ECE 406, Winter 2021, Assignment 4

Marks: every problem is worth the same, with an equal split across sub-parts. This assignment is worth 50/4 towards your marks in the course.

In this assignment, we use the notation "$A \leq B$" for two decision problems A and B to indicate that A reduces to B. Where "reduces" is defined under Definition 1 of Lecture 9(c). Also, we refer to $\mathcal{C}$-hard and $\mathcal{C}$-complete for a complexity class $\mathcal{C}$ as defined under Definitions 2 and 3 in Lecture 9(c).

Scope of this assignment: Upto and including Week 10.

1. Suppose $\{0,1\}^*$ is the set of all bit-strings. Let the function $f\colon \{0,1\}^* \to \{\mathsf{true}, \mathsf{false}\}$ be the function: $f(x) = \mathsf{true}$ for every $x \in \{0,1\}^*$. That is, given input any bit-string, f maps it to the constant true. Observe that f is a decision problem, and that an instance of f is a bit-string.

   Prove that f is <u>not</u> **NP**-hard under $\leq$.

2. Prove that the following decision problem is **P**-complete under $\leq$.

   Given as input (i) a non-empty undirected graph $G = \langle V, E\rangle$, and, (ii) a positive integer k, does G have exactly k connected components?

   (Re-emphasis: the problem refers to **P**-complete, <u>not</u> **NP**-complete.)

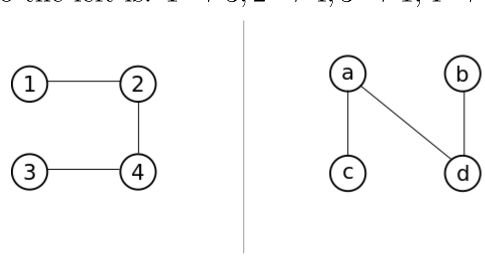
3. Let k-CIRCUIT-SAT be the decision problem: given as input a boolean circuit which allows AND, OR and NOT gates, where the AND and OR gates can take any number ≥ 2 input wires, is that circuit satisfiable?

   Let 2-CIRCUIT-SAT be the decision problem: given as input a boolean circuit which allows AND, OR and NOT gates, where the AND and OR gates can take exactly two input wires only, is that circuit satisfiable?

   Prove that k-CIRCUIT-SAT $\leq$ 2-CIRCUIT-SAT.

4. Given two undirected graphs $G_1 = \langle V_1, E_1\rangle, G_2 = \langle V_2, E_2\rangle$, we say that they are *isomorphic* to one another if: there exists an invertible function $f\colon V_1 \to V_2$ such that $\langle u, v\rangle \in E_1 \iff \langle f(u), f(v)\rangle \in E_2$. That is, if we rename every vertex u in G_1 as $f(u)$, then we get exactly the graph G_2. We say that an undirected graph G is *automorphic* if it is isomorphic to itself under a function that is not the identity mapping between the vertices.

   For example, the two graphs below are isomorphic to one another. An isomorphic mapping is $1 \mapsto c, 2 \mapsto a, 3 \mapsto b, 4 \mapsto d$. Also, each is automorphic. An automorphic mapping of the graph to the left is: $1 \mapsto 3, 2 \mapsto 4, 3 \mapsto 1, 4 \mapsto 2$.



Graph isomorphism, ISO, is the decision problem: given two undirected graphs, are they isomorphic to one another? Graph automorphism, AUTO, is the decision problem: given an undirected graph, is it automorphic?

Prove that if ISO $\in$ **P**, then AUTO $\in$ **P**.

Hint: suppose you have a polynomial-time algorithm, I, for ISO. You need to devise a polynomial-time algorithm A for AUTO. Towards this, given an undirected graph $G = \langle V_G, E_G \rangle$, suppose you make a copy $H = \langle V_H, E_H \rangle$ of G, but ensure that $V_G \cap V_H = \emptyset$, i.e., change the names of all the vertices. Then, presumably, what you'd like to do is to invoke $I(G, H)$. But the problem is that that is guaranteed to return `true` *with your renaming function between vertices of G and H as the underlying isomorphic mapping. So what you should do is create such a copy H, and then, somehow alter both G and H so that $I(G, H)$ correctly returns* `true` *or* `false`*. Note that we are allowed to invoke I multiple times within A, so long as A remains polynomial-time.*

Aside (i.e., not needed to solve the problem): this is an example of a seemingly weaker notion of a reduction than Definition 1, Lecture 9(c).

5. Recall that given an undirected graph $G = \langle V, E \rangle$, a *vertex cover* for G is a set $C \subseteq V$ with the property that $\langle u, v \rangle \in E$ implies either $u \in C$, or $v \in C$, or both.

   Consider the following decision problem, which we can call VERTEX-COVER-BOUNDS. Given as input (i) a non-empty undirected graph G, and, (ii) two non-negative integers $k_1, k_2 \in \mathbb{Z}_0^+$, does there exist a vertex cover of G of size $\leq k_1$, and for every vertex cover of G, is it of size $\geq k_2$?

   (a) Show that VERTEX-COVER-BOUNDS is **NP**-hard. You are allowed to reduce from any problem that is **NP**-hard that we have discussed in the course.

   (b) Do you think VERTEX-COVER-BOUNDS is in **NP**? Why or why not? (I am not asking for a proof. Just some arguments for whichever of 'why' or 'why not' that you pick.)