# CS 570 - HW 6

Due Thursday, April 29, 2021 (by 4 AM Pacific Time)

## 1 Integer Programming (20 Points)

In Linear Programming, variables are allowed to be real numbers. Consider that you are restricting variables to be only integers, keeping everything else the same. This is called Integer Programming. Integer Programming is nothing but a Linear Programming with the added constraint that variables be integers. Prove that integer programming is NP-Hard by reduction from SAT.

## 2 HALF-SAT (20 points)

We know that the SAT problem is NP-complete. Consider another variant of the SAT problem: given a CNF formula $F$, does there exist a satisfying assignment in which exactly half the variables are true? Let us call this problem HALF-SAT. Prove that HALF-SAT is NP-complete.

## 3 Taking courses (20 points)

There is a set of courses, each of them is represented by a set of disjoint time intervals with the starting and finishing times. For example, a course could require the time from 9am to 11am and 2pm to 3pm and 4pm to 5pm. You want to know, given a number $K$, if it's possible to take at least $K$ courses. You cannot choose any two overlapping courses. Prove that the problem is NP-complete, which means that choosing courses is indeed a difficult thing in our life. Use a reduction from the Independent set problem.

## 4 Approximation 1 (20 points)

It is well-known that planar graphs are 4-colorable. However finding a vertex cover on planar graphs is NP-hard. Design an approximation algorithm to solve the vertex cover problem on planar graph. Prove your algorithm approximation ratio.

## 5 Approximation 2 (20 points)

Consider the following heuristic to compute a minimum vertex cover of a connected graph $G$. Pick an arbitrary vertex as the root and perform depth first search. Output the set of non-leaf vertices in the resulting depth first search tree.

1. Show that the output is indeed a vertex cover for $G$.
2. How good is this approximation? That is, upper bound the ratio of the number of vertices in the output to the number of vertices in a minimum vertex