CS325- Homework5

Zhouxiang Meng

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.

   This is not true, because X could just be in NP and not necessarily NP-complete.

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

   This is not true because X could be any of the classes harder than NP.

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

   This doesn't have to be true because X could just be in NP.

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

   This is true because Y is definitely in NP and if X is NP-complete then it follows that so is Y because Y is at least as hard as X.

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

   This is not true because there are scenarios where they can be as shown in part d directly above.

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

   This is false because Y is at least as hard as X it could be in NP and still satisfy the condition set.

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

   This is true because there is nothing easier than P therefore if Y is in P then X has to also 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?

a) SUBSET-SUM ≤p COMPOSITE.

   This statement does not hold true. The reason for this is because SUBSET-SUM is NP-complete which then allows us to reduce it to any other NP-complete problem. We do not know if COMPOSITE is also NP-complete only that it resides in NP. Knowing these things, we cannot say definitively that SUBSET-SUM can be reduced to COMPOSITE.

b) If there is an O(n3) algorithm for SUBSET-SUM, then there is a polynomial time algorithm for COMPOSITE.

   This is true because if SUBSET-SUM is in NP-complete therefore if SUBSET-SUM is solvable in polynomial time than all problems in NP-complete are solvable in polynomial time.

c) If there is a polynomial algorithm for COMPOSITE, then P = NP.

   This is false because we are told that COMPOSITE is within NP not necessarily within NP-complete therefore if COMPOSITE can be solved in polynomial time than it could be a matter of COMPOSITE being misclassified like the algorithm used to find prime numbers.

d) If P $\neq$ NP, then no problem in NP can be solved in polynomial time.

   This statement is true because it follows that if one problem in NP-complete is solvable in polynomial time than all problems in NP-complete and NP are solvable in polynomial time. The assertion stated just before is simply the negation of the statement in d.

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 ≤p TSP.

   This statement is true because as observed in the lecture slides with the NP-completeness graph that 3-SAT can be reduced from itself to DIR-HAM-CYCLE to HAM-CYCLE to TSP.

b. If P ≠ NP, then 3-SAT $\leq_p$ 2-SAT.

   This statement is false because 2-SAT has a solution that runs in polynomial time and 3-SAT is NP-complete so it doesn't follow that P ≠ NP because 2-SAT is within P so if you could reduce 3-SAT to 2-SAT it would also be in P but also still be within NP-complete which causes a contradiction with P ≠ NP.

c. If P ≠ NP, then no NP-complete problem can be solved in polynomial time.

   This statement is true because by definition of NP-complete if at least one NP-complete problem can be solved in polynomial time then it follows that all NP-complete problems can be solved in polynomial time.

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

   We aim to show that the language HAM-PATH can be verified in polynomial time. Let the input x be ⟨ G, u, v⟩ and let the certificate y be a sequence of vertices {v1, v2, · · · , vn}. An algorithm A(x, y) verifies HAM-PATH by executing the following steps:

(a) Check if $|G.V| = n$;

(b) Check if $v_1 = u$ and $v_n = v$;

(c) Check if $\forall_i \in \{1,2,\cdots,n\}$, $v_i \in G.V$;

(d) Check if $\forall_{i,j} \in \{1,2,\cdots,n\}$, $v_i \neq v_j$;

(e) Check if $\forall_i \in \{1,2,\cdots,n-1\}$, $(v_{i,}v_{i+1}) \in G.E$;

If any of the above step fails, return false. Else return True. Steps (a) and (b) takes O(1) time; step (c) takes O(V) time; step (d) runs in $O(V^2)$ time and step (e) runs in O(E) time. Therefore, the verification algorithm runs in $O(V^2)$ time. Hence HAM-PATH $\in$ 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.

LONG-PATH is within NP since the verifying of the path is the yes-no which can be checked in polynomial time that it is a path and it is at least k units long. LONG-PATH is also NP-complete because a Hamiltonian path is a special variant of LONG-PATH where we state the start, end node and k equals the number of vertices in the path minus l.
First we show that LONG-PATH is verifiable in polynomial time. Given a sequence of vertices that make up the path we can traverse that sequence in O(n) time and check the adjacency list that the next vertex is within that list, thus showing that this portion belongs in NP. The second part is to reduce an NP-complete problem to a polynomial time variant. Hamiltonian cycle detection is NP-complete. If you take the graph passed into LONG-PATH and solve it using Hamiltonian path, then all that would remain would be to check every vertex that results to make sure there exists an edge to the subsequent vertex. That process can be completed in O(V+E) time.