Problem Set 7 Solutions

This problem set is due **at 10:00pm** on **Wednesday, April 22, 2020**.

EXERCISES (NOT TO BE TURNED IN)

There are no exercises from the textbook this week.

Problem 7-1. Link Sharing [40 points]

Alice and Bob are renting an apartment together. Although they get along pretty well, the issue of sharing an internet connection has become quite contentious. Specifically, we model their sharing as a game in which the action of Alice (resp. Bob) corresponds to specifying a fraction $0 \leq x_A \leq 1$ (resp. $0 \leq x_B \leq 1$) of the connection they want to claim. Then, the utility u_A of Alice and the utility u_B of Bob become:

$$u_A = \begin{cases} 0 & \text{if } x_A + x_B > 1, \\ x_A(1 - x_A - x_B) & \text{otherwise;} \end{cases} \quad \text{and} \quad u_B = \begin{cases} 0 & \text{if } x_A + x_B > 1, \\ x_B(1 - x_A - x_B) & \text{otherwise.} \end{cases}$$

In the above, the top pair of cases ($x_A + x_B > 1$) corresponds to the internet link being overloaded, and the bottom pair reflects the need for having enough bandwidth left to maintain network speed.

(a) [10 points] Find a pair of actions x_A and x_B that give rise to maximal *total* utility $u_A + u_B$ for Alice and Bob.

Solution: We want to choose x_A and x_B so as to maximize

$$u = u_A + u_B = (x_A + x_B)(1 - (x_A + x_B))$$

subject to $x_A, x_B \geq 0$ and $x_A + x_B \leq 1$.
Noticing that the dependence of u (and the constraint) on both x_A and x_B is always through the sum, we can simplify by substituting $y = x_A + x_B$. After the substitution, our problem is that of maximizing $u = y(1-y)$ subject to $y \in [0, 1]$. This is easily seen (by symmetry, for example) to be maximized when $y = 1/2$. Therefore, any pair of actions such that $x_A + x_B = 1/2$ maximizes total utility. Taking, for example, $x_A = x_B = 1/4$ would work. For these actions, the total utility is $1/4$.

(b) [15 points] Show that the pair of actions $x_A = \frac{1}{3}$ and $x_B = \frac{1}{3}$ constitutes a Nash equilibrium of this game. What is the *total* utility $u_A + u_B$ it achieves?

Solution: To prove that a pair of actions constitutes a Nash Equlibrium, we need to show that none of the players have an incentive to deviate. Given the symmetric nature of the set-up, it is enough to consider Alice. Alice's utility at $(1/3, 1/3)$ is $1/9$. Keeping Bob's decision fixed, what would Alice's utility be if she changes her action from $1/3$ to $1/3 + \Delta$ for some Δ? Let's compute:

$$\begin{aligned} u_A &= (1/3 + \Delta) \cdot (1 - (1/3 + \Delta) - 1/3) \\ &= (1/3 + \Delta) \cdot (1/3 - \Delta) \\ &= 1/9 - \Delta^2 \\ &\leq 1/9 \end{aligned}$$

so Alice indeed does not have an incentive to deviate. In fact, any move from $1/3$ would result in strictly less utility for Alice.

(Many students arrived at a similar conclusion by computing the maximum possible utility for Alice while keeping Bob's action fixed at $1/3$.)

Finally, at $(1/3, 1/3)$, the total utility equals $1/9 + 1/9 = 2/9$.

As an aside, this is a good example of a "game" where coordinating players (part a) can attain greater total utility than competing players (part b). Indeed, $1/4 > 2/9$!

(c) [15 points] Alice and Bob managed to provide bandwidth via alternative means. As a result, their utilities are the following:

$$u_A = \begin{cases} 0 & \text{if } x_A + x_B > 1, \\ x_A & \text{otherwise;} \end{cases} \quad \text{and} \quad u_B = \begin{cases} 0 & \text{if } x_A + x_B > 1, \\ x_B & \text{otherwise.} \end{cases}$$

List *all* the *deterministic* Nash equilibria of this game.

