CS 473 ✧ Spring 2017

Homework 10

Due Wednesday, April 26, 2017 at 8pm

---

1. An ***integer linear program*** is a linear program with the additional explicit constraint that the variables must take *only* integer values. The ILP-Feasibility problem asks whether there is an integer vector that satisfies a given system of linear inequalities—or more concisely, whether a given integer linear program is feasible.

   Describe a polynomial-time reduction from 3Sat to ILP-Feasibility. Your reduction implies that ILP-Feasibility is NP-hard.

2. There are two different versions of the Hamiltonian cycle problem, one for directed graphs and one for undirected graphs. We saw a proof in class (and there are two proofs in the notes) that the *directed* Hamiltonian cycle problem is NP-hard.

   (a) Describe a polynomial-time reduction from the *undirected* Hamiltonian cycle problem to the *directed* Hamiltonian cycle problem. Prove your reduction is correct.

   (b) Describe a polynomial-time reduction from the *directed* Hamiltonian cycle problem to the *undirected* Hamiltonian cycle problem. Prove your reduction is correct.

   (c) Which of these two reductions implies that the *undirected* Hamiltonian cycle problem is NP-hard?

3. Recall that a 3CNF formula is a conjunction (And) of several distinct clauses, each of which is a disjunction (Or) of exactly three distinct literals, where each literal is either a variable or its negation.

   Suppose you are given a magic black box that can determine **in polynomial time**, whether an arbitrary given 3CNF formula is satisfiable. Describe and analyze a **polynomial-time** algorithm that either computes a satisfying assignment for a given 3CNF formula or correctly reports that no such assignment exists, using the magic black box as a subroutine. *[Hint: Call the magic black box more than once. First imagine an even more magical black box that can decide Sat for arbitrary boolean formulas, not just 3CNF formulas.]*

Rubric (for all polynomial-time reductions): 10 points =

+ 3 points for the reduction itself
  – For an NP-hardness proof, the reduction must be from a known NP-hard problem. You can use any of the NP-hard problems listed in the lecture notes (except the one you are trying to prove NP-hard, of course).
+ 3 points for the "if" proof of correctness
+ 3 points for the "only if" proof of correctness
+ 1 point for writing "polynomial time"

- An incorrect polynomial-time reduction that still satisfies half of the correctness proof is worth at most 4/10.
- A reduction in the wrong direction is worth 0/10.