

CS 473: Algorithms, Spring 2021

HW 10 (due Wednesday, April 28 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 for a (integer) linear-programming formulation of some problem, a full credit solution requires the following components:

- A list of variables, along with a brief English description of each variable. (Omitting these English descriptions is a Deadly Sin.)
- A linear objective function (expressed either as minimization or maximization, whichever is more convenient), along with a brief English description of its meaning.
- A sequence of linear inequalities (expressed using \leq , $=$, or \geq , whichever is more appropriate or convenient), along with a brief English description of each constraint.
- A proof that your linear programming formulation is correct, meaning that the optimal solution to the original problem can always be obtained from the optimal solution to the linear program. This may be very short.

It is *not* necessary to express the linear program in canonical form, or even in matrix form. Clarity is much more important than formality. 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. *Valid Coloring* of an undirected graph $G = (V, E)$ is an assignment of colors to vertices such that no two vertices connected by an edge have the same color; it can also be thought of as a function $Color : V \rightarrow \{1, \dots, k\}$, where k is the number of colors used, such that $Color(u) \neq Color(v)$ if $(u, v) \in E$. Chromatic number $\chi(G)$ of a graph G is the minimum number of colors needed to obtain a valid coloring of G .

Design an integer linear program to compute $\chi(G)$ for a given graph G .

2. Give polynomial-time reductions for the following.

- (a) (5 pts) Give a reduction from the *Dominating Set* problem to the *Set cover* problem.

An instance of *Dominating Set* consists of a graph $G = (V, E)$ and a target d . The goal is to decide if there is a subset $D \subseteq V$ of $d = |D|$ vertices such that every vertex in G is either in D , or has a neighbour in D .

An instance of *Set Cover* consists of a target k , and a collection of subsets $S = \{S_1, \dots, S_m\}$ of a universe U . In other words, $S_i \subseteq U$ for all $1 \leq i \leq m$, and it is assumed that $\bigcup_{i=1}^m S_i = U$. The goal is to decide if there is a subset of S of size k that covers U , i.e. a collection of indices I of size k such that $\bigcup_{i \in I} S_i = U$.

- (b) (5 pts) Give a reduction from the *Subset Sum* problem to the *2-Partition* problem.

An instance of *Subset Sum* consists of n non-negative integers a_1, a_2, \dots, a_n and a target B . The goal is to decide if some subset of the n numbers sums exactly to B .

An instance of *2-Partition* consists of n non-negative integers a_1, a_2, \dots, a_n , and the goal is to decide if there is a subset of the n numbers whose sum 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*.

3. Prove that the following problems are NP-complete:

- (a) Given a directed graph $G = (V, E)$ that is connected, check if it has a *simple path* of length at least $\lceil \frac{|V|}{2} \rceil$. Simple path is a path on which no vertex repeats.
- (b) The *hitting set* problem: Given a set U of n elements, a collection A_1, \dots, A_m of subsets of U , and an integer k , check if there exists a subset $X \subset U$ such that $|X| \leq k$, and for each i from 1 to m , $A_i \cap X \neq \emptyset$.

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.
- Reduce 3-COLOR to 11-COLOR, and 11-COLOR to 3-SAT. The latter gives a reduction from 11-COLOR to 3-COLOR.
- Describe a polynomial time reduction from Set Cover to Dominating Set.
- *Valid coloring* of an undirected graph $G = (V, E)$ is an assignment of colors to vertices such that no two vertices connected by an edge have the same color.

Suppose you are given access to an oracle \mathcal{X} the k -COLOR problem: given as input an undirected graph G and an integer $k > 0$, returns 1 if G has a valid coloring with at most k colors, and 0 otherwise. Assume $O(1)$ -time access to the oracle.

1. Chromatic number of a graph G is the minimum number of colors needed to obtain a valid coloring of G . Design an efficient algorithm to find the chromatic number $\chi(G)$ of G .
 2. Find an actual minimum coloring of the graph. That is, describe an efficient algorithm that computes a function $\phi : V \rightarrow \{1, 2, \dots, \chi(G)\}$ such that coloring each vertex $u \in V$ with color $\phi(u)$ produces a valid coloring of G .
- 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 et al 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>