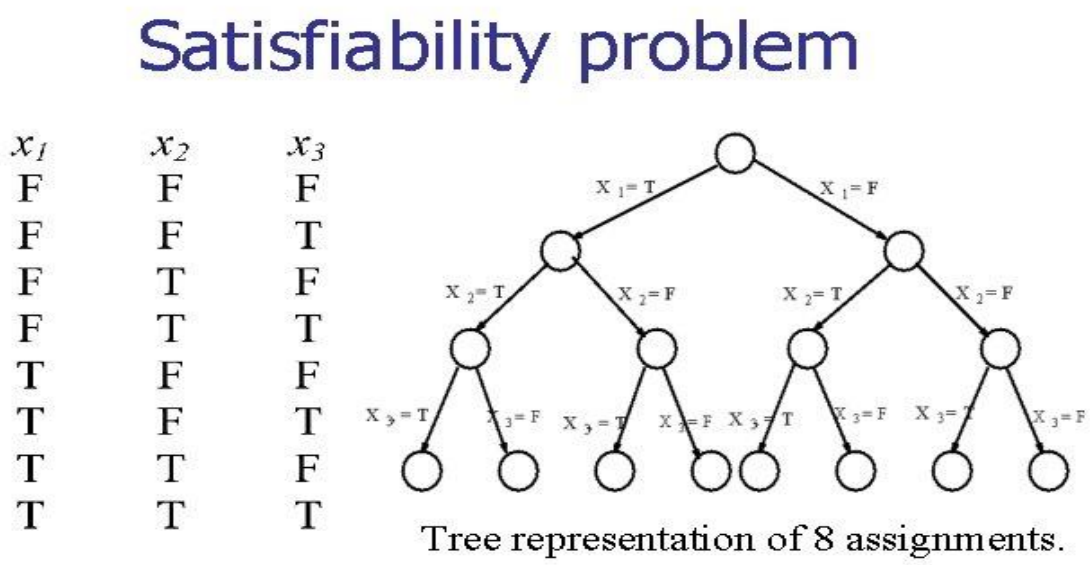
Non-Deterministic Algorithms :

In Non-deterministic algorithms, the path of execution is not same in every execution. The outcomes are in consistent

Reducibility : Let L1 and L2 be two problems. Problem L1 reduces to L2 also written as L1 ∞ L2. It means if we have a polynomial time algorithm for L2, then we can solve L1 in polynomial time. Here ∞ is transitive relation

If L1 ∞ L2 and L2 ∞ L3 then L1 ∞ L3

Satisfiability – Satisfiability is problem where given a boolean expression, determining if there exists a truth assignment to its variables



If there are n variables $x_1, x_2, \ldots, x_n$, then there are 2^n possible assignments.

NP – Hard :

A problem L is Np-hard if and only if satisfiability reduces to L.

To show that a problem L2 is NP-hard, it is adequate to show L1 ∞ L2, where L1 is some problem already known to be NP-hard.

Example

Halting problem, vertex cover problem.

NP – Complete :

A problem is NP-complete if it is both NP and NP-hard.

If one could solve an NP-complete problem in polynomial time, then one could also solve any NP problem in polynomial time. For an NP-Complete problem there should be a non deterministic algorithm.

Example

satisfiability problems, clique problem.