CS41 Homework 10

This homework is due at 5:00PM on Friday, November 20. Write your solution using LaTeX. Submit this homework using **github** as a file called **hw9.tex**. This is a **partnered homework**. It's ok to discuss approaches at a high level with others; however, you should **primarily discuss approaches with your homework partner.**

If you do discuss problems with others, you should not reveal specific details of a solution, nor should you show your written solution to anyone else. The only exception to this rule is work you've done with a lab partner *while in lab.* In this case, note (in your **homework submission poll**) who you've worked with and what parts were solved during lab.

1. **Test Your Intuition.** Consider the following claim:

   Claim 1. *Let G be an arbitrary flow netwrok, with a source s, sink t, and a positive integer capacity c_e on every edge e. Let (A, B) be a minimum $s-t$ cut with respect to these edge capacities $\{c_e : e \in E\}$. Now, suppose we add 1 to every edge capacity. Then, (A, B) is still a minimum $s-t$ cut with respect to these new capacities $\{1 + c_e : e \in E\}$.*

   Answer whether you think the claim is TRUE or FALSE. If you answer TRUE, give a short explanation why it is true. If you answer FALSE, give a counterexample showing the claim is false.

2. **Evacuation Congestion.** (K& T 7.14) When handling natural disasters like hurricanes or wildfires, state and federal agencies need to plan how to evacuate potentially thousands of people. One major challenge is that evacuees might take the same routes to evacuation points. The resulting road congestion can hinder evacuation. In this problem, you will design an algorithm to help agencies minimze this congestion. The input to this problem is as follows;

   - a directed graph $G = (V, E)$.
   - a collection of populated vertices $A \subset V$.
   - a collection of safe vertices $B \subset V$.

   Assume A and B are disjoint. In case of an emergency, we want evacuation routes from the populated vertices to the safe vertices. A set of evacuation routes is defined as a set of paths in G so that (i) each node in A is the tail of one path, (ii) the last node in each path is in B, and (iii) the paths do not share any edges. Such a set of paths gives a way for the occupants of the populated vertices to *escape* to B without overly congesting any edge in G.

   (a) Given G, A, and B, show how to decide in polynomial time whether such a set of evacuation routes exists.

   (b) Suppose we have the same problem as before, but now condition (iii) is that paths do not share any *vertices*. Show how to decide in polynomial time whether such a set of evacuation routes exists.

   (c) Provide an example input G, A, B such that the answer is yes to the question in (a) but no to the question in (b).

3. **Advertising contracts** (K& T 7.16)

   Back in the euphoric early days of the Web, people liked to claim that much of the enormous potential in a company like Yahoo! was in the "eyeballs"—the simple fact that millions of people look at its pages every day. Further, by convincing people to register personal data with the site, a site like Yahoo! can show each user an extremely targeted advertisement whenever the user visits the site, in a way that TV networks or magazines couldn't hope to match. So if a user has told Yahoo! that she is

a 21-year-old computer science major at Swarthmore, the site can present a banner ad for apartments in Philadelphia suburbs; on the other hand, if she is a 50-year-old investment banker from Greenwich, CT, the site can display a banner ad pitching luxury cars instead.

But deciding on which ads to show to which people involves some serious computation behind the scenes. Suppose that the managers of a popular site have identified k distinct *demographic groups* $G_1, G_2, \ldots, G_k$. (Some may overlap.) The site has contracts with m different advertisers to show a certain number of copies of their ads to users of the site. Here's what a contract with the i^{th} advertiser looks like:

- For a subset $X_i \subseteq \{G_1, \ldots, G_k\}$ of the demographic groups, advertiser i wants ads shown only to users who belong to at least one of the groups listed in X_i.
- For a number r_i, advertiser i want its ads shown to at least r_i users each minute.

Consider the problem of designing a good *advertising policy* — a way to show a single ad to each user of the site. (Imagine a world where each user saw only *one* ad per site.) Suppose at a given minute, there are n users visiting the site. Because we have registration about each of the users, we know that user j belongs to a subset U_j of the demographic groups.

The problem is: is there a way to show a single ad to each user so that the site's contracts with each of the m advertisers is satisfied for this minute?

Give an efficient algorithm to decide if this is possible, and if so, to actually choose an ad to show each user.

4. Multiple-Interval-Scheduling (K&T 8.14) In this problem, there is a machine that is available to run jobs over some period of time, say 9AM to 5PM.

   People submit jobs to run on the processor; the processor can only work on one job at any simgle point in time. However, in this problem, each job requires a **set of intervals** of time during which it needs to use the machine. Thus, for example, one job could require the processor from 10AM to 11AM and again from 2PM to 3PM. If you accept this job, it ties up your machine during these two hours, but you could still accept jobs that need any other time periods (including the hours from 11AM to 2PM).

   Now, you're given an integer k and a set of n jobs, each specified by a set of time intervals, and you want to answer the following question: is it possible to accept at least k of the jobs so that no two of the accepted jobs have any overlap in time?

   In this problem, you are to show that Multiple-Interval-Scheduling $\in$ NP-complete. To assist you, we've broken down this problem into smaller parts:

   (a) First, show that Multiple-Interval-Scheduling $\in$ NP.

   (b) In the remaining two parts, you will reduce

   $$\text{Independent-Set} \leqslant_{\text{P}} \text{Multiple-Interval-Scheduling}\ .$$

   Given input $(G = (V, E), k)$ for Independent-Set, create a valid input for Multiple-Interval-Scheduling. First, divide the processor time window into m distinct and disjoint *intervals* $i_1, \ldots, i_m$. Associate each interval i_j with an edge e_j. Next, create a different job J_v for each vertex $v \in V$. What set of time intervals should you pick for job J_v?

   (c) Finally, run the Multiple-Interval-Scheduling algorithm on the input you create, and output yes iff the Multiple-Interval-Scheduling algorithm outputs yes. Argue that the answer to Multiple-Interval-Scheduling gives you a correct answer to Independent-Set.

5. **(Extra Challenge Problem.)** Does P = NP? Answer yes or no. Justify your response with a formal proof.