136 Final Project: Stable Partners Problem

Lily Orgeron and Ayana Yaegashi

December 8, 2023

1 Introduction

In the Stable Marriage Problem, we start with two non-overlapping groups of agents. Each agent in one group has a complete preference order of all agents in the other group. One of the most effective and efficient ways to find a stable matching between these two groups is the Gale-Shapley Deferred Acceptance Algorithm, in which one group continuously proposes to the other and the proposed group can hold onto a proposal from the proposing group until a better one comes along. Similarly, the proposing group will move down their lists of preferences proposing until they are not rejected. This algorithm always finds a stable matching, and in this stable matching, it is always weakly the best stable matching for the proposing group and weakly the worst stable matching for the proposed group.

In a binary-gendered society where only heterosexuality is relevant, this Stable Marriage Problem would work wonderfully in finding stable partnerships. However, this version of society is not reflective of our reality, in which there exists a multitude of genders and sexualities. This begs the question, then: how can we find an effective and efficient way of finding stable partnerships for a general pool of participants, with a variety of genders and sexualities, while also maximizing their happiness with the outcome?

Enter the Stable Roommates problem. In the Stable Roommates problem, we have only one group of people, no longer separated into two groups, and we wish to find a stable matching such that as many participants as possible are matched with one and only one other participant, and there are no blocking pairs.

The goal of our project here is to build a prototype for a date-matching simulator, modelling it as the Stable Roommates problem, and we will be using Robert W. Irving's 1984 paper, 'An Efficient Algorithm for the "Stable Roommate" Problem' as a blueprint. Of course, we will be making our own adjustments to achieve fit our own needs. Since a person can have a sexuality that can exclude participants that fall under a specific gender category, we plan to effectively truncate participants' preference lists to only include participants that fall under the genders that they are attracted to. Furthermore, Irving's Stable Roommates algorithm only finds a stable matching if one exists, and it does not return a matching at all if no stable matchings exist. Because we wish for our dating simulation to maximize utility of participants, we will be altering the algorithm to return a matching regardless of stability. We will ensure to specify if the matching found is stable or unstable. Furthermore, our simulation will return a matching, with no requirement that every single participant is matched.

Moreover, our simulation will test two scenarios: when all agents play truthfully and when all agents play strategically. That is, participants can report their preferences truthfully, or they can lie and report strategically in hopes of increasing their utility. In either case, preference order will be based on a "desirability score." Each participant will be randomly and uniformly assigned a numerical

representation of their attractiveness, with 100 being the highest attractiveness possible and 0 being the lowest attractiveness possible. When participants are truthful, they will rank other participants (among the genders they are attracted to) from highest to lowest desirability. When participants are strategic, they will rank other participants by proximity to their own desirability score: the closer another participant's desirability score to their own, the higher on their list they appear. Of course, we also keep track of the strategic participants' truthful preferences to later calculate utility.
In regards to utility, as a baseline, an unmatched participant will have a utility of 0. We then let the highest possible utility be n, where n is the number of participants in the simulation. A matched participant's utility, then, is $n - k$, where k is the index of their match in their truthful preference list (this list is 0-indexed). We then can calculate the average utility per participant as a normalized measure of effectiveness of our simulation and happiness of participants.
Lastly, sexuality can either be assigned to participants uniform randomly ("Random"), such that every possible sexuality is evenly weighted (possible sexualities being [M,F,NB], [M,F], [M,NB], [F,NB], [M], [F], [NB]), or "Weighted" such that [M,F,NB] (attracted to all genders) is significantly more likely than any other.

2 Code

2.1 Agent

We created two agent types, a truthful agent and a strategeic agent. Both agents have the following populated properties:

```
id
gender
sexuality
desirability
preference_order_list
cutoff
```

In order, the "id" is a unique identifier for every agent. "Gender" is one of three values: 0, 1, or 2 corresponding to female, male, and non-binary. We recognize that despite our intention to make the algorithm more expansive of different gender identities, this three-gender structure is still limited. However, we believe it is still useful as it is a step more expansive than just two genders as in the stable marriage problem. "Sexualtiy" is a sublist of $[0, 1, 2]$ and thus includes the genders that the agent can be attracted to. "Desirability" is a randomized value between 0 and 100, where higher values indicate the agent being viewed by other agents as more desirable. A limitation to note here is that because we have made "desirability" a global variable, an assumption that does not take into account subjectivity. "Preference_order_list" is a list of the other agents in the simulation, ordered by the current agent's preference. Because "preference_order_list" includes all agents, even those outside of the agent's sexuality, those who fall outside of the agent's sexuality are all at the end of the "preference_order_list". The "cutoff" of an agent is the index of the last acceptable agent to be paired with in their "preference_order_list".
In addition, once the algorithm is run, each agent also has the following three properties:

```
paired_with
just_rejected
ask_rank
```

