CS 473: Algorithms, Fall 2016
HW 10 (due Wednesday, November 30 at 8pm)

This homework contains three problems. **Read the instructions for submitting homework on the course webpage**.

Collaboration Policy: For this home work, each student can work in a group with up to three members. Only one solution for each group needs to be submitted. Follow the submission instructions carefully.

For problems that ask to prove that a given problem X is NP-hard, a full-credit solution requires the following components:

- Specify a known NP-hard problem Y, taken from the problems listed in the notes.
- Describe a polynomial-time algorithm for Y, using a black-box polynomial-time algorithm for X as a subroutine. Most NP-hardness reductions have the following form: Given an arbitrary instance of Y, describe how to transform it into an instance of X, pass this instance to a black-box algorithm for X, and finally, describe how to transform the output of the black-box subroutine to the final output. A cartoon with boxes may be helpful.
- Prove that your reduction is correct. As usual, correctness proofs for NP-hardness reductions usually have two components (one for each f).

---

1. Recall the facility location problem from HW 9. Prove that the decision version of it is NP-Complete.

2. k-Color is the problem of deciding whether a given graph $G = (V, E)$ can be colored with k colors.

   - **Not to submit:** Describe a polynomial-time reduction from 3-Color to 10-Color.
   - Describe a polynomial-time reduction from 10-Color to SAT. Why does this imply a polynomial-time reduction from 10-Color to 3-Color?

3. Directed Hamiltonial-Path is the problem of deciding whether a given directed graph $G = (V, E)$ has a path that visits each vertex. Suppose you had black-box algorithm for Directed Hamiltonian-Path (note that this algorithm only answers YES or NO). Using the black box algorithm describe a polynomial-time algorithm that given a directed graph $G = (V, E)$ outputs a Hamiltonian Cycle in G if it has one, or returns NO if it does not have any. Note that you are allowed to use the algorithm for Hamiltonian-Path more than once.

The remaining problems are for self study. Do *NOT* submit for grading.

- Reduce 3-SAT to 5-SAT. How does this generalize when you want to reduce 3-SAT to k-SAT where k is some fixed constant? This is a useful exercise to understand the reduction from SAT to 3-SAT.

- We briefly discussed in class how to reduce Dominating Set to Set Cover. Describe a polynomial time reduction from Set Cover to Dominating Set.

- An instance of Subset Sum consists of n non-negative integers $a_1, a_2, \ldots, a_n$ and a target B. The goal is to decide if there is a subset of the n numbers whose sum is exactly B. The 2-Partition problem is the following: given n integers $a_1, a_2, \ldots, a_n$, is there a subset S such that the sum of the numbers in S is equal to $\frac{1}{2}\sum_{i=1}^{n} a_i$. It is easy to see that 2-Partition reduces to Subset Sum. Do the reverse. Reduce Subset Sum to 2-Partition.

- See HW 1 from Sariel's course in Fall 2015. `https://courses.engr.illinois.edu/cs473/fa2015/w/hw/hw_01.pdf`.

- Jeff's notes, Kleinberg-Tardos and Dasgupta etal have several nice problems on NP-Complete reductions. Skim through several of them to quickly identify which problem you would use for the reduction.

- See Jeff's home work 10 from Spring 2016. last spring. `https://courses.engr.illinois.edu/cs473/sp2016/hw/hw10.pdf`