CS325_ HW7

Yihong Liu

1.

$X <=_p Y$, X is just an instance of Y. Y is more complicated than X.

a. False, because X has to be the subset of NP complete.

b. False, because it's possible for Y to be in NP-hard.

c. False, because it's possible for X to be in P as long as X is less complicated than Y.

d. True, yes, because Y is more complicated than X, since X is already NP-complete, it's possible for Y to be in NP- complete.

e. X can be NP-complete if and only if Y is NP-complete also.

f. False, Y can be in NP or harder.

g. True, since X is easier than Y, X can only be P here.

2.

a. No

SUBSET-SUM is NP-complete and can be reduced to any other NP-complete problem, but, the COMPOSITE problem is in NP, and we don't know it that is in NP-complete, so we can't decide the statement.

b. Yes

the given running time is polynomial in 'n', and SUBSET-SUM is NP-complete, which means P = NP, so that every algorithm in NP, would have a polynomial-time algorithm include COMPISITE.

c. No

No, because COMPOSITE is in NP, not know if it is NP-complete, so we can't infer P = NP.

d. No

All problems in P are also in NP, since P is the subset of NP, we know that they all can be solved in polynomial time, proving P is not equal to P would result that NP-complete problems can not be solved in polynomial time, but in fact, NP-complete problems can be solved by polynomial time.

3.

3-SAT, TSP are NP complete, 2-SAT is P.

a.

True, by definition, X can be reduced to Y implied that X is no harder than Y. Because 3-SAT and TSP are both NP complete and can be solved in polynomial time, 3-SAT can be first reduced to DIR-HAM-CYCLE problem then HAM-CYCLE, finally reduced to TSP problem.

b.

False, if P ! = NP, it means there is no polynomial-time algorithm for 3-SAT. However, 3-SAT can be reduced to 2-SAT problem shows that 3-SAT is also in P. which causes contradiction.

c.

True, because if one NP-complete problem can be solved in polynomial time, then all NP problems can be solved in polynomial time, then NP = P, which causes contradiction, so no NP-complete problems can be solved in polynomial time when P != NP.

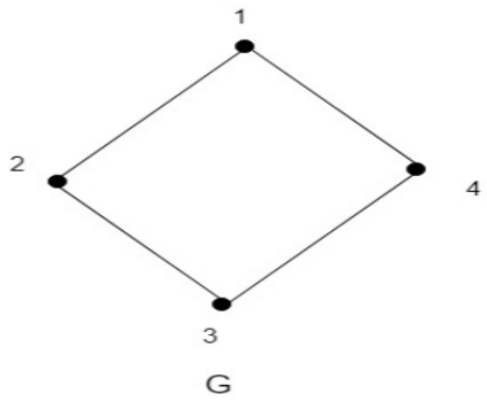
4.

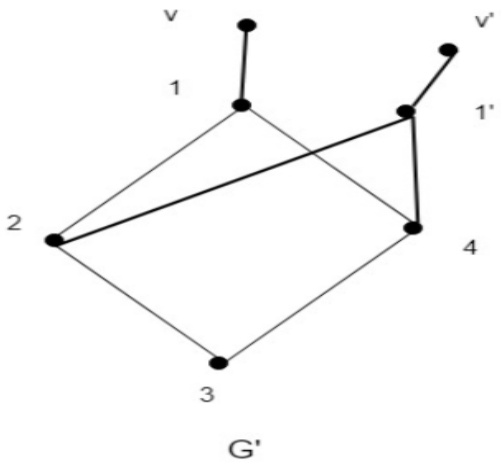
So, for this problem, we first need to prove that there is a ham-path is in NP, then we need to show that there exists a NP-complete problem can be reduced to HAM-path.

First,

Given a graph G' (u,v) that has HAM-path, we can just starting from node u and traverse the path, let's visits each vertex once and reach the ending node v, it can be done is polynomial time for sure, therefore, Ham-path belongs to NP.

Second,





Given the screenshot, referencing the screen shot above, it is known that HAM-CYCLE is NP-complete and can be reduced to HAM-PATH. For HAM-CYCLE to be reduced to HAM-PATH, we can assume that we have the same graph with the same number of edges. A HAMCYCLE in a graph is a just a simple path that visits every vertex exactly once and starts and ends with the same node. From this, we can reduce a HAM-CYCLE to a HAM-PATH is trivial as if there is a HAM-CYCLE in a graph then there must be a HAM-PATH as all nodes are connected and there is a path through the graph where each vertex is visited exactly once. Hence, reducing a HAM-CYCLE to a HAM-PATH can be done and therefore we proved that there is a problem in NP-COMPLETE can be reduced to HAM-PATH.

Hence, due to first and second point, we proved that HAM-PATH is a NP-COMPLETE problem.

5.

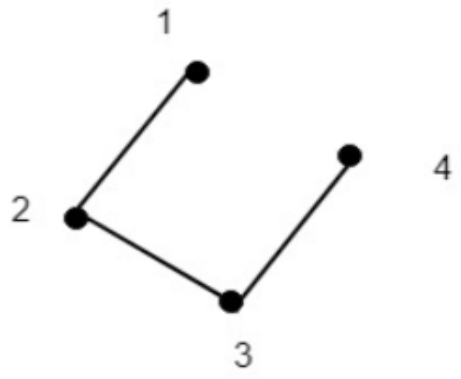
Same with problem 4, we need to first prove that LONGPATH is in NP problem, then we need to prove that there exists a problem can be reduced to LONGPATH.

First,

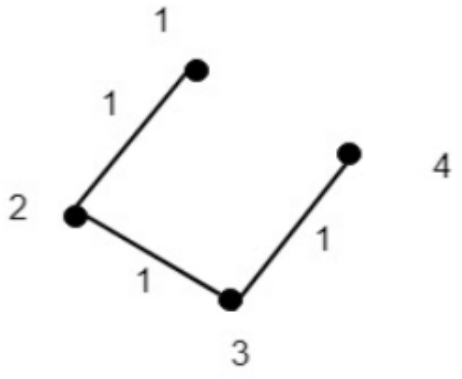
Given a graph G(u,v) that has LONGPATH, we can start from the starting node u and traverse the path and visit each vertex once and reach the ending node v. since it has a simple path in G from u to v of length at least k, it can do done in polynomial time for sure. Therefore, LONGPATH in in NP.

Second, referencing the screen shot below, it is known that HAM-PATH is NP-complete and can be reduced to LONG-PATH. For HAMPATH to be reduced to LONG-PATH, there is the same graph with the same number of edges. From here, we can find the longest path it takes to traverse through all the vertices without repeating an edge. This is very similar with LONG-PATH except we are trying to find a path from u to v that is at least length of k. We can simply make all the edges have a weight of 1 and then try and find a path from u to v that crosses at least k number of edges. Thus, we proved that there is a NP-COMPLETE can be reduced to LONGPATH.

Hence, first and second point proves that LONGPATH is a NP-COMPLETE problem.



G



G'