Introduction to NP-completeness

Till Now, all the Problem we studied in this course have efficient solutions.
By efficient, we mean the algorithms that runs in polynomial times with respect to the size of the input.
But there are many classical problems, for which researchers unable to find efficient solutions.

Example. (1) Longest path in graph.
(2) solving a Sudoku
(3) 0-1 knapsack
These Problems are studied for many decades now w the best solution known are equivalent to explore

all possible solutions are eventually find the optimal one. there are thousands of such Problems exists that have many practical applications. The study of these kind "difficult" problems is the theory of NP- completeness.

Some basic Definitions

The class P : The set of Problems for which polynomial time algorithm exists.

Examples of problems in P

(1) Finding Minimum spanning tree
② searching in an array

③ Longest common subsequence

The class NP : The set of Problems for which a solution can be varified in polynomial time.

example : A Sudoku may take large time to solve but if someone fills the blank cell of a Sudoku, it can be easily verified that whether it is filled properly or not

So in P. you need to Produce a solution in polynomial time. In NP, if a solution instance is provided to You, You need to verify it in polynomial time.

Decision problems A Decision Problem is a Problem which requires to output either yes or NO Answer.

Every optimization problem can be simplified as a decision problem.

Example

DMST - Decision version of Minimum spanning tree

For a given graph G=(V,E, w) where $, w : E \to \mathbb{R}^+$

and a real value $d \in \mathbb{R}$

does there exist a spanning tree with weight at most d?

Similarly you can define DShortest path, Dlongest Path etc.

Note: while defining the decision version of an optimization Problem, use at most for minimization Problem and atleast for maximization problem. This definition ensures if
the optimization problem can be solved, then the yes or No answer of the decision problem directly follows.

for example, if we can find a minimum spanning tree with weight M, then the answer of the DMST problem is yes if d<=M, else the answer is NO.
If we use " whether there exist a spanning tree a cost at least d?", then finding the actual minimum spanning tree does not guarantee the correct answer of the decision problem, for d>M.

Polynomial time reduction

let A, B be two decision problems, we say the problem A is Polynomial time reducible to Problem B, If for every input a of the Problem A, it is possible to convert a into an input b of Problem B

in polynomial time such a way that a is an yes input of A off b is an Yes input for B.

The class NP-hard:

A Problem Z is said to be NP-hard, if every problem in NP is polynomial time reducible to Z.

The class NP-complete: A problem is said to be Np -complete if it is in NP and NP-hard