**Homework 1**
**Sara Pesavento**
**spesaven@usc.edu**
**7618843154**

1. **Reading Assignment: Kleinberg and Tardos, Chapter 1.**
   Completed.

2. **Solve Kleinberg and Tardos, Chapter 1, Exercise 1.**
   Decide whether you think the following statement is true or false. If it is true, give a short explanation. If it is false, give a counterexample.
   *True or false? In every instance of the Stable Matching Problem, there is a stable matching containing a pair (m, w) such that m is ranked first on the preference list of w and w is ranked first on the preference list of m.*
   **False**. Let us look at a counterexample - a case where the above statement does not hold.

| m | m' | w | w' |
|---|---|---|---|
| w | w' | m' | m |
| w' | w | m | m' |

   This is **an** instance of the Stable Matching Problem which results in (m, w) and (m', w'). There are no instabilities because both m and m' are matched with their first preference and have no incentive to leave their current partner. However, in this stable matching, the women are not with their first preference. So there is no pair (m, w) such that w is ranked first on m's preference list and m is ranked first on w's preference list.

3. **Solve Kleinberg and Tardos, Chapter 1, Exercise 2.**
   Decide whether you think the following statement is true or false. If it is true, give a short explanation. If it is false, give a counterexample.
   *True or false? Consider an instance of the Stable Matching Problem in which there exists a man m and a woman w such that m is ranked first on the preference list of w and w is ranked first on the preference list of m. Then in every stable matching S for this instance, the pair (m, w) belongs to S.*
   **True**. Recall when men propose, they end up with their best valid partners (valid partner meaning they appear in some stable matching, best meaning it is the highest stable match in their preference list).
   Proof by contradiction:

| m | m' | w | w' |
|---|---|---|---|
| w | w | m | m |
| w' | w' | m' | m' |

   In the above example, suppose we had the pairs (m, w') and (m', w). This would be an instability.
   - If m was rejected by a valid partner w, in favor of m', then that m' would need to be higher on w's preference list - a contradiction.
   - If w is not matched with m, then that means that m never proposed and she had to settle for m'. That means that w was not first on the preference list of m - a contradiction.

Since this is not the case, the pair (m, w) must belong to S.

4. **State True/False: An instance of the stable marriage problem has a unique stable matching if and only if the version of the Gale-Shapley algorithm where the male proposes and the version where the female proposes both yield the exact same matching.**

**True**. Consider the following case which results in the same stable matching:

| M1 | M2 | M3 | W1 | W2 | W3 |
|---|---|---|---|---|---|
| W1 | W1 | W1 | M1 | M2 | M3 |
| W2 | W2 | W2 | M2 | M3 | M2 |
| W3 | W3 | W3 | M3 | M1 | M1 |

In this situation,

- If men propose: (M1, W1), (M2, W2), (M3, W3)
- If women propose: (M1, W1), (M2, W2), (M3, W3)

Here every man and woman ends up with their best valid partner across all stable matchings.

> Note: A pair (m, w') is called a blocking pair if m prefers w' to w and w' prefers m to m'. A blocking pair is likely to abandon their respective mates and match up with each other; thus a matching containing a blocking pair is not stable

For M1 and W1, they rank each other highest, thus they must end up together or they become a blocking pair. Now for M2, W1 is taken, so they are matched with W2, which is the highest among their remaining preference list. For M3, matching with W1 and W2 would form blocking pairs, so it is matched with W3, the remaining choice. There is nothing of lower order! So since each pair is matched with their best valid partner (no lower order is preferred), then we have a unique equilibrium.

Let us show an example where the G-S algorithm where the male and female proposes result in a different matching - hence where we do not get a unique stable matching. In the following case, two different stable matchings occur: (M, W) and (M', W') OR (M', W) and (M, W').