"paired_with" is populated with the agent object that the current agent is matched with in the matching algorithm, or None if they are not matched. "just_rejected" is a field used to keep track of the agent they previously rejected during the rounds of proposals during the matching algorithm. "ask_rank" is the index in the agent's "preference_order_list" of their final match.
The truthful and strategic agents differ in how they order their "preference_order_list". The truthful agent will order their list by the desirability of the potential matches that align with their sexuality. We implement this as follows:

```
attracted = sorted(attracted, lambda x: x.desirability, reverse=True)
```

On the other hand, the strategic agent tries to order agents by how closely they fall to their own desirability, in order to maximize the chances of being matched successful. This strategic behavior comes from the fact that not every agent will be matched in our matching algorithm described below, due to the truncated preference lists caused by multiple sexualities. We implement this as follows:

```
attracted = sorted(attracted,
                   lambda x: abs(self.desirability - x.desirability),
                   reverse=True)
```

2.2 Matching Algorithm

```
    def runMatch(self):
        set_proposed_to = set()
        for participant in self.participants:
            proposer = participant
            next_choice = proposer.preference_order_list[proposer.ask_rank]
            while next_choice not in set_proposed_to and proposer.ask_rank < proposer.cutoff:
                if self.proposal_accepted(proposer, next_choice):
                    if next_choice in set_proposed_to:
                        proposer = next_choice.just_rejected
                        proposer.ask_rank += 1
                        next_choice = proposer.preference_order_list[proposer.ask_rank]
                    else:
                        set_proposed_to.add(next_choice)
                else:
                    proposer.ask_rank += 1

def proposal_accepted(self, proposer, next_choice):
    current_next_choice_pair_id = next_choice.paired_with.id if next_choice.paired_with is not None
    for i in range(next_choice.cutoff):
        if next_choice.preference_order_list[i].id == proposer.id:
            if next_choice.paired_with is not None:
                next_choice.just_rejected = next_choice.paired_with
                next_choice.just_rejected.paired_with = None
            next_choice.paired_with = proposer
            proposer.paired_with = next_choice
            return True
        elif next_choice.preference_order_list[i].id == current_next_choice_pair_id:
            return False
```

```
return False
```

Above is our modified implementation of Irving's Stable Roommates algorithm. This code is based on the pseudocode presented in Irving's paper. We start by creating an empty set that will keep track of every individual that has been proposed to. Then, we iterate once through every participant in the simulation, assigning that participant as the proposer, and we consider the proposed, or the "next choice," the highest-ranking individual in the proposer's preference order that the proposer has not yet proposed to. We then begin a loop in which the proposer proposes to this next choice so long as the proposed has not been proposed to before (as in, they are not in the set we created above) and the proposed is among the genders our proposer is attracted to (they are before the demarcated truncation index for that participant).

We then must check if the proposal is accepted, and we do so by checking whether the proposed already has a tentative match, and if they do, we check whether or not the proposer is higher in the preference list of the proposed than that tentative match. If there is no tentative match and the proposer is a gender that the proposed is attracted to or if the proposer is of higher rank than the tentative match, the proposer is now tentatively matched with the proposed. Otherwise, our proposer gets a rejection.

In the case of a match, our new proposer is the previous tentative match of the proposed. In the case of a rejection, our original proposer iterates down their preference list and proposes again. The inner loop repeats until we reach a participant that has already been proposed to. Eventually, the entire algorithm terminates once we have done what was described for every participant in the simulation.

3 Experiment 1: Matching with All Truthful Agents

Table 1: Simulation Data of 100 Reps of All Truthful Agents, Random Sexuality

# Total	# M	# W	# NB	Sexuality Dist.	% Matched	Avg. Utility	% Stable
30	10	10	10	Random	18.47	5.01	48
60	20	20	20	Random	10.01	5.37	47
120	40	40	40	Random	5.18	5.55	50
300	100	100	100	Random	2.09	5.56	39
900	300	300	300	Random	0.72	5.71	50
900	450	225	225	Random	0.70	5.51	52
900	225	450	225	Random	0.71	5.66	52
900	225	225	450	Random	0.71	5.65	47
900	432	432	36	Random	0.71	5.68	48