Solution:

We can proceed by cases:

Case I: $x_A + x_B < 1$.

In this case, any player can increase their utility by increasing their action by a small amount while maintaining the constraint $x_A + x_B \leq 1$. Therefore, there are no Nash equilibria in this case.

Case II: $x_A + x_B = 1$.

In this case, any deviation for any player decreases their utility function. Indeed, consider Alice. If she increases her action, then $x_A + x_B > 1$, and she receives 0 utility. If she decreases her action, then her utility is similarly decreased by the corresponding amount. Therefore, any pair (x_A, x_B) such that $x_A + x_B = 1$ constitutes a Nash equilibrium.

Case III: $x_A + x_B > 1$.

In this case, the initial utility for both players is 0. If one of x_A or x_B is strictly less than 1, then the other player can increase their utility by decreasing their action so that $x_A + x_B = 1$. For example, assume that $x_A < 1$, Then Bob can decrease his action to $1 - x_A$ and receive a utility of $1 - x_A > 0$.

On the other hand, if both x_A and x_B are equal to 1, then the pair constitutes a Nash equilibrium. Indeed, in this case, no player can change to receive positive utility. Consider Alice. If Alice changes her action to $y_A > 0$, then $y_A + x_B > x_B = 1$ so Alice would still receive utility 0. If she changes to $y_A = 0$, then she also still receives utility 0, as $u_A = y_A = 0$. The analysis for Bob is analogous. Therefore, in this case, the pair $(x_A, x_B) = (1, 1)$ also constitutes a Nash equilibrium.

(Most students recognized the Nash equilibria from Case II. Only a handful also recognized the equilibrium from Case III.)

Problem 7-2. The Interview [15 points]

After stints at eLake and Pewlett-Hackard (PH), Wheg Mitman is now the CEO of Whobi, and she is determined to make Whobi the next big thing. She hears of your experience with algorithms and invites you to interview for a position on her board. Answer the following interview questions correctly to get the job.

(a) [5 points] **True or False:** In every instance of the Stable Matching Problem, there is a stable matching containing a pair (a, b) such that a is ranked first on the preference list of b and b is ranked first on the preference list of a. Justify your response with a short explanation or a counterexample.

Solution: This is false. One counterexample is the following: suppose that there are two groups A, B and P, Q and that the rankings are as follows:

- A prefers P, then Q
- B prefers Q, then P
- P prefers B, then A
- Q prefers A, then B

There are no pairs such that a is ranked first on the preference of b and b is ranked first on the preference list of a. Thus, there cannot be a stable matching that satisfies this condition.

(b) [10 points] Suppose we have two television networks, called A and B. There are n prime-time programming slots, and each network has n TV shows. Each network needs to devise a *schedule*—an assignment of each show to a distinct slot, to attract as much market share as possible.

Each show has a fixed *rating*, and we'll assume that no two shows have exactly the same rating. The winner of each time slot is the network whose show has a larger rating. Each network wants to win as many slots as possible.

This season, network A reveals a schedule S, and network B reveals a schedule T. We will define a pair of schedules (S, T) to be *stable* if neither network can unilaterally change its own schedule and win more time slots. That is, there is no schedule S' such that A wins more time slots with (S', T) than it did with (S, T), and symmetrically, there is no schedule T' such that B wins more time slots with (S, T') than it did with (S, T).

For every set of TV shows and ratings, is there always a stable pair of schedules? Justify with a short explanation or a counterexample.

Solution: Such a stable pair of schedules does not always exist. A counterexample follows.

Suppose there are two time slots, network A has shows ranked 1 and 3, and network B has shows ranked 2 and 4. Then, it can be shown that there is no stable arrangement of shows to time slots.

