CS41 Homework 9

This homework is due at 11:59PM on Sunday, November 19. Write your solution using LaTeX. Submit this homework using **github** as **.tex** file; the code should be in a file called **pretty-print.py**. This is a **partnered homework**. You should primarily be discussing problems with your homework partner.

It's ok to discuss approaches at a high level with others. However, you should not reveal specific details of a solution, nor should you show your written solution to anyone else. The only exception to this rule is work you've done with a lab teammate *while in lab*. In this case, note (in your **homework submission poll**) who you've worked with and what parts were solved during lab.

1. **Optimization vs Decision Problems**.
   Recall that a decision problem requires a YES/NO answer, and an optimization problem requires the "best possible answer", which often means maximizing or minimizing over some *cost* or *score*.

   For most optimization problems, there is an obvious analogue as a decision problem. For example, consider the following problem:

   VC-OPT: Given a graph $G = (V, E)$,
   return the size of the smallest vertex cover in G.

   The problem VC-OPT has a natural decision problem, namely VERTEX-COVER.

   VERTEX-COVER: Given a graph $G = (V, E)$ and an integer k,
   is there a vertex cover of G of size at most k?

   In fact, every optimization problem can be converted to a decision problem in this way.

   (a) Show that VERTEX-COVER $\leqslant_{\mathrm{P}}$ VC-OPT.

   (b) Let B be an arbitrary optimization problem, and let A be the decision version of B. Show that

   $$A \leqslant_{\mathrm{P}} B \ .$$

   In order to show that $A \leqslant_{\mathrm{P}} B$, you will need to:

   - Describe an algorithm for A that uses a black box for B as a subroutine.
   - Argue that your algorithm only does polynomially much work, only calls the box for B polynomially many times.
   - Argue that your algorithm is a correct algorithm that solves problem A.

   (c) Show that VC-OPT $\leqslant_{\mathrm{P}}$ VERTEX-COVER.

2. In this problem, you will prove that THREE-COLORING is NP-COMPLETE. You have already worked on several pieces of this problem in lab, so you should definitely use that work and *not* start from scratch.

   THREE-COLORING: Given $G = (V, E)$, return YES iff the vertices in G can be colored, using at most three colors, such that each edge $(u, v) \in E$ is *bichromatic*.

(a) Prove that THREE-COLORING $\in$ NP.

(b) Given an input x for 3-SAT, create an input for THREE-COLORING using the gadgets below (Figures 1 through 5). For each clause in x, you should create a piece of the graph G which will be an input for THREE-COLORING.

Describe how to do this, and what the final graph G consists of. How is the satisfiability of the clause related to the colorability of the piece of the graph?

Recall from lab that our gadgets are three-colorable graphs which include at least three vertices marked a, b, c. Except for the specified property, the remaining vertices are *unconstrained*. For example, unless the problem states that, e.g., a cannot be red, it must be possible to color the graph in such a way that a is red. Colors for other vertices may be fixed, just not a, b, c.

Figure 1: A graph such that a, b, c all have different colors.

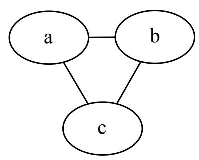


Figure 2: A graph such that a, b, c all have the same color.

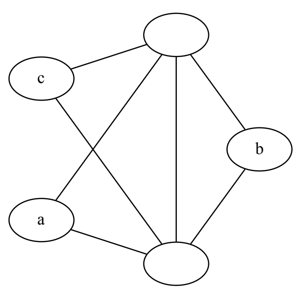


Figure 3: A a graph such that a, b, c do *NOT* all have the same color.

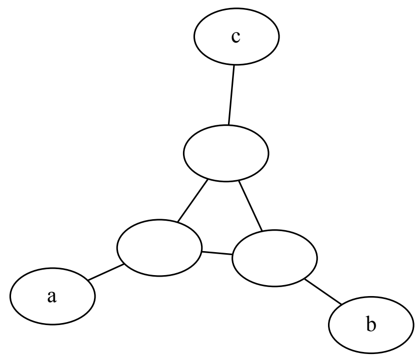


Figure 4: A graph such that none of a, b, c can be green.

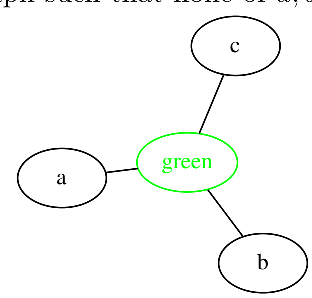
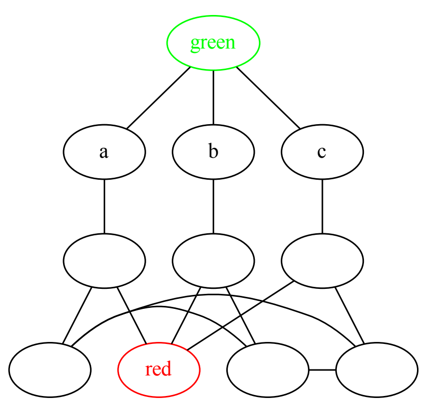


Figure 5: A graph such that none of a, b, c are green, and they cannot *all* be blue.



(c) Run the THREE-COLORING algorithm on the input G you create, and output YES iff the THREE-COLORING algorithm outputs YES. Argue why this procedure gives you a correct answer for 3-SAT. (Hint: Associate the color red with TRUE and the color blue with FALSE.)

3. **Other coloring problems.** It is natural to wonder whether there is something special about the 3 in THREE-COLORING that makes it such a hard problem.

   (a) In the TWO-COLORING problem, the input is a graph $G = (V, E)$, and you should output YES iff the vertices in G can be colored using at most two colors such that each edge $\{u, v\} \in E$ is *bichromatic*. Prove that TWO-COLORING $\in$ P.
   (Hint: look at your notes from earlier in the semester.)

   (b) In the FOUR-COLORING problem, the input is a graph $G = (V, E)$, and you should output YES iff the vertices in G can be colored using at most four colors such that each edge $\{u, v\} \in E$ is *bichromatic*. Prove that FOUR-COLORING $\in$ NP-COMPLETE.
   (Hint: for your reduction, you can pick any NP-COMPLETE problem, but some will make your life easier. Try to do a reduction from a *very* similar problem.)