Table 2: Simulation Data of 100 Reps of All Truthful Agents, Weighted Sexuality

# Total	# M	# W	# NB	Sexuality Dist.	% Matched	Avg. Utility	% Stable
30	10	10	10	Weighted	18.33	4.80	65
60	20	20	20	Weighted	10.12	5.15	67
120	40	40	40	Weighted	5.19	5.20	72
300	100	100	100	Weighted	2.11	5.43	72
900	300	300	300	Weighted	0.70	5.42	59
900	450	225	225	Weighted	0.72	5.58	73
900	225	450	225	Weighted	0.70	5.35	68
900	225	225	450	Weighted	0.70	5.35	73
900	432	432	36	Weighted	0.72	5.61	67

4 Experiment 1: Discussion

In this first set of experiments, all agents were truthful in their preference reports. We used a variety of participant numbers as well as gender distributions to test the matching algorithm, and in the first case, sexuality was uniformly distributed among all possible sexualities, whereas in the second case, attraction to all genders was weighted more heavily in the randomization than other sexualities. In both cases, an average utility of between 5 and 6 was maintained across all trials, with utility slightly increasing with number of participants. This was the case for both random and weighted sexuality. Similarly, the proportion of stable matches did not deviate much in each case, though we can see that the percentage of stable matches rose by a solid 10 to 20 percent on average when the sexuality distribution was changed from random to weighted.

The percent of participants who were matched is approximately the same across both sexuality distributions, with percent matched starting decently high, nearing 20%, for smaller groups and tapering off to be less than 1% for the largest groups. Therefore, we see that though changes in sexuality distribution did make a difference in the number of stable matchings, distribution of gender made no major difference, so long as the total number of participants remained the same, even in the last 432/432/36 split, which was meant to be more reflective of gender distributions in reality.

As for the trend of utility staying the same and percent matched going down as number of participants increased, the data suggests that while less matches were successfully made in larger groups, those matches occurred among participants high in each other's preference list, bringing in more total utility from fewer matches.

5 Experiment 2: Matching with All Strategic Agents

Table 3: Simulation Data of 100 Reps of All Strategic Agents, Random Sexuality

# Total	# M	# W	# NB	Sexuality Dist.	% Matched	Avg. Utility	% Stable
30	10	10	10	Random	29.07	6.54	24
60	20	20	20	Random	17.82	7.89	22
120	40	40	40	Random	9.56	8.44	18
300	100	100	100	Random	3.88	8.61	17
900	300	300	300	Random	1.34	8.63	15
900	450	225	225	Random	1.31	8.63	30
900	225	450	225	Random	1.31	8.80	16
900	225	225	450	Random	1.32	8.82	18
900	432	432	36	Random	1.30	8.76	16

Table 4: Simulation Data of 100 Reps of All Strategic Agents, Weighted Sexuality

# Total	# M	# W	# NB	Sexuality Dist.	% Matched	Avg. Utility	% Stable
30	10	10	10	Weighted	27.34	5.54	51
60	20	20	20	Weighted	17.18	6.95	44
120	40	40	40	Weighted	9.28	7.46	46
300	100	100	100	Weighted	3.93	7.88	54
900	300	300	300	Weighted	1.31	7.92	40
900	450	225	225	Weighted	1.32	8.04	55
900	225	450	225	Weighted	1.32	8.06	40
900	225	225	450	Weighted	1.32	8.12	45
900	432	432	36	Weighted	1.31	7.98	43

6 Experiment 2: Discussion

In this second set of experiments, all agents were strategic in their preference reports. We used the same variety of participant numbers and gender distributions as above, as well as the same two possible sexuality distributions.

We see when all agents play the simulation strategically, there is a significant increase in average utility when compared to the same attributes but with truthful agents. For lower numbers of participants, the improvement starts out small, at around 1-2 higher on average, but this improvement plateaus about 3-4 on average higher for the largest groups (900 participants). Therefore, unlike when all agents are truthful, average utility actually increases as participant number increases, though gender distribution again has little effect on this trend. However, we also see that average utility is about 1 higher in each category when sexuality is weighted over when sexuality is uniform random.

Like with the truthful pool, we also see a clear jump in percent stable matches between random and weighted sexualities. However, in this case, for uniform random sexualities, the percentage of stable matches appears to be significantly lower than the all-truthful pool, decreasing as the number of participants increases. When sexuality is weighted, however, percentage of stable matches increases by approximately 30%, becoming more comparable to the stability of all truthful agents with uniform random sexualities. So again, weighting sexualities brings about significantly more stable matchings than weighted.

Lastly, both cases using only strategic agents significantly improves the percent of participants matched. For smaller groups this difference is at most just over 10%, and as the number of participants increases, the difference shrinks, but unlike the entirely truthful cases, neither all-strategic case ever dips below 1% in our tests.

