Game Theory
Lecture #4 – Matching

Focus of Lecture:

- The Marriage Problem
- The Roommates Problem

1 Matching Problems

In the social choice paradigm of the previous chapter, we found that Arrow's Theorem guarantees that there are no fully satisfying mechanisms for aggregating the preferences of many into a single monolithic societal preference. In a sense, we saw that every imaginable social choice mechanism has some undesirable flaw. As we move forward, we will keep this in mind: in social systems, not all problems are completely solvable in a satisfying way.

Fortunately, what we saw in social choice is not completely universal – some problems relating to social systems do actually have elegant and satisfying solutions; in this chapter we will investigate some of these in the context of *matching*. Matching problems take many forms; in this lecture, we will focus on scenarios in which a matching is specified by a set of mutually-exclusive *pairs* of items from some larger set. The following applications give examples of the types of matching problems we may consider.

Application 1.1 (Load Balancing in Distributed Computer Systems) *Suppose you operate a large video streaming platform with hundreds of servers and millions of users. At any given time, thousands of users are currently requesting to watch cat videos, and each user needs to be connected with a server so their video can be streamed. It is important to use the content delivery network efficiently (so that the servers are all mostly operating at a similar load), and it is also important to ensure low latency and quick service to users so they don't leave your platform for some other platform that treats them better. Is there a way to assign users to content servers that is fair, efficient, and keeps everyone happy?*

Application 1.2 (Matching Drivers and Riders in a Ridesharing Platform) *On a ridesharing platform such as Uber and Lyft, millions of rides are requested per day across the world. Each time a rider requests a ride, the platform must find a driver to take that rider to her destination. To ensure a timely pickup, that driver should currently be as close as possible to that rider. Furthermore, the platform could take the rider's destination into account when selecting drivers, as some drivers may prefer not to accept trips that take them too far from home. In other words, riders have preferences over drivers, and drivers have preferences over riders. Is there a fast algorithm that can perform these matches?*

Application 1.3 (Medical Resident Matching) *Every year in the United States, some 40,000 medical students graduate from medical school every year and enter the market for a* *medical residency (a critical phase of medical training). Various hospitals around the country run residency programs which hire these new graduates; there are somewhere in the vicinity of 32,000 open positions each year. Some of the features of this problem are:*

- *Students have preferences over residency programs.*
- *Residency programs have preferences over student applicants.*
- *The number of applicants is typically greater than the number of positions.*

Is there a good way to match hospitals with residents? Can we make sure that the hospitals and residents are satisfied with the outcome of the matching?

1.1 The Marriage Problem

In this lecture, our primary focus will be on matching problems in which each of the items (persons, drivers, servers) to be matched has preferences about who or what it is matched with. As a general framework for modeling this, we will use the classic problem known as the "marriage problem." In an instance of the marriage problem, there are two types of item to be matched, and N of each type. Classically, one type is called "men" and the other type is called "women." Each man has a ranked list of all the women which describes his preferences over the women, and each woman has such a list of the men. For example, an instance of the marriage problem with $N = 4$ is given by the following preference rankings:

	Ann	Beth	Cher	Dot
Al	1	1	3	2
Bob	2	2	1	3
Cal	3	3	2	1
Dan	4	4	4	4

Women's Preferences

	Ann	Beth	Cher	Dot
Al	3	4	1	2
Bob	2	3	4	1
Cal	1	2	3	4
Dan	3	4	2	1

Men's Preferences

The left grid expresses the womens' rankings of the men; the right grid expresses the mens' rankings of the women. In this example (reading the leftmost column on the left grid), Ann likes Al best, then Bob, then Cal, and likes Dan the least. On the other hand (reading the uppermost row in the right grid), Al likes Cher best, then Dot, then Ann, and likes Beth the least.

The goal of the marriage problem is to assign each man to a unique woman and each woman to a unique man in the best way possible. But what does "best way possible" mean in this context? To explore this question, consider the following potential matching where each person has been matched with (where possible) their 2nd or 3rd choice:

Al	Bob	Cal	Dan
|	|	|	|
Dot	Ann	Beth	Cher
(2×2)	(2×2)	(2×3)	(2×4)

This is clearly a valid matching (as each man is matched to a unique woman, and vice versa), but it it a "good" matching? To see if it is, we might imagine going from person to person and asking each one "are you happy with your mate?" For instance, if we asked Al, he would say "I only got my 2nd choice Beth – but I would have preferred to be matched with Cher." Al might then head over to Cher to find out where she stands on the matter. To Al's delight, Cher was matched with her least favorite (Dan), so she's willing to take any other man that comes her way and would accept a proposal from Al. Thus, the matching above is problematic: Al and Cher both prefer each other to their matches. We say that this matching is not *stable.*

Definition 1.1 (Stable Matching) *We say that a matching between N men and N women is* stable *if no pair of unmatched mates prefer each other to their matches.*

It may be simpler to state this definition in terms of what makes a matching fail to be stable: suppose that man m is matched to woman w', and woman w is matched to man m'. If man m prefers w over his match w', *and* woman w prefers man m over her match m', then this matching is not stable.