**M M' W W'**
W W' M' M
W' W M M'

5. **A stable roommate problem with 4 students a, b, c, d is defined as follows. Each student ranks the other three in strict order of preference. A matching is defined as the separation of the students into two disjoint pairs. A matching is stable if no two separated students prefer each other to their current roommates. Does a stable matching always exist? If yes, give a proof. Otherwise give an example roommate preference where no stable matching exists.**

No, a stable matching does not always exist. Consider:
a -> b, c, d
b -> a, c, d
c -> a, b, d
d -> a, b, c

Here a and b end up roommates. Then c is stuck with their last choice, d, even though it prefers a or b from the other disjoint pair.

6. **Solve Kleinberg and Tardos, Chapter 1, Exercise 3.**

There are many other settings in which we can ask questions related to some type of "stability" principle. Here's one, involving the competition between two enterprises. Suppose we have two television networks, whom we'll call A and B. There are n prime-time programming slots, and each network has n TV shows. Each network wants to devise a schedule—an assignment of each show to a distinct slot—so as to attract as much market share as possible. Here is the way we determine how well the two networks perform relative to each other, given their schedules. Each show has a fixed rating, which is based on the number of people who watched it last year; we'll assume that no two shows have exactly the same rating. A network wins a given time slot if the show that it schedules for the time slot has a larger rating than the show the other network schedules for that time slot. The goal of each network is to win as many time slots as possible. Suppose in the opening week of the fall season, Network A reveals a schedule S and Network B reveals a schedule T. On the basis of this pair of schedules, each network wins certain time slots, according to the rule above. We'll say that the pair of schedules (S, T) is stable if neither network can unilaterally change its own schedule and win more time slots. That is, there is no schedule S' such that Network A wins more slots with the pair (S', T) than it did with the pair (S, T); and symmetrically, there is no schedule T' such that Network B wins more slots with the pair (S, T') than it did with the pair (S, T). The analogue of Gale and Shapley's question for this kind of stability is the following: For every set of TV shows and ratings, is there always a stable pair of schedules? Resolve this question by doing one of the following two things:

(a) give an algorithm that, for any set of TV shows and associated ratings, produces a stable pair of schedules; or

(b) give an example of a set of TV shows and associated ratings for which there is no stable pair of schedules.

There is not always a stable pair of schedules. Suppose Network A has two shows (a1, a2) with ratings 30 and 55. Network B has two shows (b1, b2) with ratings 5 and 40. Each network can reveal two schedules (total of 2x2 = 4, but for the following if both swapped it'd be the same situation/dilemma):

| A | B |
|---|---|
| 30 | 5 |
| 55 | 40 |

*Unstable: If this schedule came out, Network A would get two slots and Network B would get 0. In this case, Network B would want to swap it's shows so that 40 > 30 and it gets 1 slot instead of 0.

| A | B |
|---|---|
| 30 | 40 |
| 55 | 5 |

*Unstable: If this schedule came out, Network A would get 1 slot and Network B would get 1 slot. In this case, Network A would want to swap it's shows so that 30 > 5 and 55 > 40 and it gets 2 slots instead of 1.

7. **Solve Kleinberg and Tardos, Chapter 1, Exercise 4.**

Gale and Shapley published their paper on the Stable Matching Problem in 1962; but a version of their algorithm had already been in use for ten years by the National Resident Matching Program, for the problem of assigning medical residents to hospitals. Basically, the situation was the following. There were m hospitals, each with a certain number of available positions for hiring residents. There were n medical students graduating in a given year, each interested in joining one of the hospitals. Each hospital had a ranking of the students in order of preference, and each student had a ranking of the hospitals in order of preference. We will assume that there were more students graduating than there were slots available in the m hospitals. The interest, naturally, was in finding a way of assigning each student to at most one hospital, in such a way that all available positions in all hospitals were filled. (Since we are assuming a surplus of students, there would be some students who do not get assigned to any hospital.)

We say that an assignment of students to hospitals is stable if neither of the following situations arises.

First type of instability: There are students s and s', and a hospital h, so that

- s is assigned to h, and
- s' is assigned to no hospital, and
- h prefers s' to s.

