COSC 240, Spring 2020: Problem Set #5

Assigned: Thur, 4/16.
Due: Tue 4/28, at the beginning of class (submit electronically in Zoom).
Lectures Covered: 24 to 26
Academic Integrity: You can work with other people in the class but you must write up your own answers in your own words. You can also use the textbook and talk to the professor. You may not use any other resources (e.g., material found online) or talk to people outside the class about these problems. See the syllabus for details on the academic integrity policy for problem sets.

Problems

1. In class we proved that there does not exist an algorithm that *decides* the following language:

$$A_{TM} = \{\langle A, w\rangle \mid A \text{ is an algorithm that accepts input } w\}.$$

   As discussed, however, there does exist an algorithm that *accepts* A_{TM} (i.e., the algorithm that on input $\langle A, w\rangle$ simulates A on w and accepts $\langle A, w\rangle$ if A accepts w in the simulation).

   This problem ask you to consider the language $\overline{A_{TM}}$, which contains every string that is *not* in A_{TM}. Prove that there does not exist an algorithm that *accepts* $\overline{A_{TM}}$.

   (*Hints:* Your proof should leverage the two facts reviewed in the problem statement; i.e., that A_{TM} cannot be decided but can be accepted. Your proof should be a contradiction proof; e.g., assume for contradiction that some algorithm H accepts $\overline{A_{TM}}$, and then use this assumption to create a new algorithm that uses H as a subroutine to solve something we previously proved impossible.)

2. Prove that each of the following three languages are in NP:

   (a) $COMPOSITES = \{\langle x\rangle \mid x = pq, \text{ for integers } p, q > 1\}$.
   (b) $NEQ = \{\langle \phi, \phi'\rangle \mid \phi$ and ϕ' are boolean formulas that are *not* equivalent$\}$.
   (c) $BADPAIRS = \{\langle S, k\rangle \mid S$ is a set containing natural numbers, and for every $x, y \in S, xy \neq k\}$.

3. Prove that the language $5SAT = \{\langle \phi\rangle \mid \phi$ is a boolean formula in 5-cnf with a satisfying assignment$\}$ is NP-complete.

   (*Note:* 5-cnf is defined the same as 3-cnf except there are 5 literals in each or clause instead of 3.)