James Palmer

Homework 5 – CS 325

1)

a) If Y is NP-complete, and we can reduce X to Y in polynomial time, then X is also NP-complete

b) If X is NP-complete, but can be reduced to Y in polynomial time, then Y has to be NP-complete, else X would be P

c) Y reduces to X means that if you had a black box to solve X efficiently, you could use

it to solve Y efficiently. Y is no harder than X.

d) X reduces to Y means that if you had a black box to solve Y efficiently, you could use

it to solve X efficiently. X is no harder than Y.

e) Either X or Y can be reduced in polynomial time, then one can be NP while the other cannot be NP and vice versa.

f) Y reduces to X means that if you had a black box to solve X efficiently, you could use

it to solve Y efficiently. Y is no harder than X.

g) X reduces to Y means that if you had a black box to solve Y efficiently, you could use

it to solve X efficiently. X is no harder than Y.

2)

a) No. SUBSET-SUM is NP-complete and so may be reduced to any other NP-complete problem. However, we don’t know that COMPOSITE is NP-complete, only that it is in NP. Hence, we cannot say for sure that SUBSET-SUM reduces to COMPOSITE.

b) Yes. The given running time is polynomial in n and log t. Since SUBSET-SUM is NP-complete, this implies P = NP. Hence, every algorithm in NP, including COMPOSITE, would have a polynomial-time algorithm.

c) No. COMPOSITE is in NP, but it is not known to be NP-complete. Hence, a polynomial-time algorithm for COMPOSITE does not imply P = NP.

d) No. The class P is a subset of NP, and it is clearly not empty! Proving P *≠* NP would show only that NP-complete problems cannot be solved in polynomial time.

3)

a) True. There exists a reduction from any NP-complete problem to any other such problem.

b) False. If P *≠* NP, there is no polynomial-time algorithm for 3 -SAT. However, 2 -SAT is known to be in P; if the reduction existed, it would imply a polynomial-time algorithm for 3 -SAT.

c) True. A polynomial-time algorithm for one NP-complete problem yields polynomial-time algorithms for all others. Hence, either all these problems are in P, or none are. P*≠* NP implies the latter.

4) Long Path is in NP since the path is the certificate. Since we can easily check in polynomial time that it is a path, and that its length is k or more, and NP-complete since Hamiltonian Path (the variant where we specify a start and end node) is a special case of Long Path, namely where k equals the number of vertices of G minus 1.

5) a) If there exists 2 - coloring of a graph and suppose we have two colors: 0 and 1. Just randomly choose one vertex V0 and color it as black or white, then do a modified Depth - First - Search to color all other vertices. b) The decision problem is: Given an undirected graph G and an integer k, can we color G with k colors such that adjacent vertices have different colors? 1.) If we have solution to this problem in polynomial time, we can assign k values from |V| down to 2 and check if it is colorable, and stop when k reaches some value that it is not colorable. Then we choose the minimum number from all those numbers make the graph colorable, that is the number we want in Graph-coloring problem. It is easy to know the time is still polynomial. 2.) If Graph-coloring problem is solvable in polynomial time, then we know the minimum number of colors needed, say l, then we can just compare the given number k in decision problem with l, if k < l.

b) To prove that 3-COLOR is NP-complete, we use a reduction from 3-CNF-SAT. Given a formula " of m clauses on n variables x1, x2,..., xn, we construct a graph G D .V; E/ as follows. The set V consists of a vertex for each variable, a vertex for the negation of each variable, 5 vertices for each clause, and 3 special vertices: TRUE, FALSE, and RED. The edges of the graph are of two types: “literal” edges that are independent of the clauses and “clause” edges that depend on the clauses. The literal edges form a triangle on the special vertices and also form a triangle on xi, :xi, and RED for i D 1; 2; : : : ; n.