One possible situation is that the two networks A and B assign programs to slots as follows:

	A	B
T1	1	2
T2	3	4

Then network A loses both slots and is incentivized to switch its programs between the two slots, yielding

	A	B
T1	3	2
T2	1	4

This now gives the other possible assignment of programs to slots, which yields one win for each network. But now network B can switch its programs, yielding

	A	B
T1	3	4
T2	1	2

Again, network B now wins both slots, and this assignment is essentially tantamount to the first but with the slots switched. Network A can again swap its programs, yielding

	A	B
T1	1	4
T2	3	2

Now, each network wins one slot again, and network B swaps its programs to yield the original assignment in which it wins both slots.

Therefore, we will cycle between these four possible assignments, and there is no stable assignment.

Problem 7-3. Who Goes Where? [35 points]

Wheg Mitman is pleased with your performance and offers you a spot on her board! She decides that the first step to improving the productivity at Whobi is to promote good connections between employees and the divisions of Whobi to which they belong.

In order to do so, Wheg organizes the hiring and matching process as follows. Whobi has m divisions, and each division d_i has c_i openings, where we define $c = \sum_{i=1}^{m} c_i$. Wheg has selected $n > c$ potential hires to come on-site and interview with each of the m divisions of Whobi. Each potential hire, h_j, ranks all m divisions of Whobi, and each division of Whobi, d_i, ranks all n potential hires. Finally, they all submit their rankings to Wheg, who matches the potential hires to the divisions, ensuring that each division's positions are filled.

Wheg recruits you to help her with this problem. She explains to you that after researching the stable matching problem, she has determined that she would like her matching to satisfy some notion of stability. In particular, she would like to prevent certain types of instabilities that indicate inefficient matchings. We say that an assignment of potential hires to divisions that fills all openings is *stable* if no such type of instability is present in the assignment.

One type of instability that she has borrowed from the stable matching problem with two groups of equal size can be stated as follows:

Instability Type #1: There are potential hires h_α and h_β and divisions d_i and d_j such that

- h_α is assigned to d_i
- h_β is assigned to d_j
- d_i prefers h_β to h_α
- h_β prefers d_i to d_j

However, Wheg understands that her problem is slightly different, since her divisions would like to hire multiple employees, and she has an excess of potential hires. Therefore, she asks you to consider if there might be some other type of instability that can exist in an assignment.

(a) [5 points] Help Wheg precisely formulate another type of instability that can exist in an assignment of potential hires to divisions. (Hint: consider those potential hires who are not assigned to any divisions by a given assignment.) Formally state the conditions for this type of instability.

Solution: The other type of instability possible in the problem is as follows:
Instability Type #2: There are potential hires h_α and h_β and a division d_i such that

- h_α is assigned to d_i

- h_β is assigned to no division
- d_i prefers h_β to h_α

(b) [30 points] As discussed above, an assignment of potential hires to departments that exhibits neither the instability type proposed by Wheg nor the one that you proposed in part (a) is called a stable assignment. Propose an efficient algorithm to make a stable assignment of potential hires to divisions, and prove that this algorithm always terminates with the guarantee of a stable assignment. In addition, analyze the number of offers made by this algorithm.

Solution: The general idea of our algorithm is that we will allow a division that has not yet filled its openings to "propose" to its favorite potential hire. If a potential hire is not already "engaged," then the potential hire will temporarily accept the offer. Otherwise, if the potential hire is already "engaged," then the potential hire will drop the previous offer and temporarily accept the new offer if the potential hire prefers this division and reject the new offer otherwise. This process repeats until all divisions have filled their openings.

A precise description of the algorithm follows:

1. Each division d_i maintains an opening count o_i representing how many of its positions are currently unfilled, and this count is initialized to c_i. In addition, each potential hire h_j maintains a current best offer, which is initialized to be empty.
2. While there is a division d_i for which $o_i > 0$:
   (a) Choose a division d_i with $o_i > 0$
   (b) Let h_α be the highest-ranked potential hire in d_i's preference list to whom d_i has not yet proposed an offer
   (c) If h_α has no current offer, then h_α accepts the offer, and o_i is decremented by one
   (d) Else h_α has previously accepted an offer from some other division d_j
      i. If h_α prefers d_j to d_i, then h_α rejects d_i's offer and the slot remains free
      ii. Else h_α prefers d_i to d_j, so h_α accepts d_i's offer and rejects d_j's offer, meaning that o_i is decremented by one, and o_j is incremented by one
3. All divisions have been filled, so all potential hires officially accept their offers, and the assignment is returned

