HW7 – Trevor Stahl stahltr@oregonstate.edu

ANALYSIS OF ALGORITHMS (CS_325_400_W2019)

1. (6 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.

X in polynomial time reduces to Y means that X is no harder than Y.

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

False, we only know that X reduces to Y which does NOT indicate that if Y is NPC then X must be NPC. X might not be NP.

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

False, we only know that X reduces to Y which does NOT indicate that if X is NPC then Y must be NPC. Y might not be NP.

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

False, we only know that X reduces to Y which does not necessarily imply the opposite.

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

True, NPC is the hardest, X reduces to Y so X is easier than Y AND Y is NP so we can say Y is NPC.

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

False, X is easier than or equivalent to Y, which means Y can be harder than X, Y might not be P.

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

True, X reduces to Y in P time, so X is easier or similar to Y, P is the easiest subset with NP, So if Y is P and is harder or equal to X then X must be P.

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 follow from that fact that COMPOSITE is in NP and SUBSET-SUM is NP-complete:

a. SUBSET-SUM ≤p COMPOSITE.

Does NOT follow

This does not make sense because SUBSET-SUM is NPC. A NPC problem does not necessarily reduce in P time to a NP problem.

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

FOLLOWS

Big Oh of N cubed is polynomial and with a small K. SUBSET-SUM is said to be NPC. So if we have a P algorithm for an NPC problem this means there is a P algorithm for COMPOSITE because all NP problems can reduce to an NPC problem.

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

Does NOT follow

We only know that COMPOSITE is within NP. To collapse all NP into P(P = NP) we would need an NPC problem to have a P solution for P = NP

d. If P does NOT equal NP, then no problem in NP can be solved in polynomial time.

Does NOT follow

P is within NP and P problems can “be solved in polynomial time”.

3. (8 pts) A Hamiltonian path in a graph is a simple path that visits every vertex exactly once. Prove 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.

LEMMA 1 – Show that HAM-PATH is within NP

a) Show that a solution to HAM-PATH can be verified in polynomial time

(Give a polynomial time algorithm that verifies an instance of HAM-PATH in polynomial time)

Given a graph G = (V, E) a certificate would come in the form of |V| vertices that possibly make up a HAM-PATH within graph G. To verify or certify the certificate, the verification algorithm would first check that every vertex in the certificate is in G. Then it would verify that the number of vertices in the certificate is equal to number of V in G. Most importantly it would next verify that there is an edge between each vertex, such that the certificate is plausible. And that no vertex is listed twice.

This would run in P time at most.

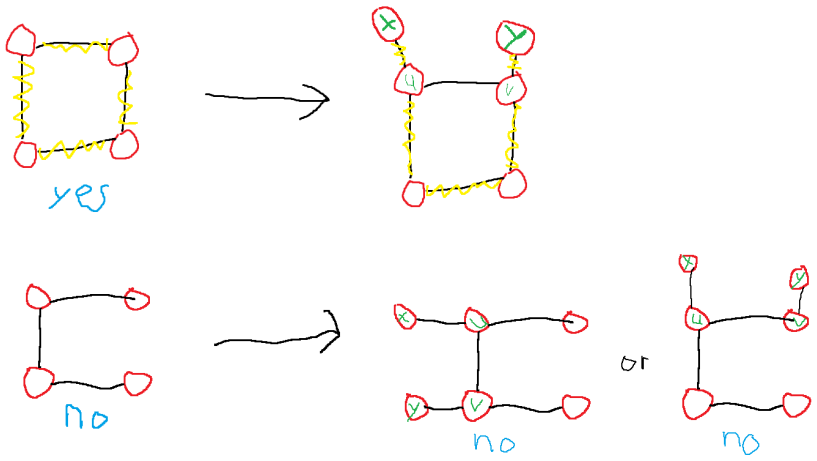
LEMMA 2 – Show that Some NPC problem R reduces in P time to HAM-PATH

a) Choose a known NP complete problem that can reduce to HAM-PATH

HAM-CYCLE – question allows the use of “the fact that HAM-CYCLE is NP-complete”

b) describe transformation that maps instances of HAM CYCLE (known NPC problem) to HAM-PATH

Let G = (V, E) be any graph at all. Let G’ be equivalent to G aside from following modifications. Let u and v be two neighboring vertices connected with an edge within G. Let x and y be two additional vertices added, not from G, that each have a degree of one attaching to u and v respectively.



