Homework 7

CS 325

Daniel Kim

1. *(7 pts)* 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 in NP then X is NP-complete.

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

a) **Statement: If Y is NP-complete then so is X.** The statement is **false**.
   - Since, for NP complete, X has to be the subset of NP complete.

b) **Statement:** If Y is NP-complete then so is X. The given is false.
   - Since, for NP complete, X has to be the subset of NP complete

c) **Statement:** If X is NTP-complete then so is Y. The is false.
   - The problem Y cannot be NP Complete, since X is reducible to Y, not Y is reducible to X.

d) **Statement:** If Y is NP-complete and X is in NP then X is NP-complete.
   - The given statement is false. Since, X is reducible to Y, so X can be efficiently solved by Y, but not vice-versa for the given relation of X and Y.

e) **Statement:** If X is NP-complete and Y is in NP then Y is NP-complete.
   - **Explanation:** The given statement is true. Since, the problem X reduces to problem Y so that if here is a black box to solve Y efficiently, then that can be used to solve the problem X efficiently. X is easier than Y.

f) **Statement:** X and Y can't both be NP-complete.
   - The given statement is false. X can be NP-complete if an only if Y is NP-complete.

g) **Statement:** If X is in P, then Y is in P.
   - The given statement is false. The problem X is reducible to Y, therefore, if Y is P then X is in P.

h) **Statement:** If Y is in P, then X is in P.
   - The given statement is true. Since, the problem X reduces to problem Y so that if there is a black box to solve Y efficiently, then that can be used to solve the problem X efficiently. X is easier than Y. **Therefore, per the relation of the problem X and Y only the d. and g. options can be inferred.**

2. *(4 pts)* 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 ≤p COMPOSITE.

b. If there is an $O(n^3)$ algorithm for SUBSET-SUM, then there is a polynomial time algorithm for COMPOSITE.

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

d. If P ≠ NP, then ***no*** problem in NP can be solved in polynomial time.

- **The given statement does not follow from the fact.**
  - Given that SUBSET-SUM problem is "NP-complete", it can be reduced to another "NP-complete" problem.
  - The COMPOSITE problem is in "NP", but it is not certain that the COMPOSITE problem is in "NP-complete".
  - Thus, it is not definite that the SUBSET-SUM problem reduces to the COMPOSITE problem.
- **The given statement follows.**
  - $O(n^3)$ algorithm is polynomial in "n" , and SUBSET-SUM is "NP-complete". As a result, it follows that "P = NP".
  - In addition, problems in "NP" will have polynomial-time algorithm.
  - Hence, polynomial time algorithm exists for COMPOSITE problem.
- **The given statement never follows.**
  - The COMPOSITE problem is in "NP". It is not definite that the COMPOSITE problem is in "NP-complete". If the COMPOSITE problem has polynomial algorithm, the argument "P = NP" never follows.
- **The given statement does not follow.**
  - We know that "P" is the subset of "NP", and P is not empty.
  - Attempting to prove that "P" is not equivalent to "NP", we will get problems in "NP-complete" that cannot be solved in polynomial-time. However, problems in "NP-complete" can have polynomial-time algorithm.

3. *(3 pts)* 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.

b. If P ≠ NP, then 3-SAT ≤p 2-SAT.

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

a) 3-SAT ≤p TSP. Statement is true.

- If one NP-Complete problem can be solved in polynomial time, then all NP problems can be solved in polynomial time. 3-SAT is NP-Complete and TSP is also NP Problem. Thus, it can be solvable in polynomial time.

b) If P is not equal to NP, then 3-SAT ≤p 2-SAT. Statement is false.

- Suppose that P ≠ NP and that there exists a polynomial-time reduction from 3-SAT to 2-SAT.
- Then, 2-SAT is NP-complete, but 2-SAT is also solvable in polynomial time.
- Thus, for all decision problems where A ∈ NP, one can decide x ∈ A reducing *x* to the corresponding 2-SAT *instance* in polynomial time, and then proceed to solve it in polynomial time.
- As a result, A∈ P and NP⊆ P. But we also have P⊆NP, so P=NP which is a contradiction.

c) If P is not equal to NP, then no NP-complete problem can be solved in polynomial time. Statement is true.

