ECN 820 Research Project

GALE-SHAPLEY TWO-SIDED MATCHING:
CHARACTERISTICS AND TRANSFORMATIONS

Keith Qu

ECN 820 Research Project Supervisor: Professor Tsogbadral Galaabaatar
ECN 820 Research Project Second Reader: Professor Min Seong Kim

The Research Paper is submitted
In partial fulfillment of the requirements for the
Bachelor of Commerce degree
in
Business Management, Economics and Management Science
Ryerson University
Toronto, Ontario, Canada

Author’s Declaration Page

I hereby declare that I am the sole author of this Research Paper.

I authorize Ryerson University to lend this Research Paper to other institutions or individuals for the purpose of scholarly research.

_________________________________ _April 28, 2016_________________

Signature Date

I further authorize Ryerson University to reproduce this Research Paper by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

__________________________________ _April 28, 2016_________________

Signature Date

Gale-Shapley two sided matching: characteristics and transformations

A Research Paper presented to Ryerson University in partial fulfillment of the requirement for the degree of Bachelor of Commerce in Business Management

By Keith Qu

ABSTRACT

This paper considers some applications of Gale-Shapley two-sided matching games with modifications. It looks at the effects of outside influence on one of the participants, and the resultant changes to the final pairings. In addition, games in which there is a single optimal solution for both sides will be examined. The proportion of such games for any given game size will be estimated with a Monte Carlo algorithm.

Acknowledgements

My advisor, Prof. Bagi, was helpful on every step of the way.

Prof. Kim provided some great notes that helped me clarify several points.

I am grateful to Prof. Maurice Roche for allowing me to be the first student in the business economics program to write an undergraduate economics thesis.

And of course, thanks to Dr. Johnson for reminding us that puns are the lowest form of humour.

Dedication

For my grandparents.

Table of Contents

Tables and Figures

1. Introduction

Gale and Shapley (1962) first proposed certain conditions under which a two-sided matching game will always have at least one stable outcome. Such games can be used for college acceptances, school districting, and matching hospitals with resident medical students. The simplest form of this game is known as the Stable Marriage Problem, where men are matched with women to form stable marriages. A deferred-acceptance model is used to find matchings that are “stable”, at which point the selected matchings cannot be improved by any blocking pair.

While stability is assured, these solutions are optimal only for the side making the proposals. Gale and Shapley, corroborated by Roth and Sotomayor (1992), demonstrated that the deferred-acceptance algorithm finds matchings that are optimal for the proposing side. There are, however, games for which the male-proposal and female-proposal algorithms arrive at the same set of matchings. This will be a solution that is optimal for both parties.

We will examine the existence of these generally-optimal matchings, and develop a computational method to determine whether a matrix representing a given set of preferences has such a stable set. It will be shown that when this type of stable matching exists, it is the unique solution for that game. A Monte Carlo simulation will be used to determine the percentage of possible games with unique optimal solutions for any given matrix size. Additionally, we will consider games in which one player is being influenced by a third party, such as a parent, who is not involved in the standard game. Some mechanisms for breaking possible ties will also be explored.

2. Review of literature, notation, and methods

2.1. The Stable Marriage Problem

2.1.1. Setup and algorithm.

A deferred-acceptance procedure is used to match men and women, somewhat similar to a college waitlist. Men propose based on their preferences, and females accept or reject them based on theirs. We assume that preferences are strict and unchanging, men can propose to a woman only once, and all men and women prefer marriage to being single. There are 2n participants: n males and n females. The Gale-Shapley matching algorithm:

1. Men make proposals to their first choice.

2. If a woman receives only one proposal, she pre-accepts for now. If she receives more than one, she becomes engaged to her favoured choice among current suitors and rejects the others.

3. Men who have been rejected make proposals to their next-highest choice.

4. Women make the same decision as in step 2, and she will break off an engagement from step 2 if a better suitor comes along.

