CS352 (Winter 2018) - Homework 7

Marc Tibbs (tibbsm@oregonstate.edu)

Due Date: March 4, 2018

Problem 1: *(7 points)* 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.
b) If X is NP-complete then so is Y.
c) If Y is NP-complete and X is NP then X is NP-complete.
d) If X is NP-complete and Y is NP then Y is NP-complete.
e) X and Y can't both be NP-complete.
f) If X is in P, then Y is in P.
g) If Y is in P, then X is in P.

We can infer statements (d) and (g).

Both (a) and (b) cannot be inferred since we do not have enough information to deduce whether X or Y is NP-Complete. By definition, we know that if $X \leq_p Y$ and X is NP-Complete then Y is NP-Hard. Additionally, if $Y \in NP$, then Y is NP-Complete. Therefore, (d) is a true statement. This does not hold for statement (c) since we only know that X reduces to Y and not vice versa. Also, by definition of X $\leq_p$ Y, if X $\in$ P, then Y $\in$ P. Therefore (g) is also true.

Problem 2: *(4 points)* 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.
No, we do not know if COMPOSITE is NP-complete, so we cannot be sure that SUBSET-SUM reduces to COMPOSITE.

b) If there is an $\mathcal{O}(n^3)$ algorithm for SUBSET-SUM, then there is a polynomial time algorithm for COMPOSITE.
Yes, since we know that SUBSET-SUM has a polynomial run time and is NP-complete, we can infer that every algorithm in NP would also have an algorithm that runs in polynomial time.

c) If there is a polynomial algorithm for COMPOSITE, the P = NP.
No, we do not know if COMPOSITE is NP-Complete so we cannot assume that P = NP.

d) If P $\neq$ NP, then no problem in NP can be solved in polynomial time.
No, P is a subset of NP and we know that there are problems in P that are solved in polynomial time. If P $\neq$ NP, then we can only infer that **NP-Complete** problems can't be solved in polynomial time.

Problem 3: *(3 points)* 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.
True. A reduction from any NP-Complete problem can be made to any other such NP-Complete problem.

b) If P $\neq$ NP, then 3-SAT $\leq_p$ 2-SAT.
False. 3-SAT $\leq_p$ 2-SAT would imply that P = NP, since we know 3-SAT is NP-Complete and 2-SAT has a polynomial-time algorithm.

c) If P $\neq$ NP, then no NP-complete problem can be solved in polynomial time.
True. A polynomial-time algorithm for one NP-complete problem would mean that there are polynomial-time algorithms for all other NP-complete problems. Therefore, either P = NP and all NP-complete algorithms can be solved in polynomial-time or *$P \neq N$ and no NP-complete problem can be solved.*

Problem 4: *(6 points)* 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.

(Proof)

Step 1: Show that HAM-PATH is in NP.

Given an instance of a problem the certificate is a graph G with vertices u and v. The certifier (verification algorithm) checks every vertex to make sure that each is visited exactly once. This procedure can be done in polynomial-time. Therefore, we can say that HAM-PATH is NP.

Step 2: Prove that HAM-PATH is NP-Hard. We can show that HAM-CYCLE $\leq_p$ HAM-PATH and HAM-CYCLE $\in$ NP-Complete.

To reduce from HAM-CYCLE to HAM-PATH we can create a graph G' from a graph G and add vertices u' and v' that connect only to the original vertices u and v. Suppose that graph G has a

Hamiltonian cycle with an edge **p** from vertex u to v. Each vertex in **p** is in G'. Thus **p** is a path in G' and can be extended to include u' and v' and remain a Hamiltonian path. Conversely, if we have a graph G' that has a HAM-PATH from u' to v' then there is also a HAM-CYCLE that uses the edge between u and v.

Similarly, if graph G does not have a Hamiltonian cycle, then we can conclude that graph G' also does not have a Hamiltonian path. And, if graph G' does not have a Hamiltonian path between u' and v' then a there can be no Hamiltonian cycle between u and v.

