CS 70 Fall 2022 — Discrete Mathematics and Probability Theory — Satish Rao and Babak Ayazifar — HW 02

Due: Saturday, 9/10, 4:00 PM
Grace period until Saturday, 9/10, 6:00 PM

# Sundry

Before you start writing your final homework submission, state briefly how you worked on it. Who else did you work with? List names and email addresses. (In case of homework party, you can just describe the group.)

# 1 Airport

Suppose that there are $2n+1$ airports, where $n$ is a positive integer. The distances between any two airports are all different. For each airport, exactly one airplane departs from it and is destined for the closest airport. Prove by induction that there is an airport which has no airplanes destined for it.

# 2 Proving Inequality

For all positive integers $n \geq 1$, prove that

$$\frac{1}{3^1} + \frac{1}{3^2} + \ldots + \frac{1}{3^n} < \frac{1}{2}.$$

(Note: while you can use formula for an infinite geometric series to prove this, we would like you to use induction. If you're having trouble with the inductive step, try strengthening the inductive hypothesis. Can you prove an equality statement instead of an inequality?)

# 3 Inductive Charging

There are $n$ cars on a circular track. Among all of them, they have exactly enough fuel (in total) for one car to circle the track. Two cars at the same location may transfer fuel between them. More formally:

- Order cars clockwise around the track, starting at some arbitrary car.
- Let the fuel in car $i$ be $f_i$ liters, where 1 liter of gas corresponds to 1 kilometer of travel.

- Let the track be $D$ kilometers around, and let the distance between car $i$ and car $i+1$ (in the modular sense) be $d_i$ kilometers.

(a) Prove, using whatever method you want, that there exists at least one car that has enough fuel to reach the next car along the track.

(b) Prove that there exists one car that can circle the track, by gathering fuel from other cars along the way. (That is, one car moving and all others stopped). Hint: Use the previous part. [1]

# 4 A Coin Game

Your "friend" Stanley Ford suggests you play the following game with him. You each start with a single stack of $n$ coins. On each of your turns, you select one of your stacks of coins (that has at least two coins) and split it into two stacks, each with at least one coin. Your score for that turn is the product of the sizes of the two resulting stacks (for example, if you split a stack of 5 coins into a stack of 3 coins and a stack of 2 coins, your score would be $3 \cdot 2 = 6$). You continue taking turns until all your stacks have only one coin in them. Stan then plays the same game with his stack of $n$ coins, and whoever ends up with the largest total score over all their turns wins.

Prove that no matter how you choose to split the stacks, your total score will always be $\frac{n(n-1)}{2}$. (This means that you and Stan will end up with the same score no matter what happens, so the game is rather pointless.)

# 5 Pairing Up

Prove that for every even $n \geq 2$, there exists an instance of the stable matching problem with $n$ jobs and $n$ candidates such that the instance has at least $2^{n/2}$ distinct stable matchings.

# 6 Nothing Can Be Better Than Something

In the stable matching problem, suppose that some jobs and candidates have hard requirements and might not be able to just settle for anything. In other words, each job/candidate prefers being unmatched rather than be matched with those below a certain point in their preference list. Let the term "entity" refer to a candidate/job. A matching could ultimately have to be partial, i.e., some entities would and should remain unmatched.

Consequently, the notion of stability here should be adjusted a little bit to capture the autonomy of both jobs to unilaterally fire employees and/or employees to just walk away. A matching is stable if

- there is no matched entity who prefers being unmatched over being with their current partner;

[1] To think about, after you complete this problem: Is your proof constructive or non-constructive? (That is, does it actually point to the exact car that can complete the track, or does it just prove that one such car must exist?) If it's non-constructive, then how do we actually find this car? (Can we write a program to do this, faster than actually trying every car?)

- there is no matched/filled job and unmatched candidate that would both prefer to be matched with each other over their current status;
- there is no matched job and matched candidate that would both prefer to be matched with each other over their current partners; and
- similarly, there is no unmatched job and matched candidate that would both prefer to be matched with each other over their current status;
- there is no unmatched job and unmatched candidate that would both prefer to be with each other over being unmatched.

(a) Prove that a stable pairing still exists in the case where we allow unmatched entities.

*(HINT: You can approach this by introducing imaginary/virtual entities that jobs/candidates "match" if they are unmatched. How should you adjust the preference lists of jobs/candidates, including those of the newly introduced imaginary ones for this to work?)*

(b) As you saw in the lecture, we may have different stable matchings. But interestingly, if an entity remains unmatched in one stable matching, it/she must remain unmatched in any other stable matching as well. Prove this fact by contradiction.