5. This continues until all couples are engaged, at which point they get married.

This procedure guarantees that each woman receives at least one proposal, so when there are equal numbers of men and women, it will never fail to pair up all the men and women. We know that there are no possible blocking pairs, since even if man m prefers another w' to his current betrothed w, he had already been rejected by her. Similarly, w prefers no potential suitor more than m, and even if she prefers m' to m, m' will never propose if he is not rejected by a higher-ranked match. Therefore, the matching from this algorithm is stable since for every pair, no one member can form a blocking pair with anybody else. If women propose, the algorithm remains the same but with the genders reversed.

2.1.2. Notation.

The problem can be restated in mathematical notation, following Adachi (2000). For simplicity we begin with sets of *n* males and *n* females[1], where $M = (m_1, m_2, \dots, m_n)$ and $W = (w_1, w_2, \dots, w_n)$. Each has an ordered list of preferences over the opposite sex, with the operator $>_m$ representing a strict male preference.

e.g. $w >_m w'$: male strictly prefers *w* to *w'*,

$w \geq_m w'$: male weakly prefers *w* to *w'*; *w* is at least as good as *w'*,

$w \sim_m w'$: indifference between *w* and *w'*.

The same applies for female preferences represented by $>_w$.

Matchings are represented by μ, where $\mu(m) \in W, \mu(w) \in M,$ when everyone is matched with someone of the opposite sex, and $\mu(m) = m$, $\mu(w) = w$ when "matching" someone with themselves – being single – is a better option than marriage. The expression $\mu(m) >_m w$ means that a male prefers his current match to some other *w*.

Assumption 1. Strict preferences.

$w >_m w'$, or $w' >_m w$,

[1] With different numbers of each sex and everyone prefers marriage to being single, the sex with fewer members will have members not part of any stable matching.

$m >_f m'$, or $m' >_m m$, for all $w \neq w', m \neq m'$.

That is to say, there are no potential ties. The matching algorithm is not guaranteed to work when this condition is relaxed, and in section 5 we will explore ways in which non-strict or more complex preferences can be converted into strict ones.

Assumption 2. All individuals prefer marriage to remaining single.

$f >_m m, \forall f$

$m >_f w, \forall m$

For the purposes of this paper, we make this assumption to guarantee *n* pairs of stable matchings where everyone is matched with someone of the opposite sex. It is not a necessary assumption for Gale-Shapley, since it is possible in a stable solution for some to remain single.

Ordered preferences may be represented as:

$P(m_1) = (w_1, w_2 \sim w_3, m_1)$, $\qquad$ $P(w_1) = (m_2, m_3, m_1, w_1)$,

Where m_1 is indifferent between f_2, f_3 but prefers f_1 over either, and having himself in the last rank means that he prefers any wife to remaining single. $p(m_1, w_1) = (1,3)$ denotes m_1's first-ranked preference for w_1, and w_1's third-ranked preference for m_1.

2.1.3 Examples and Implications.

In a one-stage male-proposal game, a stable pairing will be immediately noticeable when each male strictly prefers his first choice to any other females, and no two males have the same first choice. This is readily apparent in the first example provided by Gale/Shapley (1962), where cell has values $p(m_i, w_i)$.

	$\boldsymbol{w_1}$	$\boldsymbol{w_2}$	$\boldsymbol{w_3}$
m_1	1,3	2,2	3,1
m_2	3,1	1,3	2,2
m_3	2,2	3,1	1,3

Here, the men propose to their first choices: m_1 proposes to w_1, m_2 proposes to w_2, and m_3 to w_3. Each woman has only received one proposal, so they are all engaged immediately. None of the men have been rejected, so there is no need for a second round of proposals. Despite the fact that all females are now engaged to their last choices, the matching game is over, and the matches are stable.

We can verify this stability by checking for potential incentives to deviate. w_2 prefers m_1 and m_3 to m_2, but m_1 and m_3 prefer their respective pre-match engagements over w_2. In this case w_2 cannot improve her situation, and the same is true for w_1 and f_3.

If the roles had been reversed, with the females making the proposals, then by the same method we obtain the stable pairings $V^T = \begin{bmatrix} m_1 & m_2 & m_3 \\ w_3 & w_1 & w_2 \end{bmatrix}$. Additionally, we can see that there are stable matchings that exist outside of the algorithm and are optimal for neither side, such as an all second-choice equilibrium of $V^T = \begin{bmatrix} m_1 & m_2 & m_3 \\ w_2 & w_3 & w_1 \end{bmatrix}$. However, these will not always be so readily apparent, especially in games where *n* is large.