~~Let G = (V, E) be any graph at all. Let s be any vertex in G. For each neighbor of S attempt HAM-PATH on G' where G' is G but with the edge connecting S and one of its neighbors removed.~~

~~Let G = (V, E) be any graph at all. Now we construct a second graph G' = (V', E'). For every edge, e_i, within E let G_{ei} be a graph with all vertices V and edges E – e_i, each edge we are examining. That is for each edge let there be a graph excluding said edge but with all V. Also, e_i will have form (v_i, u_i). Next we form G' by adding to it a copy of each G_{ei} for every e_i. Finally, G' also has an additional vertex x connected by an edge to u_1, arbitrarily chosen. And G' has edges connecting v_i to u_{i+1} for 1<= i <= |E| - 1 and a vertex y and edge from $v_{|E|}$ to y.~~

c) Prove the transformation work correctly

If G had a cycle there must be a HAM-PATH from x to y in G', and if G did not have a cycle there cannot be a HAM-PATH from x to y in G'. This is because for G to have a cycle you must be able to reach all vertices in a path from u to v. And given that we said that there is an edge from u to v there is a cycle. Adding the neighboring appendages with vertices x and y means that there is now not a cycle in the entire graph but a sub-graph cycle that can be used in part to form a HAM-PATH from x to y.

If G' did not have a HAM-PATH from x to y, this would mean that G does not have a simple path to make a HAM-PATH out of alongside the edge definitionally stated as existing between u and v.

Path for G' relies on a cycle in G. And vice versa.

~~If G has a HAM-CYCLE, then each G_{ei} has a HAM-PATH from u_i to v_i for each i because the graph would be the same graph but with a missing edge that G uses as the last edge for the cycle. This means that, G' has a HAM-CYCLE from x to y via taking each of these paths after each other. If G does not have a HAM-CYCLE, then each G_{ei} would not have a path~~

d) Prove the transformation runs in polynomial time

The transformation runs in constant time because the number of vertices and edges to add is always 2 for each and then at most a few traversals to find neighbors that share an edge. O(1)

Because both Lemma 1 and 2 are true then HAM-PATH is NPC, according to NPC proof template.

4. (12 pts) K-COLOR. Given a graph G = (V,E), a k-coloring is a function c: V -> {1, 2, … , k} such that c(u) does not equal c(v) for every edge (u,v) □ E. In other words the number 1, 2, .., k represent the k colors and adjacent vertices must have different colors. The decision problem K-COLOR asks if a graph can be colored with at most K colors.

a. The 2-COLOR decision problem is in P. Describe an efficient algorithm to determine if a graph

has a 2-coloring. What is the running time of your algorithm?

Input is G(V, E)

While there is a vertex V left in G do the following {

If all vertices in V are uncolored then pick one somehow (randomly, or based on input form) and color it with red.

While any vertex v in G is colored do the following {

For each vertex connected to v referred to as v' do the following {

If v'.color == v.color then return NO

Else if v.color == red then v'.color = blue

Else v'.color = red

}

Remove v and dependent edges from G

}

}

Return yes

- IN WORDS ONLY -

Start at any vertex in G and color it Red. Continue by following a breadth first search where each new vertex is set an alternating color starting with Blue because the source was set red. If there is ever a contradiction return No if not return Yes once done with BFS. A contradiction being that something is already colored a color which is NOT the color it should be colored based on current position in graph search.

This would work using a depth first search as well but possibly slower.

b. It is known that the 3-COLOR decision problem is NP-complete by using a reduction from SAT.

Use the fact that 3-COLOR is NP-complete to prove that 4-COLOR is NP-complete.

1) Show that 4-Color is within NP.

Give a polynomial time algorithm that verifies an instance of Qin polynomial time. In other words, if I give you an "answer" can you verify it polynomial time.

Input: G =(V, E, c) – a graph with vertices V edges E and colors for each vertex

First check that the total number of different colors c is less than or equal to 4

Second, color each node of an actual G as specified by the input

Next, for each node check that all neighbors have a color not equal to its own

If all nodes are different from neighbors then return Yes for decision, otherwise return No upon encountering first pair of neighboring vertices with same coloring.

This would happen in polynomial time, specifically it would run at most O(n*(n-1)). This occurs not only in a graph that meets the criteria and should return yes but in a graph that also is complete with all vertices connected to every other. This is the worst-case scenario and is polynomial so 4-color is in NP.

2) Show that 3-Color ≤P 4-Color

Show a polynomial algorithm to transform an arbitrary instance x of 3-Color into an instance of 4-Color

Let a graph be G =(V, E, c) with vertices V, edges E and color assignments c for each vertex.

Let G' be equivalent to G aside from the following modifications. G' contains an additional vertex which is connected to every other vertex in G and G' with a number of edges equal to number of vertices in G. Color this additional vertex the fourth new color and disallow the fourth color to be used in any other vertex.

Prove that this transformation holds such that 3-Color is yes iff and only if 4-Color is yes, no if and only if 4-color is no and vice versa.

G' will only pass 4-Color if G passes 3 color because the transformation from G to G' meant that the only uncertainty in whether G or G' pass their respective decision problems is completely enclosed within whether G passes 3-Color. This is because the additional vertex is colored the fourth color and no other vertices are colored that. None of the added edges in G' can possibly cause conflicts in colors. And because of limiting the fourth color to be used once only for the one additional vertex for G' there is no case which breaks the transformation.

Because 1) and 2) are true 4-Color is in NP-Complete