Second type of instability: There are students s and s', and hospitals h and h', so that

- s is assigned to h, and
- s' is assigned to h', and
- h prefers s' to s, and
- s' prefers h to h'.

So we basically have the Stable Matching Problem, except that (i) hospitals generally want more than one resident, and (ii) there is a surplus of medical students. Show that there is always a stable assignment of students to hospitals, and give an algorithm to find one.

Have hospitals h and h' rank their student preferences. Have students rank their hospital preferences. At any point in time, a student is either committed to a hospital or free. A hospital either has available positions or is full. Here is the algorithm:

```
while there is a hospital hi that has available positions:
      accept the first student si on it's preference list
      if the student si is free/unmatched:
            student si accepts the offer
      else student already accepted hospital hj:
            if student si prefers hospital hj to hi:
                  reject hi and remain committed to hj
            else si revokes their offer and joins hospital hi:
                  # of available positions at hospital hj increases by one
                  # of available positions at hospital hi decreases by one
```

If there are m hospitals and n students, then the algorithm will terminate in O(mn) steps because each hospital offers a position to a student at most once, and in each iteration, some hospital offers a position to some student. The algorithm terminates when all positions (p) are filled, since they must offer one to every student and p < n. Let us argue that the assignment is stable by looking at the two forms of instability.

1. Instability 1 involves a case where s is assigned to h, but h prefers s' and s' is assigned to no hospital. This cannot happen, as if h prefers s', then h must have accepted s' first before s. In that case, s' would have accepted a position for some hospital and it would not be free (either hopsital h or it might have accepted another hospital that ranked higher for it) - a contradiction.

2. Instability 2 involves when s is assigned to h and s' is assigned to h'. h prefers s' to s and s' prefers h to h'. This cannot happen because if h prefers s' to s, it must have extended an offer to it before s. That means at some point s' turned down h for a hospital that was higher ranking on its preference list. Therefore s' cannot prefer h to its current match. - this is a contradiction.

8. **N men and N women were participating in a stable matching process in a small town named Walnut Grove. A stable matching was found after the matching process finished and everyone got engaged. However, a man named Almazo Wilder, who is engaged with a woman named Nelly Oleson, suddenly changes his mind by preferring another woman named Laura Ingles, who was originally ranked right below Nelly in his preference list, therefore Laura and Nelly swapped their positions in Almanzos preference list. Your job now is to find a new matching for all of these people and to take into account the new preference of Almanzo, but you don't want to run the whole process from the beginning again, and want to take advantage of the results you currently have from the previous matching. Describe your algorithm for this problem. Assume that no woman gets offended if she got refused and then gets proposed by the same person again.**

Now that Almazo prefers Laura, he can propose to her.

- If she prefers her current partner to Almazo, then he is rejected and will need to propose to Nelly again. The algorithm stops and everything is stable.
- If she prefers Almazo, then she breaks up with her current man M in favor of Almazo. Now man M can go down his list. Nelly is the woman available but she may be very far down his list compared to Laura! Also, some other men might prefer Nelly and they got rejected in the past because she preferred Almazo to them.

The following steps need to be completed to solve the instabilities:

- Because of the instability created, one could put all the men who were rejected by Nelly in favor of Almazo into a pool. And put Nelly and the prior fiances of the free men into another pool. This will create more instabilities similar to the one created by freeing up Nelly. So we would need to recursively build pools with free men and free women.
- Then run the G-S algorithm as usual on the pool of free men, each moving down their preference lists. In doing so, if they propose to someone they haven't proposed to in the past, this will cause more men (and women) to become free again as well.
- Once this algorithm is done, we will be back to a stable matching.

| Almazo | Almazo' | Laura | Laura' | M' |
|---|---|---|---|---|
| …. | …. | …. | …. | …. |
| Nelly | Laura | M | Almazo | Nelly ← would rather match |
| Laura | Nelly | …. | …. | …. |
| …. | …. | …. | M | W |
| …. | …. | Almazo | …. | …. |