CS 325 - Homework 7

Student: Joaquin Saldana

1. Let X and Y be two decision problems. Suppose we know that X reduces to Y in polynomial time. Which of the following can we infer? Explain

   a. If Y is NP-complete then so is X.

      Answer: Not true because X could just be in NP and not be NP-complete

   b. If X is NP-complete then so is Y.

      Answer: Not true because Y could be a class harder than NP

   c. If Y is NP-complete and X is in NP then X is NP-complete.

      Answer: Untrue, because X could just be in NP

   d. If X is NP-complete and Y is in NP then Y is NP-complete.

      Answer: True, because Y is in NP and X is in NP complete then Y is at least as hard as X

   e. X and Y can't both be NP-complete.

      Answer: This is a contradiction to 1.d. above, which showed a scenario where both were NP complete

   f. If X is in P, then Y is in P.

      Answer: False, because Y could be in NP and still meet the condition.

   g. If Y is in P, then X is in P.

      Answer: True, because P is the easiest as a result if Y is in P, then X should be in P.

2. Consider the problem COMPOSITE: given an integer *y*, does *y* have any factors other than one and itself? For this exercise, you may assume that COMPOSITE is in NP, and you will be comparing it to the well-known NP-complete problem SUBSET-SUM: given a set S of *n* integers and an integer target *t*, is there a subset of *S* whose sum is exactly *t*?
Clearly explain whether or not each of the following statements follows from that fact that COMPOSITE is in NP and SUBSET-SUM is NP-complete:

   a. SUBSET-SUM $\leq_p$ COMPOSITE.
      Answer: This is not true. Since SUBSET-SUM is NP complete it can be reduced to any other NP complete problem. However, we do not know if COMPOSITE is NP-complete we only know it's in NP. Without knowing if it's in NP-complete we can't confirm if SUBSET-SUM can be reduced to COMPOSITE.

b. If there is an $O(n^3)$ algorithm for SUBSET-SUM, then there is a polynomial time algorithm for COMPOSITE.
Answer: True. Since SUBSET-SUM is NP-complete, if it's solvable in O(n^3) then all NP problems are solvable in polynomial time.

c. If there is a polynomial algorithm for COMPOSITE, then P = NP.
Answer: False, because COMPOSITE is in NP it may be in P, and P is a subset of NP.

d. If P $\neq$ NP, then ***no*** problem in NP can be solved in polynomial time.
Answer: True, because if one problem in NP-complete is solved in polynomial time, then all problems in NP-complete and NP can be solved in polynomial time. The statement proposed in 2.d. is a negation of the statement just presented.

3. Two well-known NP-complete problems are 3-SAT and TSP, the traveling salesman problem. The 2-SAT problem is a SAT variant in which each clause contains at most two literals. 2-SAT is known to have a polynomial-time algorithm. Is each of the following statements true or false? Justify your answer.

a. 3-SAT $\leq_P$ TSP.
Answer: True, per the slide contents from the lecture showing all problems that are NP-complete and polynomial reduce to one another, the 3-SAT problem can be reduced to the directed Hamiltonian cycle problem, to the Hamiltonian cycle problem, to the TSP problem.

b. If P $\neq$ NP, then 3-SAT $\leq_P$ 2-SAT.
Answer: False, because 2-SAT has a solution that runs in polynomial time meaning it's in P. While 3-SAT is NP-complete and if could be reduced to 2-SAT it would mean it's in P and NP-Complete which cannot happen based on the if condition P != NP.

c. If P $\neq$ NP, then no NP-complete problem can be solved in polynomial time.
Answer: True, because if one NP-complete problem can be solved in polynomial time, then all others can be solved in polynomial time as well.

4. A Hamiltonian path in a graph is a simple path that visits every vertex exactly once. Show that HAM-PATH = { (G, u, v): there is a Hamiltonian path from u to v in G} is NP-complete. You may use the fact that HAM-CYCLE is NP-complete
Answer: Per the properties to prove NP-completeness we must prove the following:

a. Show that the Hamiltonian Path problem is in NP
b. Find a known NP-complete problem that can reduced to Hamiltonian Path, in this case we will go with Hamiltonian-Cycle which we know is NP-Complete.

We can prove the problem is in NP because for graph G by choosing edges from G making sure we visit each vertex exactly once. Since this can be performed in polynomial time, the problem is in NP. Now to fulfill the second property, we look at a problem that similar in nature is where a graph contains a Hamiltonian cycle which we know is NP-complete.

We construct a graph, G', such that G contains a Hamiltonian cycle if and only if G' contains a Hamiltonian path. We do this by choosing an arbitrary vertex u in G and adding a copy of u to G', labeled as u' connecting to same vertices as u. We then add vertex v and v' to G'. We get a Hamiltonian path by starting in v in G' following the cycle we got from G back to u' and ending in v'.

By doing this we showed that G contains a Hamiltonian cycle if and only if G' contains a Hamiltonian path proving that Hamiltonian path is NP-complete.

5. LONG-PATH is the problem of, given (G, u, v, k) where G is a graph, u and v vertices and k an integer, determining if there is a simple path in G from u to v of length at least k. Prove that LONG-PATH is NP-complete.
Answer: Per the properties to prove NP-completeness we must prove the following:

a. Show that LONG-PATH is within NP
b. Ensure LONG-PATH is NP-Hard by reducing an existing NP problem.

Given a certificate, a sequence of vertices that make up a path, you can go through each vertex and check its adjacency list in O(n) time. Doing so shows the problem exists in NP. Next, we need to reduce an NP-complete problem to a polynomial time variant. The Hamilton Cycle is NP-complete and we use it to solve G. Then you check each vertex to ensure an each connects to the next vertex. This is performed in O(V+E) time.