15-351 / 15-650 / 02-613: Exercises 3a

1. Imagine a school is hosting a job fair where students have the opportunity to interview with potential employers. The school wants to match as many students to employers as possible based on the following constraints:

   (a) There are n students and m employers.

   (b) Each employer has $l_1, l_2, ..., l_m$ interview slots available

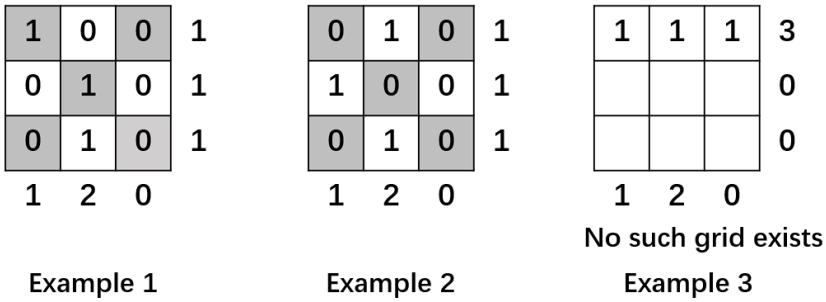
   (c) Each student has a list of employers they are interested in interviewing with, which can be represented using matrix $M_{n \times m}$, $M[i, j] = 1$ if student i is interested in company j, else $M[i, j] = 0$

   (d) Each student can only interview with each employer at most once.

   (e) Each student can have at most $e_1, ..., e_n$ interviews.

   Design a network flow-based algorithm that maximizes the total number of interviews conducted.

2. Consider the Binary Magic Square problem: Given a list of n integers $\mathbf{r} = (r_1, \ldots, r_n)$, and a list of m integers $\mathbf{c} = (c_1, \ldots, c_m)$, we ask whether there is an $n \times m$ grid of 0/1 such that row i sums to r_i and column j sums to c_j. We assume $\sum_i r_i = \sum_j c_j$.
   Example 1: n = m = 3 with $\mathbf{c} = (1, 2, 0)$, $\mathbf{r} = (1, 1, 1)$
   Example 2: n = m = 3 with $\mathbf{c} = (1, 2, 0)$, $\mathbf{r} = (1, 1, 1)$
   Example 3: n = m = 3 with $\mathbf{c} = (1, 2, 0)$, $\mathbf{r} = (3, 0, 0)$, no such grid exists.



   Use network flow to create a polynomial-time algorithm to decide whether it is possible to design a grid that obeys the given $\mathbf{r}$ and $\mathbf{c}$.

3. You're helping to organize a class on campus that has decided to give all of its students wireless laptops for the semester. Thus, there is a collection of n wireless laptops; there is also a collection of n wireless access points, to which a laptop can connect when it is in range.

   The laptops are currently scattered across campus; laptop l is within range of a set S_l of access points. We will assume that each laptop is within range of at least one access point (so the sets S_l are nonempty); we will also assume that every access point p has at least one laptop within range of it.

To make sure that all the wireless connectivity software is working correctly, you need to try having laptops make contact with access points in such a way that each laptop and each access point is involved in at least one connection. Thus we will say that a test set T is a collection of ordered pairs of the form (l, p) for a laptop l and access point p, with the properties that

(a) If $(l, p) \in T$, then l is within range of p (i.e., $p \in S_l$).

(b) Each laptop appears in at least one ordered pair in T.

(c) Each access point appears in at least one ordered pair in T.

This way, by trying out all the connections specified by the pairs in T, we can be sure that each laptop and each access point have correctly functioning software.

The problem is: Given the sets S_l for each laptop (i.e., which laptops are within range of which access points), and a number k, decide whether there is a test set of size at most k.

- Give an example of an instance of this problem for which there is no test set of size n.
- Give a polynomial-time algorithm that takes the input to an instance of this problem (including the parameter k) and decides whether there is a test set of size at most k.

4. Here's an linear program in standard form:

$$\begin{aligned} \text{maximize} \quad & 2x_1 + x_2 \\ \text{s.t.} \quad & x_1 + x_2 \leq 4 \\ & x_1 - x_2 \leq 1 \\ & x_1 \leq 3 \\ & \vec{x} \geq 0 \end{aligned}$$

(a) Write a series of tableaux of obtained by the simplex method starting from the initial tableau to an optimal one.

(b) What is the optimal $\vec{x}$ and optimal value of the objective function?

5. On graduate school visit days, professors need to meet with visiting students to explain to them how great CMU is. Visiting students have certain professors they want to meet with and professors have certain visiting students they want to impress. Over visit day, there are meetings at various fixed time slots, and some professors and students are only available at certain times. Your task is to find an assignment of students, professors, and times that allows as many meetings to take place as possible.

More formally, we have:

- A set of visiting students $S = \{s_1, \ldots, s_m\}$.

- A set of professors $P = \{p_1, \ldots, t_n\}$.
- A set of time slots $T = \{t_1, \ldots, t_a\}$.
- A collection of sets $A_{p_i} \subseteq T$, where A_{p_i} gives the time slots that professor p_i is available.
- A collection of sets $A_{s_i} \subseteq T$, where A_{s_i} gives the time slots that student s_i is available.
- A collection of sets $M_{p_i} \subseteq S$, where M_{p_i} gives the students with whom professor p_i is interested in meeting.
- A collection of sets $M_{s_i} \subseteq P$, where M_{s_i} gives the professors that visiting student s_i wants to meet.

Professor p_i will only meet with students on her list M_{p_i} and student s_i will only meet with professors on their list M_{s_i}. (We'll relax these requirements later.) A meeting can only happen at time t_i if both the student and professor are available. At any time, each student can meet with at most 1 professor and each professor can meet with at most 1 student.

Example input:

Professor Meeting Preferences:

	Steve	Tom	Uri	Wei	Xin	Yang	Zhang
Prof. Abe	x			x		x	
Prof. Bob	x	x	x				
Prof. Carl	x			x		x	
Prof. Dave	x	x	x	x	x	x	x

S; = M_{Abe}; P

Student Meeting Preferences:

	Prof. Abe	Prof. Bob	Prof. Carl	Prof. Dave
Steve			x	
Tom		x	x	
Uri	x	x		x
Wei			x	
Xin			x	
Yang			x	
Zhang			x	

= M_{Steve}

Everyone's availability:

	2:00-2:30	2:30-3:00	3:00-3:30	3:30-4:00	4:00-4:30
Prof. Abe	x			x	
Prof. Bob	x	x	x		
Prof. Carl	x			x	
Prof. Dave		x		x	x
Steve	x	x		x	x
Tom		x	x	x	x
Uri	x		x	x	x
Wei	x	x		x	x
Xin	x		x	x	x
Yang	x	x	x	x	x
Zhang	x	x	x	x	x

T; = A_{Abe}; = A_{Steve}

(a) Write an integer linear program that finds maximum the number of (prof, student, time) meetings that can happen.

(b) Explain how to get the actual meeting schedule from your integer linear program.

(c) Suppose you are now also given, for each professor, a "bigshot score" b_{p_i} that is a positive number that says how important it is to satisfy that professor's demands. For every meeting that professor p_i takes with a student in her preference list M_{p_i}, we get b_{p_i} points. Describe how to modify your integer linear program to find a schedule that maximizes the number of bigshot points your schedule gets.