Stability seems to be an important property: if a matching is stable, then at the very least, it is not easy to break. Let us try another matching; this time, we will construct the matching by going down the list of women and giving each her first choice:

Al	Bob	Cal	Dan
|	|	|	|
Ann	Cher	Dot	Beth
(3×1)	(4×1)	(4×1)	(4×4)

Is this matching stable? Ann, Cher, and Dot each have their first choice, so they would not accept any proposals from other men. But Beth might, since she is matched with her last choice so she would take anybody else other than her match. Likewise, *both* Bob and Cal are matched with their last choices, so either of them would take any proposal by anybody other than their matches. Thus, without even looking at the full preference lists we can see that Beth and Bob would prefer each other to their matches, and so would Beth and Cal – so this too is not a stable matching.

Before taking a more formal approach to the problem, let us try one more matching; this time by going down the list of men and assigning their top choices:

Al	Bob	Cal	Dan
|	|	|	|
Cher	Dot	Ann	Beth
(1×3)	(1×3)	(1×3)	(4×4)

Now Al, Bob, and Cal each have their first choice – so they will reject any proposal from other women. In particular, this means that Beth cannot get out of her match with Dan, even though he is her last choice. So if this matching is not stable, it will have to be Dan that breaks it up. However, Dan has a problem: he is every woman's last choice, so every other woman will reject his proposal. Thus, since all of the men are either satisfied or not preferred, this matching is stable.

Finally, in this instance, we were able to find a stable matching – though it is not obvious how easy this will be in general or whether or not a stable matching even exists. Furthermore, the women did not fare particularly well in the stable matching shown for this example. This exercise raises some important questions:

- Is stability an *interesting* property, in the sense that it is achievable? That is, are stable matchings guaranteed to exist?
- If we find a stable matching, in what sense is it actually *good*?

2 Stable Roommates

Before we tackle the stable *marriage* problem, it is worth our time to take a look at an ostensibly simpler version of the problem, known as the stable *roommates* problem. Note that the stable marriage problem asks for a bipartite matching – that is, the group of people is divided into two groups, and a person cannot match with someone from their own group. It seems that the problem could be simplified by removing the bipartite constraint, and allowing anybody to match with anybody. This simplified problem is called the stable roommates problem because it models a situation in which each of an even-sized group of university students submits a preference ranking of all the other students, and the matching algorithm's goal is to pair each student off with one other student in a desirable way. We may ask the same question here that we did in the stable marriage problem: do any stable matchings exist?

2.1 An Example

Let us begin with a simple example, with only four roommates, whom we will call Alice, Bob, Carol, and Dan, or $\{A, B, C, D\}$. Each roommate submits a preference ranking of the other 3; suppose these preferences are given by the rows in the following figure:

	A	B	C	D
A	-	1	2	3
B	2	-	1	3
C	1	2	-	3
D	1	2	3	-

Roommates' Preferences

That is, Alice prefers Bob over Carol, and Carol over Dan, and so forth. Is there a stable matching associated with these preference rankings? This example is small enough that we could simply enumerate all matchings to find the answer. First, what if Alice is matched with Bob (so that Carol is matched with Dan)? This matching fails to be stable, since Bob and Carol would rather be matched with each other than to their assigned matches.

However, it is not stable to have Bob and Carol assigned to one another (with Alice and Dan assigned to one another), since here again Alice and Carol would rather break their matches to be with each other. Once more, we can check if an Alice/Carol and Bob/Dan match is stable – and find that it is not, since Alice and Bob now prefer one another over their matches.

Since there are only 3 possible matchings between 4 roommates, the above search shows that no stable matching exists since we checked all 3 matchings. Therefore, we find that in the relatively simple roommates' problem, we are not able to obtain a guarantee that a stable matching exists.

Does this negative result apply to the marriage problem? That is, are there preference profiles in the marriage problem that do not allow for stable matchings? On the surface, it seems as though the roommates problems is strictly simpler than the marriage problem, since the marriage problem has the additional constraint that it must return a bipartite matching. It would seem that if no stable matchings of any kind are possible, simply adding a constraint does not seem likely to improve things.

3 Conclusion

Thus far, we have seen several practical examples of matching problems which can be modeled using the Stable Marriage problem. We have also seen the roommates problem, and learned that stable matchings are not guaranteed to exist in that context. Moving forward, several questions remain:

1. Do stable matchings exist for the stable marriage problem?
2. If a stable matching exists for an instance of the problem, is the matching unique?

3. Given an instance of the problem, how can we determine algorithmically whether a stable matching exists?

4. Are there efficient algorithmic approaches to this problem?

5. In what sense is stability "good" – namely, if a stable matching is found, can we argue that it is also optimal in some sense?

4 Questions

1. Consider the Roommate Problem discussed in lecture where we have the following two sets of preferences

	A	B	C	D
A	-	3	1	2
B	2	-	1	3
C	2	3	-	1
D	1	3	2	-

Roommates' Preferences

	A	B	C	D
A	-	1	3	2
B	2	-	1	3
C	2	1	-	3
D	2	3	1	-

Roommates' Preferences

Is there a stable division into pairs for either of the preference structures above? If so, provide an example. If not, show that no stable matching system exists by checking all the possibilities.

2. Consider the Marriage Problem with the following set of preferences

	A	B	C
a	3	2	1
b	2	3	3
c	1	1	2

Women's Preferences

	A	B	C
a	2	1	3
b	3	2	1
c	3	1	2

Men's Preferences

Does a stable matching exist? If so, characterize all the stable matchings?