First, we would like to prove that this algorithm terminates. To see that the algorithm terminates, we prove that the maximum number of potential hires to whom any division will make offers is $c = \sum_{i=1}^{m} c_i$, the total number of openings in all m divisions. Suppose that division d_i has already made offers

to c potential hires. Then c potential hires must already be matched with an opening in a division, since each potential hire to which d_i made an offer must either have accepted d_i's offer or already been matched to another division that this potential hire prefers. Moreover, all c potential hires must still be matched at this point in time because once a potential hire has been matched, this potential hire will only "trade up" and thus always remain matched. Having demonstrated that $c = \sum_{i=1}^{m} c_i$ potential hires are matched with openings in divisions, it follows that every opening has been filled. In particular, d_i has no unfilled openings and can thus no longer propose another offer. This proves that each division can propose no more than c offers.

If each of the m divisions proposes at most c offers, then the algorithm will terminate after at most $m \cdot c$ offers are made, and an assignment A of potential hires to divisions will be generated.

We would also like to prove that every opening has been filled when the algorithm terminates. Given the structure of this algorithm, this is fairly clear. Suppose that the algorithm has terminated but that some opening has not been filled in the resulting assignment A. Then if this opening is in division d_i, we must have $o_i > 0$, in which case the algorithm will not have terminated yet, a contradiction. Therefore, all openings are filled in the assignment A produced upon termination of the algorithm.

Now, it remains to prove that the assignment A of potential hires to divisions is stable.

To do so, we assume that there is an instability in assignment A generated by the algorithm and find a contradiction.

Instability Type #1: Assume that the first type of instability exists in assignment A. In particular, assume that there are pairs (d_i, h_α) and (d_j, h_β) in the assignment but that d_i prefers h_β to h_α, and h_β prefers d_i to d_j.

In the algorithm, d_i made offers in order of preference until all openings are filled. By assumption, d_i prefers h_β to h_α, so d_i would have made an offer to h_β before making an offer to h_α. Because h_l is not matched with d_i, it follows that the offer that d_i made to h_β was either rejected or accepted then rejected later in favor of some division d_k that h_β prefers to division d_i. Since d_j is the eventual match for potential hire h_α, we must have that $d_k = d_j$ or that h_β prefers d_j to d_k. In either case, since h_β must prefer d_k to d_i to have rejected d_i's offer, it follows by transitivity that h_β must prefer d_j to d_i. This provides a contradiction to the assumption that h_β prefers d_i to d_j.

Instability Type #2: Assume that the second type of instability exists in assignment A. In particular, assume that h_α is assigned to d_i, h_β is assigned to no division, and d_i prefers h_β to h_α.

In the algorithm, d_i makes offers in order of preference, and any potential hire who has not yet received an offer accepts an offer from any division and never

again is unpaired. Because h_β is not assigned to a division, it follows that h_β must have received no offers. However, had d_i preferred h_β to h_β, as per the assumption, then d_i would have made an offer to h_β before making an offer to h_α. This provides a contradiction to the assumption.

Therefore, neither type of instability can occur in the assignment A, and so the assignment generated by the algorithm must be a stable assignment of potential hires to divisions.

(Note that there is an alternate solution in which potential hires propose trial matches to divisions and that the algorithm in this solution can be analyzed similarly, although the upper bound on the number of proposals made is nm rather than mc and is thus not quite so tight; it is a worthwhile exercise to consider why this is so.)

Problem 7-4. Feedback Form [10 points] Please fill out a feedback form about this problem set at

`https://forms.gle/85beLrcf1ZA4cnMB8`.

It should not take more than a few minutes and will greatly help us improve teaching and material for future semesters!