Reductions: Written Problems

UPDATE (11/17/22): This homework assignment will be reduced dramatically. You only have to work on the first problem shown below. Please do your best to answer the problem and submit your pdf to Gradescope. In addition, this homework is graded on EFFORT ONLY. If you submit a reasonable answer and we can tell you made a serious attempt at the problem, you will pass this homework.

1. Consider the *Lecture Planning* problem (described below). For this question, you will prove that the *Lecture Planning* problem is NP-Complete.

   Suppose for a college course, that there are n total (unique) guest lecturers that can visit the class to present. One guest lecture will be given for each of the first l weeks of the course. For each of these weeks, you are given a subset of the n guest lecturers who are available that week (notice that most of these lecturers will have multiple free weeks).

   During the second half of the course, there are p projects that must be completed. For each project, there is a list of guest lectures that will allow the students to be able to complete that project successfully (These lists might overlap, i.e., one guest lecture might allow multiple projects to be completed). The problem, thus, is to choose a schedule of l guest lecturers, one per week, that will ensure the students can successfully complete each project.

   First, prove that this problem is in NP by describing a verification algorithm. Then, prove that lecture-planning is in *NP-HARD* by showing that 3-SAT $\leq_p$ LP where LP is the lecture-planning problem. Prove that the outputs of the two problems is always the same (notice this is a bi-conditional).

The following problems are NO LONGER REQUIRED for this homework assignment. I have left them here for your convenience in case you would like to use them as practice. They are fun and interesting challenges, but are not required to pass this homework assignment.

2. In class, we saw that the problem of satisfying a boolean formula in 3-conjunctive normal form (i.e., the *3-SAT* problem) is *NP-Complete*. Consider the problem of satisfying a boolean formula in 3-disjunctive normal form (e.g., $(x \wedge \neg y \wedge y) \vee (\neg x \wedge y \wedge z)$...). Is this problem also NP-Complete? If so, prove it. If not, find an algorithm that efficiently solves it and describe it.

3. Suppose you are designing a *flow network* to model a situation in which you assign work to different employees to assign tasks for a large project. You want to use *Ford-Fulkerson* to efficiently match workers to tasks using an algorithm. However, a problem emerges. You are modeling *flow* as time (minutes spent working). The max number of minutes working is sent into nodes representing the workers, and then these minutes are distributed to tasks. You realize though, that it is required that an edge from a worker to a task always be at

maximum capacity (because a worker always needs to fully work the given task and work all the minutes.

You realize your problem is a little different from normal *flow networks*. You can describe it as such: Given a flow network, find the maximum flow such that every edge is either at exactly 0 flow, or exactly at max capacity flow.

It turns out this problem is *NP-Complete*. Prove a reduction from the *3-SAT* problem to this new instance of max-flow (*FYI, I was actually working on this exact problem a few years ago before realizing it was NP-Complete, forcing me to start over and re-work my approach.*).

4. For this problem, we will prove that the *knapsack problem* from class is NP-Complete. First, consider the *ethical knapsack problem* (which is NP-Complete) and states the following: Given an array of the value of items a that can be stolen, a thief wants to steal a combination of items such that the total value is exactly some sum s (the thief is "ethical" in the sense that they only want to steal enough to pay off their debts, but no more. Also, the thief has no restriction on how much they can carry). Any algorithm would return yes or no: whether or not it is possible to steal the exact value s worth of items.

   Also, recall the *knapsack problem* from class in which a thief has a knapsack capacity C and the items have corresponding values and weights. Show the following:

   - formulate the decision problem for the *knapsack problem* (from class).
   - show that the decision version of the *knapsack problem* is in NP by providing a verification algorithm.
   - show that the decision version of the *knapsack problem* is NP-Hard by producing a reduction from the *ethical knapsack problem* (*HINT: Don't overthink this one. The reduction is quite simple.*).