Math/CCS 103	Professor: Padraic Bartlett

Homework 5: NP-Completeness

Due Friday, week 3 *UCSB 2014*

Homework Problems.

Pick **two** of the problems below, and solve them!

1. We defined the **traveling salesman** problem in class Wednesday. As a reminder: an **instance** of the Traveling Salesman problem consists of the following:

   - A list $\{C_1, \dots C_N\}$ of cities.
   - A distance function $d(C_i, C_j)$ that outputs distances between these cities.
   - A maximum distance D.

   Given such an instance, the **problem** itself is the following: can you create a "tour" of these cities with distance no greater than D? In other words, can you create a path that visits each city once, starting and ending at the same city, that has total length no greater than D?

   Show that the traveling salesman problem is in NP.

2. On Wednesday's HW, I asked you to show that the **Hamiltonian cycle problem** was in NP. **Reduce** the Hamiltonian cycle problem to the traveling salesman problem: i.e. assuming that you have an algorithm A that solves the traveling salesman problem, explain how you can modify A to get an algorithm that solves the Hamiltonian cycle problem. (In this sense, you are proving that solving the traveling salesman problem is at least as hard as solving the Hamiltonian cycle problem.)

3. A boolean formula is said to be in **2-conjunctive normal form** if we can write it in conjunctive normal form, where each disjunction (or) contains precisely two literals. For example, the following formula is in 3-conjunctive normal form:

   $$(x \vee y) \wedge (\neg x \vee x) \wedge (a \vee a).$$

   2SAT is the following problem: given any boolean formula written in 2-conjunctive normal form, is it satisfiable?

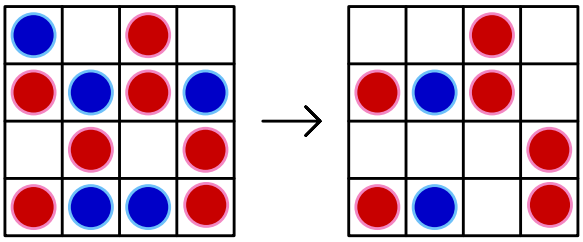
   Prove that 2SAT is in P. (Yes, P. Not NP. 2 is a weird number.)

4. (From Erickson's notes on NP-hardness.) Consider the following solitare game, which is played on a $n \times n$ board:

   - To start, we place red stones on some of the squares of our board, and blue stones on other squares of our board. We do not have to fill every square of our board; some places may be left blank.

- The goal of our game is to remove stones one by one until we satisfy both of the following conditions:
  - Every **row** contains at least one stone.
  - Conversely, no **column** contains stones of both colors in it.

Here is an example board, along with a winning state:



(a) Find a board that does not have a winning state.

(b) Show that determining whether a given board has a solution is a problem in NP.

(c) Suppose that you only allow boards where every square has a stone on it. Is it possible for a board to not have a winning state? Or starting from any completely-filled board, can you always win this solitare game?