Analysis of the Gale-Shapley Algorithm for the Stable Marriage Problem

AJ Bu, Kayla Gibson, Lucy Martinez, Natalya Ter-Saakov

Spring 2022

1 Introduction

Often in our lives, we are interested in matching people up with positions, objects, other people, etc while satisfying their preferences. In this project, we will explore a specific subset of this problem called the stable marriage problem. The idea behind this sub-problem is that you have two sets of entities (men and women, colleges and students, etc), each of which have a ranking of the other set. The stable marriage problem starts with a set of men and of women and attempts to couple them up in a way that no person would be happier with another spouse. More generally, it looks at two disjoint, equally sized sets and tries to find a stable matching. For the sake of this project, we will think of this problem as n men pairing with n women.

Definition 1. *Given two sets say A, B, a* ***matching*** *is a set of pairs P_i, each pair containing one element from A and one element from B, (a, b) such that every element is in exactly one pair.*

Definition 2. *A matching is* ***stable*** *if there is no unmatched pair $\{a, b\}$ such that both a and b prefer each other to their present partners.*

The stable marriage problem can be modified to other types of matching problems, such as college admissions. In the college version of the problem, every student has a ranking of colleges (including some that they would not attend), every college has a ranking of students (including some they would not accept) and also a quota of how many students they can support. The goal is to match every student to a college such that no college accepts more students than its quota and there is no unmatched pair student, college such that the student prefers that college to the one they are currently in and the college prefers that student to one they have currently admitted.

The question of whether or not a stable marriage exists between two sets is answered by D. Gale and L. S. Shapley [1]. As long as the sets are of equal size and there are no pairs that are not willing to partner under any circumstances, at least one stable matching exists.

In this project, in particular, we would like to investigate the quantities of "average happiness of men" and "average happiness of women" for each of the $n!^{2n}$ possible choices of rankings for the stable matching problem. We calculate the "average happiness of men" and the "average happiness of women" by diving the sum of the rankings that each man (respectively, woman) gave to the woman (respectively, man) that they were matched with by the number of men (respectively, women). Note that the Gale-Shapley algorithm (as written above) is male-optimal, i.e., no man can do any better than he does in the matching found by the algorithm.

We would also like to estimate the expectation, variance, and higher moments of these quantities. We will also write code for the extension of this algorithm into a college admissions problem.

2 Background

In this section, we present the algorithms Gale and Shapley presented in their paper *College Admission and the Stability of Marriage*[1].

Gale-Shapley Algorithm

At the start of the algorithm, each person starts with no people cancelled from his or her list. People will be cancelled from lists as the algorithm executes.
For each man m, do propose(m) as defined below:

- propose(m):
  Let W be the first uncancelled woman on m's preference list. Do: reject(W,m), as defined below.

- reject(W,m):
  Let m' be W's current partner (if any).
  If W prefers m' to m, then she rejects m, in which case:

  1. Cancel m off W's and W off m's list;
  2. Do: propose(m). (Now m must propose to someone else).

  Otherwise, W accepts m as her new partner, in which case:

  1. Cancel m' off W's list and W off m''s;
  2. Do: propose(m'). (Now m' must propose to someone else).

We provide an example of how the algorithm works. Denote the lower-case letters to be the men, and the upper-case letters be the women. Suppose the preference lists are as follows:

$$\begin{array}{ll} a : ABC & A : bac \\ b : BAC & B : cba \\ c : ACB & C : acb \end{array}$$

First, do propose(a). a's top choice is A, so we do reject(A, a). Since A has no current partner, A accepts a. Next, do propose(b). b proposes to his first choice B who accepts since B has no current partner. Finally, do propose(c). Since c's first choice is A, do reject(A, c). A is currently married to a, and A prefers a over c, then A rejects c. We cross A off of c's preference list, and cross c off of A's preference list. Do propose(c) again in which case c proposes to C, and C accepts. Thus, we have found the matching $\{aA, bB, cC\}$. Moreover, this matching is stable. Observe that a and b have their top choices, hence both a and b do not prefer anyone else. While c has his second choice, A (c's first choice) will not prefer c as he is last in A's preference list. Therefore, $\{aA, bB, cC\}$ is a stable matching.

The Gale-Shapley algorithm can also be extended to work on college admissions. The college admissions algorithm is described below. Given N students and M colleges, we need a preference list of colleges for each student (this list does not need to include all M colleges). Similarly, each college has a preference list of students, leaving off students the college would never accept. We also need a quota list which gives the quota that each college would like to meet of accepted students (each college can not accept more students than its quota).

To simplify things, if a college will never accept a student, then that student will not be permitted to apply.

Gale-Shapley Algorithm for College Admissions

- applications:
  Let A be the set of preference lists for the students. Each student tries to apply to their first choice. If they are not allowed to apply to their first choice, we move through the preference list until we find a college they may apply to.

- rejections:
  The college goes through the applications and puts their top preferences on a waitlist (as many as their quota will allow) and rejects all others.

- re-applications:
  Those who have been rejected must reapply, again to their first choice (if they are allowed to).

- rejections:
  Colleges look at their new applications along with their current waitlist, and update their waitlist to include their preferred students (as many as their quota will allow). This may mean bumping students off of the waitlist.

- stopping point:
  This process stops when either every student is on a waitlist, or those students not on a waitlist have been rejected from every college on their preference list.

Unlike the stable matching problem, the college admissions problem will not always be able to assign every student to a college.

