Part B

①

cooks theorm states that any NP problem can be converted to SAT in polynomial time

Cooks theorm implies that any NP problem is at most polynomialy harder than SAT.

this means that if we find a way to solve SAT in polynomial time then we will able to solve any NP problem in polynomial time

For example:

Suppose SAT can be converted into problem D in polynomial time.

Now take any NP problem Do. we know that we can convert it into SAT in polynomial time, and we know that we can convert SAT into D in polymial time.

so, we can convert Do in D and so, D will be NP complete.

(2)

Deterministic	non-Deterministic
For a particular input the Computer will give always same output	For particular input the computer will give different output on different execution
Can solve in Polynomial time	can't solve in polynomial time
Can determine the next step of execution	can't determine the next step of execution due to more than one path an Algorithm can take

(3) write any sorting algo you know

(4) A

(4A)

P① 3D

P-②

Algorithm DKP (P, W, n, m, r, x)

{ W:=0, P:=0;

for i:=1 to n do

{ x[i]:= choice(0,1);

W:=W+ x[i] * W[i]; P:=P+ x[i] * P[i];

}

if((W > m) or (P < r)) then Failure();

else success();

}

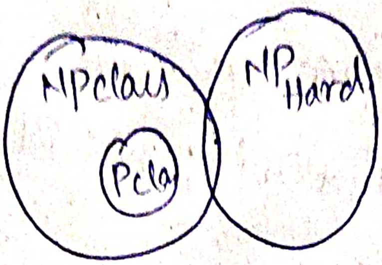
(5)

= every decision problem solved by deterministic 'P' can also be solved using NP.

But every NP will not be a 'P' because we know $\boxed{P \subseteq NP}$ But when P=NP many problems in NP can be proved if P=NP.

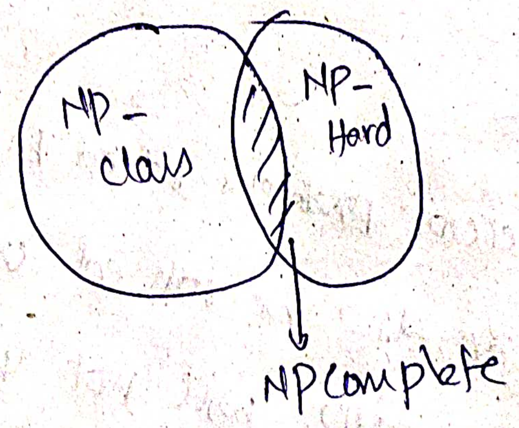
NP Hard

Every problem in NP class can be reduced into other set using polynomial time then it is called NP-Hard



NP complete

- The group of problems which are both NP and NP-Hard are called NP complete



7

(11Q) (10Q) Refer

(12Q) (Q2) Refer

(13Q) Any searching and sorting algorithm you like

(14Q) (Q4) refer

(15Q) (Q5) refer

(16Q) (Q6) refer

(17Q)

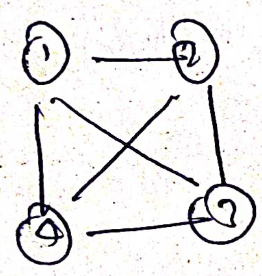
9A) We know that to prove any problem $P = L_2$ as a NP-hard problem first we have to consider an another problem which is NP-hard 'L_1'. And then we have to find a reducability Relation between these two and then we can say that 'L_2' is also NP-hard and so the 'P' also.

So, as we want to prove cliques desicion problem as NP-hard now let us consider a NP-hard problem we consider SAT problem in this case

(7)

clique decision problem

First before understanding what is meant by a clique decision problem first we will understand what is meant by a complete graph:



A complete graph is a graph who have their all adjacent vertices joined with the vertex we are considering that means simply each and every vertex should be connected to each and every other vertex in the graph

And now let us discuss about what is meant by a clique. A clique is a subgraph in some other graph which can form a complete graph.

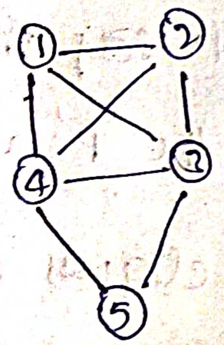
That means a clique is a subgraph in a main graph which forms a complete graph.

And now let us discuss about clique decision problem

we know that if any problem has yes/no type of answer is referred to as decision problem.

<u>For example</u>

Let us consider



Here the total graph is not a complete graph and but the sub-graph 1-2-3-4 form a complete graph so it has a clique 'k' of '4' and the subgraph 5-3-4 is also completegraph so it also have clique of size '3'. And if we consider 4-5 subgraph it is also a completegraph so it also have a clique of size '2'.

So Here we can say that here we have in this graph

* cliques of size = 4, 3, 2

And let us prove cliques ~~decision~~ decision problem as NP-hard

We know that to prove any problem $P = L_2$ as a NP-hard problem first we have to consider an another problem which is NP-hard 'L_1'. And then we have to find a reducability Relation between these two and then we can say that 'L_2' is also NP-hard and so the 'P' also.

So, as we want to prove cliques descion problem as NP-hard now let us consi a NP-hard problem we consider SAT problem in this case

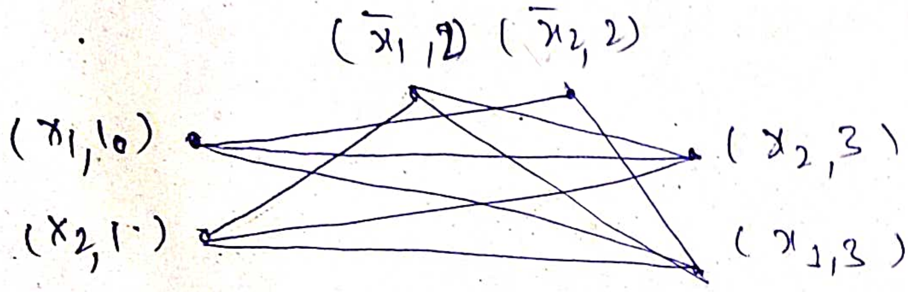
Now let us take a satisfiability formula with three variables x_1, x_2, x_3

$$F = (x_1 \vee x_2) \wedge (\bar{x}_1 \vee \bar{x}_2) \wedge (x_2 \vee x_3)$$

c_1 c_2 c_3

Now we have to prepare a graph of having a clique of size '3'.

Now let us draw the vertices.



So, Here we can see that we have clique of size '3'. $(x_2, \bar{x}_1, \bar{x}_3)$

So, check formula

$$(x_1 \vee x_2) \wedge (\bar{x}_1 \vee \bar{x}_2) \wedge (x_2 \vee x_3)$$

$$(0 \vee 1) \wedge (1 \vee 1) \wedge (1 \vee 0)$$

$$1 \wedge 1 \wedge 1 = 1$$

So, true.

Hence it satisfies so we can now say that cliques decision problem is also np-hard problem