# CS 325 - Homework 4 - Solutions

1. **( 7 points total 1pt each)** 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, B, C, E & F- False can not be inferred

D. True: If X is NP-complete and Y is in NP then Y is NP-complete.

G. True: If Y is in P, then X is in P.

2. **( 4 points 1 pt each** 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 $\le_p$ COMPOSITE.

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.

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

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

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

No. COMPOSITE is in NP, but it is not known if it is in NP-complete.

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

No. All problems in P are also in NP and can be solved in polynomial time. Proving P 6= NP would show only that NP-complete problems cannot be solved in polynomial time.

3. **(3 points, 1 pt each)** 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 $\le_p$ TSP.

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

b. If P $\neq$ NP, then 3-SAT $\le_p$ 2-SAT.

False. If P $\neq$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. 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 yields polynomialtime algorithms for all others. Hence, either all these problems are in P, or none are. P 6= NP implies the latter.

4. **(4 points total)** 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. Show that LONG-PATH is NP-complete.

**Step 1:** **(2 points)** LONG-PATH is NP. A certificate showing a pair (G,k) to be in LONG-PATH is simply a listing of the sequence of vertices in a path of length k.

- If k $\leq$ n-1, a poly-time verifier can check that the sequence is a valid path in G, is simple, and has length k.
- If k > n-1, there can be no simple path with k edges (as it would have k+1 > n vertices on it), and we can reject the input whatever certificate is offered.

**Step 2:** **(2 points)** LONG- PATH is NP-hard and thus NP-complete. We can reduce HAM-PATH, proved in problem 3. to be NP-complete, to LONG-PATH. An input to HAM-PATH is a graph G, and G is in HAM-PATH if and only if there exists a simple path in G containing all n vertices and thus having exactly n-1 edges. So G is in HAM-PATH if and only if the pair (G,n-1) is in LONG-PATH, and clearly this reduction can be computed in polynomial time. We have that HAM-PATH $\leq_p$ LONG-PATH, and thus LONG-PATH is NP-hard.

Since 1) and 2) are true LONG-PATH is in NP-Complete.

5. **(7 points total)** Graph-Coloring. Mapmakers try to use as few colors as possible when coloring countries on a map, as long as no two countries that share a border have the same color. We can model this problem with an undirected graph G = (V,E) in which each vertex represents a country and vertices whose respective countries share a border are adjacent. A k-coloring is a function c: V -> {1, 2, … , k} such that c(u) $\neq$ c(v) for every edge (u,v) $\in$ E. In other words the number 1, 2, .., k represent the k colors and adjacent vertices must have different colors. The graph-coloring problem is to determine the minimum number of colors needed to color a given graph.

a. **(total of 3 points)** State the graph-coloring problem as a decision problem K-COLOR.

<u>**Decision Problem:**</u> **K-COLOR** : Given a graph G= (V,E) is there a way to color the vertices with exactly k colors such that adjacent vertices are colored differently? **( 1 point)**

***Alternative definition**: **K-COLOR** : Given a graph G= (V,E) is there a way to color the vertices with at **most** k colors such that adjacent vertices are colored differently? Full credit*

Show that your decision problem is solvable in polynomial time if and only of the graph-coloring problem is solvable in polynomial time. **( 2 points if and only if proof)**

**CS 325 - Homework 4 - Solutions**

If we can solve K-COLOR for a given graph G in polynomial time, then the graph-coloring problem can be solved in polynomial time by assigning K from 1 to |V |and checking if the answer to K-COLOR is YES. As soon as we get the answer YES for K-COLOR we can stop since K is the minimum of any coloring. K is the solution to the graph-coloring that asks for the minimum number of colors. At most we have to try |V| colors. With K-=|V| you can color any graph G=(V,E) by assigning each vertex a different color.

If we can solve the graph-coloring problem for a given graph G=(V,E) in polynomial time then we can solve K-COLOR in polynomial time. Solve the graph-coloring problem and let K* be the minimum colors needed. The answer to K-COLOR is no if K < K* or K >|V|. The answer to K-COLOR is yes if K ≥ K*.

**b.** **( total of 4 points)** Use the fact that 3-COLOR is NP-complete to show that 4-COLOR is NP-complete

Step 1: **( 1 points) Show that 4-COLOR is in NP. Give a polynomial time algorithm to verify solution. This is the same as in part c) above but with k=4. Must Show polynomial to receive full credit**

> Given a Graph G=(V,E) and a 4-coloring function c: V -> {1, 2, 3, 4} we can verify if c is a "legal" coloring function in polynomial time. To verify the solution, for each vertex u in V we must check the colors of the adjacent vertices. All colors of adjacent vertices must be different. If for any (u, w) ∈ E, c(u) = c(w) then c is not a 4-COLORING of G.. The verification of the 4-coloring is polynomial in n (the number of vertices) since 4 ≤ n and the time required to look at all edges in G is O($n^2$).

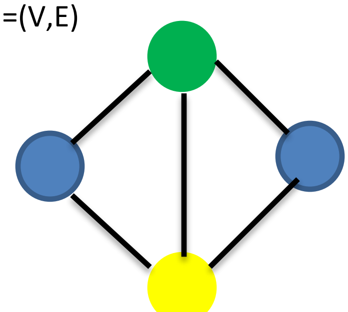
**Step 2: ( 3 points) Show that there is a polynomial reduction from 3-COLOR to 4-COLOR.**

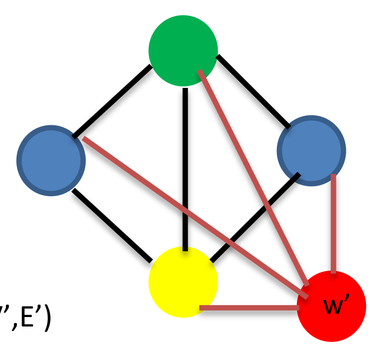
Reduce an instance G of 3-COLOR to an instance G' of 4-COLOR in polynomial time, and show that there is a 3-COLOR in G iff there is a 4-COLOR in G'. Let G=(V,E) be an instance of 3-COLOR transform G into G' by adding a new vertex w' that is connect t every other vertex. That is

G'=(V', E') where V'= V ∪{w'} and E' = E ∪ { (w', u) for all u ∈ G}

This reduction can be done in polynomial time since we are adding one vertex and at most n edges

G=(V,E)






G'=(V',E')

**blue = 1, yellow = 2, green = 3, red = 4.**

If G has a 3-COLORing then G' has a 4-COLORing. Assume G has a 3-COLORing then there exists a function c: V -> {1, 2, 3} such that for all u, w ∈ V if (u,w) ∈E then c(u) ≠c(w). Now define the 4-coloring function c' for G'

$$c'(u) = \begin{cases} c(u), & if\ u\ \in V \\ 4, & if\ u\ \notin\ V\ \ (u = w') \end{cases}$$

Therefore, if there is a 3-COLORing in G then there is a 4-COLORing in G'

If G' has a 4-COLORing then G has a 3-COLORing. Assume G' has a 4-COLORing, since w' is adjacent to all other vertices in G' then w' must be a different color. Let c' be the coloring function for G', without loss of generality we can say that c'(w')= 4 and c(u) $\neq$ 4 for all u $\in$(V' –{w}). However, (V' –{w}) = (V $\cup${w'} –{w}) = V. So we have colored all of the original vertices in V using only colors 1, 2 and 3 proving that G is 3-COLORable. Thus, the 4-Color problem is NP-Hard

**Since it was shown in Part 1 that 4-COLOR is in NP, and by Step 2 NP-Hard, 4-COLOR is NP-Complete. This statement must be included.**