CS41 Lab 1

September 10, 2018

In typical labs this semester, you'll be working on a number of problems. You are encouraged to work with two or three others. You will not be handing in solutions; the primary purpose of these labs is to have a low-pressure space to discuss algorithm design. However, it will be common to have some overlap between lab exercises and homework sets.

1. **Counterfeit Coins.** You are given n coins and a balance scale. To use this scale, you put a number of coins in a pile on the left part of the scale, and a number of coins in a pile on the right. The scale indicates which pile is heavier.

   Most of the coins are identical in every aspect; however, one of the coins is counterfeit and much heavier than the rest. Design an algorithm to identify the counterfeit coin that uses the scale at most $\log_3 n$ times.

2. **Claire's Coffee Corner.** Claire has a coffee shop, famous for its elaborate special-order coffees. Each beverage costs $2, no matter what unusual ingredients are used or how long it takes to prepare the drink. Needless to say, this deal attracts a lot of regular customers, some of whom order very unusual concoctions that take a long time to prepare.

   Claire's regular customers are impatient and demanding. They want coffees made *exactly* according to their special order, and if their drinks aren't started as soon as they enter the shop, they get angry and storm out.

   Claire used to have many employees to help her make the coffees, but unfortunately, all of Claire's coffee clerks went on strike, leaving Claire alone to make the coffees. Claire wants to make as many coffees as possible, but now can only prepare one drink at a time. Coffee preparation requires a lot of attention, so once Claire begins to make a drink, she must focus on that drink until it is ready. Only after it is prepared can Claire begin the next drink.

   Design and analyze a strategy to help Claire quickly decide which drinks to make. Your strategy should take as input the information about regular customers (when they arrive and how long their drink takes to make) and output which drinks Claire should make. Does your strategy guarantee that Claire prepares the maximum number of coffees? If so, say why. Otherwise, give an example when it does not maximize the number of coffees prepared.

3. **College Choice.** A group of n high school seniors $S = \{s_1, s_2, s_3, \ldots, s_n\}$ are touring a set of colleges $C = = \{c_1, \ldots, c_n\}$ over the course of $m \geq n$ days this autumn. Each student s_j has an itinerary where he/she decides to visit one college per day (or maybe take a day off if $m > n$). However, these students are fiercely independent and prefer not to share college campus tours with other students. Furthermore, each student is looking for one college to call their own. Each student s would like to choose a particular day d_s and stay at his/her current college admissions office for the remaining $m - d_s$ days of the autumn (laying claim to 100% of the admissions officer's time). Of course, this means that no other students can visit c_s after day d_s, since students don't like sharing.

   Show that no matter what the students' itineraries are, it is possible to assign each student s a unique college c_s, such that when s arrives at c_s according to the itinerary for s, all

other students s' have either stopped touring colleges themselves, or s' will not visit c_s after s arrives at c_s. Describe an algorithm to find this *matching*.

Hint: The input is somewhat like the input to stable matching, but at least one piece is missing. Find a clever way to construct the missing piece(s), run stable matching, and show that the final result solves this problem.

4. **Party Planner Problem.** Imagine you are helping to plan an extravagant party. Among other tasks, you must help the host with the guest list. The host has provided you a list of people they might invite to the party, but they also come with demands — – Alice and Bob need to be invited, but if Bob is invited, do not invite Carol (they have history). When Carol, Dave, and Eve get toegether, they only talk about BTS (their favorite K-pop group), so don't invite all three. However, any two of them is ok. Call these conditions *constraints*.

   The host is very demanding. Everything needs to be exactly perfect, and they will blame you if even one constraint is not satisfied. You're not even sure if this is possible, and when it's not, it would be nice to have a convincing argument for the host.

   Design an algorithm that takes a list of people, and a list of constraints, and outputs YES if there is an invite list that satisfies every constraint. Otherwise, output NO.