$$DFT\{x(n)+y(n)\}_2 = \sum_{n=0}^{N-1} x(n) W_N^{nk}$$

30/10/2020
FRIDAY

P, NP, NP-complete Problems:-

→ Introduction

[Only there are few useful algorithm, we have deg > 3]

↳ An algorithm solves a problem in polynomial time if its worst case time efficiency belongs to $O(p(n))$ where $p(n)$ is a polynomial of the problem's input size n. Example:- n, $\log n$

↳ Problems that can be solved in polynomial time are called tractable and problems that cannot be solved in polynomial time are called intractable.

Decision problems – Problems which give Yes/No (T/F) as solution

Note:-
Even though knapsack (DP) looks like tractable, when n & W both increases, it can't be solved in polynomial time ⇒ non-tractable.

⇓

[combinatorial problem]

[If a problem can't be solved in polynomial time, then it exponentially varies]

P-Class:-

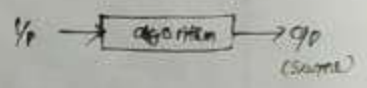
↳ Informally, we can think about problems that can be solved in polynomial time as the set that 'computer science theoreticians call P.

↳ All decision versions of problems ~~that can be solved in polynomial time~~ are in P.

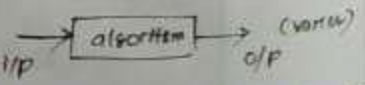
↳ 'Class P' is a class of decision that can be solved in polynomial time by (deterministic) algorithms. This class of problems is called polynomial.

↳ Decidable and undecidable problems

→ Deterministic algorithm:-



Given an algorithm and a particular i/p, what it always produces the same output is called as a deterministic algorithm.



→ Non-deterministic algorithm:- (Eg- Randomized algorithm)

Given an algorithm and a particular i/p, even the I/p same everytime, it gives us different outputs.

[Randomized algo + a particular probability gives a proper output]

⇓

Its behaviour can vary if the algorithm runs multiple times on same input.

31/10/2020
Saturday.

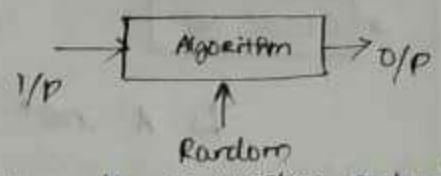
NP PROBLEM ⇒ ① Hamilt
② Travel
③ Knap
④ Part

REDMI NOTE 5 PRO
MI DUAL CAMERA

[Randomised algo + a particular probability gives a proper output]
⇓
Its behaviour can vary if the algorithm runs multiple times on same input.

→ Randomised Algorithm :-

Here in additional to I/P, a set of random numbers are taken, so the instructions executed gets vary Hence O/P varies too. [∴ During execution, it takes random choices depending on those random numbers.]



Even though some problems can't be solved in polynomial time, it may have a decision version Eg:- All Optimisation can't be solved in polynomial. But if it can be reduced to decision problem then it can be solved in polynomial. Eg:- Travel salesman problem
[Its decision version can be solved in polynomial time]

31/10/2020 Saturday.

Decision Problem

Decidable	undecidable.
↓	↓
D. Problems that can be solved by an algorithm in polynomial time	Decision problems that can't be solved by an algorithm in a polynomial time.

→ can't be solved using det. algo in PT

NP PROBLEM ⇒
(combinatorial PROBLEM)
① Hamiltonian circuit problem : Decision counterpart
② Travelling Salesman problem : Decision counterpart
③ Knapsack problem : Decision counterpart
④ Partition problem :- Given 'n' +ve integers, determine whether it is possible to partition them into 2 disjoint subsets with the same sum.

Eg:- {1, 2, 3, 4}
{1, 4} {2, 3}
[sum should be same ↳ (5 = 5) here]

Counter part of Euler circuit ⇓ [Whether Euler circuit exists in a graph (a) not]

EULER CIRCUIT can be solved in polynomial time: O(

[Start from a vertex, visit every edge exactly once and return back to the same vertex] → EULER circuit

Shortcut:- If degree is even for all vertices in a given graph, then we can conclude, that the graph has Euler circuit.

CLASS NP -

↳ A non-deterministic algorithm is a 2-stage procedure that takes as its input an instance I of a decision problem and does the following:-

(i) – Nondeterministic ("guessing") stage:- An arbitrary string S is generated that can be thought of as a candidate solution to the given instance I.

(ii) – Deterministic ("verification") stage:- A deterministic algorithm takes both I and S as i/p and outputs yes if S represents a solution to an instance I. (Cargo used here is in P)

↳ A nondeterministic algorithm solves a decision problem if and only if for every yes instance of the problem it returns a yes on some execution.

– a nondeterministic algorithm to be capable of "guessing" a solution at least once and to be able to verify its validity.

↳ A nondeterministic algorithm is said to be non-deterministic polynomial if the time efficiency of its verification stage is polynomial

↳ Class NP is the class of decision problems that can be solved by Non-deterministic polynomial algorithms. This class of problems is called as non-deterministic polynomial.

↳ $P \subseteq NP$ - Decision Problems

⇒ NP-complete problems :-

↳ A decision problem D_1 is said to be polynomially reducible to a decision problem D_2, if there exists a function t that transforms instances of D_1 to instances of D_2 such that:

- t maps all yes instances of D_1 to yes instances of D_2 and all no instances of D_1 to no instances of D_2.
- t is computable by a polynomial time algorithm

Eg:- Hamiltonian circuit and TSP.

↳ A decision problem D is said to be NP complete if:

- It belongs to class NP
- every problem in NP is polynomially reducible to (D). ↳ A related Decision problem

⇒ CNF - SATISFIABILITY PROBLEM:-

CNF - Conjuction of Disjunction

Eg:- $(a \vee b \vee c) \wedge (\bar{a} \vee \bar{c}) \wedge (\bar{a} \vee b)$

↳ Given a Boolean variable and CNF expression, is there a possibility of assigning truth value to variable and find whether the expression gives the truth (T).

$$\text{DFT}\{c_1 x(n) + c_2 y(n)\} = \sum_{n=0}^{N-1} \{c_1 x(n) + c_2 y(n)\} W_N^{nk}$$

$$= c_1 \sum_{n=0}^{N-1} x(n) W_N^{nk} + c_2 \sum_{n=0}^{N-1}$$

$$= c_1 X_1(n) + \ldots$$

Eg:- $(a \vee b \vee c) \wedge (\bar{a} \vee \bar{b}) \wedge (\bar{a} \vee c)$

Its a NP problem

(i) Take $a = T, b = T, c = F$

⇒ The final Truth value is $T \wedge F \wedge T = (F)$

(ii) Take $a = T, b = F, c = T$

⇒ Final Truth value is $T \wedge T \wedge T = (T)$