3 Maple Procedures

Accompanying this paper is the Maple package `FindingMatchings.txt`, written by AJ Bu, Kayla Gibson, Lucy Martinez, Natalya Ter-Saakov, and Doron Zeilberger. Many of the procedures involve rankings, for example when M is n men's rankings of n women. This is formatted as a list containing n lists, where the i-th list represents the way the $i-$th man ranks the n women. For example, when $n = 3$ we may have

$$M = [[1, 2, 3], [1, 3, 2], [3, 1, 2]].$$

This tells us that man 1's first choice is woman 1, then woman 2, then woman 3; man 2's first choice is woman 1, then woman 3, then woman 2; man 3's first choice is woman 2, then woman 3, then woman 1.

Running the procedure `FindHappiness(M,W,matching)`, where `M` is a list of each man's ranking of the women, `W` is a list of each woman's ranking of the men, and `matching` is a given matching of the men and women, outputs the average happiness of the men and the average happiness of the women. This procedure is used to calculate happiness in our other procedures.

The program `EstHappMenProp(n,K)` (respectively, `EstHappWomenProp(n,K)`) generates K examples of rankings for n men and n women, where K is a large integer, and estimates the average happiness of men and women when men (respectively, women) propose by finding their average happiness over the K trials of the Gale-Shapley algorithm.

To find the exact average happiness, we want to look at the average happiness over all possible rankings. The procedure `AllRank(n)` generates all the ways that n men (or women) can rank n women (or, respectively, men). In other words, it creates a list of all the ways of choosing – with repetition – n permutations of $[n]$. The procedure `AvgHapMenPropOverAllRank(n)` (respectively, `AvgHapWomenPropOverAllRank(n)`) inputs an integer n and calculates, over all possible rankings of n men and n women, the average happiness of the men and the average happiness of the women when men (respectively, women) are proposing in the Gale-Shapley algorithm.

We also want to look at the average happiness over all stable matchings for a given ranking. The procedure `EstHappAllMatch(n,K)` generates K examples of rankings for n men and n women, where K is a large integer, and estimates the average happiness of men and women over all stable matchings for each ranking. To calculate the exact average happiness, we first have the procedure `AllMatchings(M,W)`, which inputs the ranking table M of n men's rankings n women and the ranking table W of n women's rankings n men. It then finds all stable matchings by going through all permutations of $[n]$ and testing which ones are stable. `AvgHapOverAllMatch(M,W)` finds the average happiness of the men and the average happiness of the women over all of the stable matchings for the given rankings.

The extension to college admissions is given by the procedure `GaleShCollegeAdmissions(A,C,Q)`. Given A (the set of N student preference lists), C (the set of M college preference lists), and Q (the list of quotas for the M colleges), `GaleShCollegeAdmissions(A,C,Q)` outputs the list of sets of accepted students at each college. This is the verbose version which prints out what is happening during the algorithmic proccess. There is also a terse version `TerseGaleShCollegeAdmissions(A,C,Q)`.

4 Results

Given a ranking table for n men and n women, the average happiness of the men (and women) over all matchings – both stable and unstable – is

$$\frac{1}{n}\sum_{i=1}^{n} i = \frac{n+1}{2}.$$

Consequently, we are only interested in looking at the stable matchings. Note also that due to the symmetry of the problem, the average (over all rankings) happiness of men when they propose is equal to the average happiness of women when they propose. Similarly, the average happiness of women when men propose is equal to that of men when the women propose. As such, we only ran the procedures where the men propose.

Using the procedure `AvgHapMenPropOverAllRank`, we find that when 2 men propose to 2 women using the Gale-Shapley algorithm, the average happiness of men over all ranks is $\frac{5}{4}$; when 3 men propose to 3 women, the average happiness of men is $\frac{35}{24}$. Moreover, when using `EstHappMenProp(n,K)`, we estimate the average happiness when there are $n = 4, \ldots, 8$ men and women by taking the average happiness of men over K rankings. Running

```
seq(evalf(EstHappMenProp(n,100000))[1],n=4..8)
```

output

```
1.636525000, 1.796426000, 1.938558333, 2.065690000, 2.186871250.
```

If we look at $\prod_{k=2}^{n} \frac{2k+1}{2k}$ for $n = 4, \ldots, 8$, we get approximately

```
1.640625000, 1.804687500, 1.955078125, 2.094726562, 2.225646973.
```

From this data, in the case where n men propose to n women using the Gale-Shapley algorithm, we might want to conjecture that the average happiness of men over all possible rankings is around

$$\prod_{k=2}^{n} \frac{2k+1}{2k},$$

but this does not hold for larger n.

When 2 men propose to 2 women using the Gale-Shapley algorithm, the average happiness of women over all ranks is $\frac{11}{8}$; when 3 men propose to 3 women, the average happiness of women is $\frac{371}{216}$. Running

```
seq(evalf(EstHappMenProp(n,100000))[2],n=4..8)
```

output

```
2.040075000,2.340290000, 2.625396667, 2.904457143,3.173947500.
```

Using `AvgHapOverAllRankMatch`, we find that the average happiness of n men and n women over all stable matchings of all possible rankings is $\frac{21}{16}$ when $n = 2$ and $\frac{2315}{1458}$ when $n = 3$. For larger n, we can estimate the average happiness by using the procedure `EstHappAllMatch`. For example, running

```
seq(evalf(EstHappAllMatch(n,100000)),n=4..6)
```

outputs

```
[1.833473792, 1.837559000],[2.063673493, 2.063224019], [2.278836056, 2.278063265].
```

5 Conclusion

References

[1] D. Gale and L. S. Shapley. *College Admissions and the Stability of Marriage. The American Mathematical Monthly*, Vol. 69, No. 1 (Jan., 1962), pp. 9-15.