CS46 practice problems 13

These practice problems are an opportunity for discussion and trying many different solutions. It is not counted towards your grade, and you do not have to submit your solutions. The purpose of these problems is to get more comfortable with reasoning and writing about $P$, $NP$, and polynomial-time reductions.

If you are stumped or looking for guidance, ask.

1. A coloring of a graph is an assignment of colors to its nodes so that no two adjacent nodes are assigned the same color.

Define Three-Coloring as:

$$\text{Three-Coloring} = \{\langle G \rangle \mid G \text{ is colorable with three colors}\}$$

Show that Three-Coloring is NP-complete. (Hint: reduce from 3-Sat and use the subgraphs given in the textbook hint, page 325. Overall, your construction should preserve the property that: there exists a valid 3-coloring if and only if there is a satisfying assignment.)

2. One way to come up with new NP-complete problems is to generalize from a problem we already know is NP-complete. Then, if certain parameters of the problem are fixed in a certain way, the problem becomes a known NP-complete problem. One can reduce any problem to its generalization by simply introducing a new parameter, and otherwise leaving the instance as it is.

Prove that the following language is NP-complete by showing that it is the generalization of an NP-complete problem. Give the appropriate parameter restriction.

**LONGESTCYCLE**: Given a graph $G$ and integer $k$, is there a cycle, with no repeated nodes, of length at least $k$?

3. Define the language **SELF-ESTEEM$_{TM}$** to be the set of Turing machines that accept themselves.

$$\text{SELF-ESTEEM}_{TM} = \{\langle M \rangle \mid M \text{ is a Turing machine which accepts string } \langle M \rangle\}$$

Is SELF-ESTEEM$_{TM}$ decidable? recognizable? co-recognizable?

4. Show that if $P = NP$, a polynomial-time algorithm exists that produces a satisfying assignment when given a satisfiable Boolean formula.

Note: The algorithm you are being asked to write computes a function, but NP contains languages, not functions. The $P = NP$ assumption means that Satisfiability $\in P$, so there is a deterministic polynomial-time Turing machine $M_{\text{SAT}}$ which can test if a formula is satisfiable. You don’t know how this test is done, but you may use $M_{\text{SAT}}$ in your algorithm.