7 Experiment 3: Matching with 50% Strategic / 50% Truthful Agents

Table 5: Simulation Data of 100 Reps of 50% Truthful / 50% Strategic Agents, Random Sexuality

# Total	# M	# W	# NB	Sexuality Dist.	% Matched	Avg. Utility	% Stable
30	10	10	10	Random	29.13	6.95	27
60	20	20	20	Random	17.10	7.97	26
120	40	40	40	Random	9.58	8.89	14
300	100	100	100	Random	3.94	8.95	19
900	300	300	300	Random	1.35	9.31	22
900	450	225	225	Random	1.34	9.29	19
900	225	450	225	Random	1.33	9.17	25
900	225	225	450	Random	1.33	9.16	21
900	432	432	36	Random	1.33	9.49	21

Table 6: Simulation Data of 100 Reps of 50% Truthful / 50% Strategic Agents, Weighted Sexuality

# Total	# M	# W	# NB	Sexuality Dist.	% Matched	Avg. Utility	% Stable
30	10	10	10	Weighted	27.57	5.97	50
60	20	20	20	Weighted	16.27	7.04	46
120	40	40	40	Weighted	9.22	7.90	37
300	100	100	100	Weighted	3.99	8.62	55
900	300	300	300	Weighted	1.34	8.62	48
900	450	225	225	Weighted	1.35	8.82	45
900	225	450	225	Weighted	1.35	8.84	49
900	225	225	450	Weighted	1.34	8.73	52
900	432	432	36	Weighted	1.33	8.84	51

8 Experiment 3: Discussion

In our last set of experiments, there was a 50% chance that a participant was Strategic and a 50% chance that a participant was Truthful. Similarly as in experiments 1 and 2, we can see that the average utility increases as the number of participants increases, although not linearly with the number participants. Average utility appears to level off at a little above 9 when sexuality is purely randomized, and reaches about 8.8 when we weight sexuality towards "pansexuality" (the majority of participants are attracted to all genders).

We see that average utility per participant tends to be higher when we introduce weight the distribution of sexualities towards "pansexuality" over randomized sexualities. This is expected, as our weighting has the effect of creating more participants who are open to being matched with each other. Thus, in comparison to the randomized sexuality experiments, we expect that there will be fewer failed matches as a result of incompatible sexualities.
Of the experiments with 900 total participants, we notice that in both the randomized- and weighted-sexuality experiments, average utility tends to be the highest in the distribution of genders meant to simulate the real world ($M = 432, W = 432, NB = 36$). It is notable that this scenario is the closest to the setup of the original stable marriage problem, but differs still in the added complexity of sexualities.

9 Conclusion

Through our three sets of experiments, we investigated the impact on stability and average utility of strategic versus truthful agents, sexuality distribution, and total number of participants.
First, we found that on average individual utility was maximized when there was a mixture of truthful and strategic agents. We believe the strategic agents are successful for a similar reason as in the Boston Matching Problem, i.e. the strategic agents reduce crowd around the most attractive participants in the first rounds of the game, allowing for more, higher-ranking matches to be established. We also found that a group of all strategic players resulted in higher average utility than a group of all truthful players. These results indicate that our simulation's algorithm is not strategy-proof, as agents are made better off by being strategic.
Next, we found that using weighted sexuality distributions resulted in a higher percentage of our 100 repetitions to be stable. This was true across all experiments. We believe this is because in the weighted distribution, about half of the participants are made to be open to any partner in the dating pool. This results in fewer truncated preference lists, and therefore allows for more matches to be made. We believe the greater number of potential matches makes it easier for there to be stable matches as well.
Furthermore, we found that the total number of participants resulted in higher average utility per participant, but this increase was not linear in the number of participants. In particular, we found that across all experiments, the greatest unit step-up in average utility was between 30 total participants and 60 total participants. While more participants allow for more potential matches, after a certain threshold it appears that adding more participants results in diminishing returns.
Lastly, we noticed that the percentage of matched participants looks low, especially for high numbers of participants ($total = 900$). However, we believe these numbers are actually not problematic when compared to real life dating apps. For instance, Tinder's match rate is 1.63%, and men on Tinder have an even lower match rate of 0.5%[1].
In conclusion, we believe our algorithm is able to successfully create matches for a wider population of participants than the original stable marriage problem. Further directions of exploration would include testing different types of strategies that users can employ, as well as applying our algorithm to a simuulation more reflective of real life in terms of distributions of sexualities and genders.

[1] https://blog.gitnux.com/tinder-match-statistics/