- If one NP-Complete problem can be solved in polynomial time, then all NP problems can be solved in polynomial time.
- If that is the case, then the NP and P set become the same which contradicts the given the statement.

4. *(6 pts)* 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

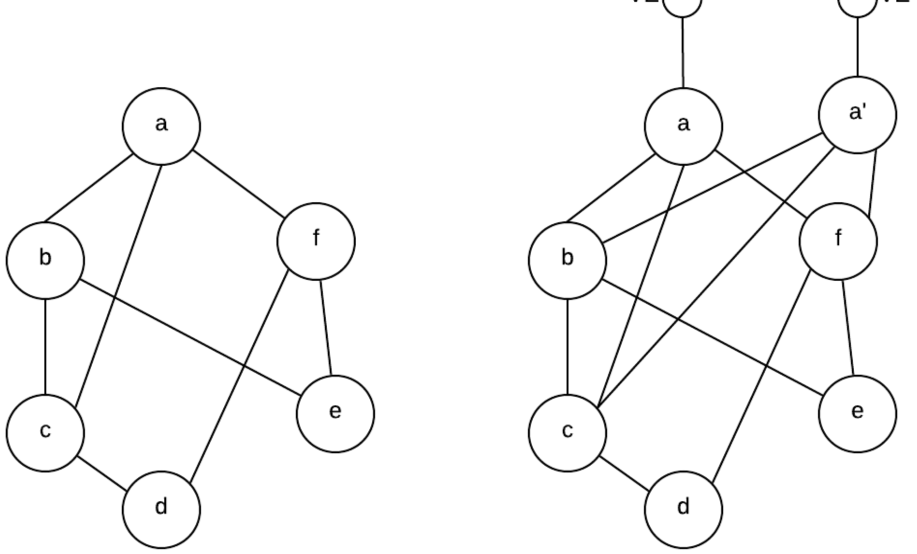
To prove any problem is **NP-complete** the process is as follows:

- **NP-complete** problem is solved m polynomial time
- **NP-complete** problem is reducible to any other NP-complete problem
- A Hamiltonian path is a simple open path that contains each vertex in a graph exactly once. The Hamiltonian Path problem is a problem that determines whether a given graph contains a Hamiltonian path.

- **To show that the Hamiltonian path problem is NP-complete.** Solve the problem in polynomial time and reduce an NP-complete problem to the Hamiltonian path problem. As a result we show that the Hamiltonian path problem is NP-complete.
- **In order to prove that any other NP-Complete problem is reducible to Hamiltonian path problem,** is the following.
    - Given that a Hamiltonian cycle problem is NP-complete.
    - In a Hamiltonian cycle, the path begins and ends at the same vertex.
    - If the Hamiltonian cycle problem is reducible to the Hamiltonian path problem, then the Hamiltonian path problem is said to be an NP-complete problem.

  We can represent this as follows:
- **To prove that there exists a Hamiltonian cycle if only there exists a Hamiltonian path in a graph, this proves the Hamiltonian path problem is NP-complete.**
- Given a graph GR = <VE, ED>, VE are the vertices and ED are the edges of the graph. Then construct a graph GR' such that GR contains a Hamiltonian cycle if and only if GR' contains a Hamiltonian path. Construct both the graphs as shown (Figure 1).

Figure 1. GR and GR' respectively



- For Graph GR (left side graph), the Hamiltonian **cycle is a, b, e, f, d, c, a.**

- In GR' (right side graph) the corresponding Hamiltonian path is **VE, a, a, b, e, f, d, c, a’, VE**
- Thus, we have shown that a graph contains a Hamiltonian cycle if only a graph also contains a Hamiltonian path. **As a result, the Hamiltonian path problem is an NP-Complete problem.**

5. *(5 pts)* 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 LONGPATH is NP-complete.

- Long Path is in NP since the path is the certificate.
- We can easily check in polynomial time that it is a path, and that its length is k or more and NP-complete since a Hamiltonian Path (where we specify a start and end node) is a special case of a Long Path. It is the Long Path where k = (# of vertices of G) – 1