Therefore, we can conclude that HAM-PATH is NP-Hard. Furthermore, Since we know that HAM-CYCLE is NP-Complete, we can also conclude that HAM-PATH is NP-Complete.

Problem 5: *(5 points)* 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.

(Proof)

Step 1: Show that LONG-PATH is in NP.

Given an instance of a problem the certificate is a graph G with vertices u and v and an integer k. The certifier (verification algorithm) checks every vertex to make sure that each is visited exactly once and verifies that the path is at least a length of k. This procedure can be done in polynomial-time. Therefore, we can say that LONG-PATH is NP.

Step 2: Prove that LONG-PATH is NP-Hard. We can show that HAM-PATH $\leq_p$ LONG-PATH and HAM-PATH $\in$ NP-Complete.

To reduce from HAM-PATH to LONG-PATH we can create a graph G' from a graph G. Suppose that graph G has a Hamiltonian path. Each vertex is in the HAM-PATH in G is also in the LONG-PATH in G'. Since a HAM-PATH requires every vertex t be visited exactly once, the longest path is necessarily one where all edges are also visted giving a length of the number of vertices - 1.

Therefore, we can conclude that LONG-PATH is NP-Hard. Furthermore, Since we know that HAM-PATH is NP-Complete, we can also conclude that LONG-PATH is NP-Complete.

Extra Credit: *(5 points)* The Traveling Purchaser Problem (TPP):

The travelling purchaser travels from marketplace to marketplace searching for goods on his list. The travelling purchaser always returns to his "home" marketplace with all the goods on his list and having spent as little money as possible.

Given a list of marketplaces, the cost of travelling between different marketplaces, and a list of available goods together with the price of each such good at each marketplace, the task is to find for a given "shopping list" of goods the purchasing route with the minimum combined cost of purchases and traveling. The purchasing route solution will indicate both the order the marketplaces are visited and the goods that are purchased at each marketplace. The purchaser must start and finish at the same marketplace and visit each other marketplace at most once (it is not required that all marketplaces be visited).

The decision version of TPP-D would ask if there exists a route for a set of marketplaces, travel costs, list of goods, and costs of goods such that the total cost of purchases and travelling is at most k.

Formally TPP-D = { (G, C, L, A, P, k) : where
 G=(V,E) where the vertices are marketplaces,
 E is the set of edges which represent the road between the marketplaces,
 C(u,v) is the cost of travelling from marketplace u to v,
 L is the shopping list of goods that the purchaser wants to buy,
 A(v) is the list of goods available at marketplace v and
 P(x,v) is the price of the good x at marketplace v.
G has a TPP route with total cost at most k }.

Prove that TPP-D is NP-Complete

(Proof)

Step 1: Show that TPP-D is in NP.

Given an instance of a problem the certificate is a path of the visited marketplaces in order and list of the goods that are purchased at each marketplace. The certifier (verification algorithm) checks every edge in the given path to make sure that it is an edge in the graph and verifies that the items bought at each location are available. Finally, the algorithm checks to make sure that the tour ends where it started and that the total cost is at most k. This procedure can be done in polynomial-time. Therefore, we can say that TPP-D is NP.

Step 2: Prove that TPP-D is NP-Hard. We can show that TSP $\leq_p$ TPP-D and TSP $\in$ NP-Complete.

To reduce from TSP to TPP-D we can create a graph G' from a graph G with additional vertices connected to the original vertices. Each edge that is added is given a weight of infinity, so we will nkow if any new vertices are visited. Each vertex in the TSP in G is also in the TPP-D in G'. Since the TSP is a version of the TPP-D where all marketplaces must be visited once, the path is necessarily one where all necessary marketplaces are also visted in the TPP-D.

Therefore, we can conclude that TPP-D is NP-Hard. Furthermore, Since we know that TSP is NP-Complete, we can also conclude that TPP-D is NP-Complete.