The stable matchings will not always be found immediately in the first round. Consider a male-proposal game, where w_1 has become more desirable to m_3 compared to the above example.

	$\boldsymbol{w_1}$	$\boldsymbol{w_2}$	$\boldsymbol{w_3}$
m_1	1,3	2,2	3,1
m_2	3,1	1,3	2,2
m_3	1,2	3,1	2,3

The following table represents, for each stage of deferred acceptance, the proposals of males for each female, along with the p value for that pairing in parentheses.

Stage	$\boldsymbol{w_1}$	$\boldsymbol{w_2}$	$\boldsymbol{w_3}$
1	$m_1(1,3)$, $m_3(1,2)$	$m_2(1,3)$	
2	$m_3(1,2)$	$m_1(2,2)$, $m_2(1,3)$	
3	$m_3(1,2)$	$m_1(2,2)$	$m_2(2,2)$

We can see that when a woman receives two proposals, she brushes the lower-ranked one aside, and that man then makes a next-best proposal. First, w_1 rejects m_1, and m_1 proposes to w_2, who then rejects m_2. Only m_3 got his first choice w_1, who never even had any contact with m_2, her first choice. If the women made proposals, based on the first example we can expect the stable matching $V^T = \begin{bmatrix} m_1 & m_2 & m_3 \\ w_3 & w_1 & w_2 \end{bmatrix}$.

In the previous example, stable matchings were achieved despite the existence of some non-strict preferences. To illustrate the importance of strict preferences, consider a similar game that results in a tie.

	$\boldsymbol{w_1}$	$\boldsymbol{w_2}$	$\boldsymbol{w_3}$
m_1	1,2	1,2	2,1
m_2	3,1	1,3	2,2
m_3	1,2	3,1	2,3

At some point, the deferred acceptance algorithm will reach an impasse. Here, m_1 is proposing to both of his favourites, who will both not receive a better proposal under the standard rules. In a sense, m_1 is tied with himself and with m_3.

Stage	$\boldsymbol{w_1}$	$\boldsymbol{w_2}$	$\boldsymbol{w_3}$
1	$m_1(1,2)$, $m_3(1,2)$	$m_1(1,2)$, $m_2(1,3)$	
2	$m_1(1,2)$, $m_3(1,2)$	$m_1(1,2)$	$m_2(2,2)$

While non-strict preferences will not *always* get stuck in such a way[2], strict preferences are the only way to guarantee a stable outcome. In these situations, breaking the ties will guarantee stable matchings. Any tie breaking method will convert non-strict preferences into effectively strict preferences, and this will be expanded further in section 5.

2.2. The Monte Carlo method

We are interested in making calculations involving the percentage of unique single-solution games for a given matrix size. Even without any calculations, it should be apparent that the number of possible permutations of a strategic form matrix game grows very rapidly. Since even the most powerful computers have limitations, to make estimations within a finite amount of time, we will use a Monte Carlo algorithm.

As we will show in 3.2, the number of possible games grows very quickly. Rather than using brute force to comb through every single possible game permutation for *n* men and *n* women, we will instead use a relatively smaller number of random permutations. By generating

[2] In this example, if only w_1 were to revert to strict preferences and rank m_3 3rd, eventually m_3 will propose to w_2 and knock out m_1's duplicate proposal.

games with values that are uniformly distributed over all possible values, we can reasonably estimate the proportion of unique solution games for a given n.

3. Games with equivalent male and female optima

3.1. Unique optimality

We have shown some examples in section 2 where the stable marriage game is optimal only for the group that takes the initiative. We are also interested in fairness, and it seems strange for anyone to marry the first person who comes along and makes an unchallenged proposal. Some games will have a solution that is optimal for both men and women but unfortunately, within the rules of the game, that will never be the case when the male-proposal and female-proposal methods have already arrived at different stable matchings.

3.1.1. **Theorem**. If for a given combination of male and female preference sets, the male-optimal and female-optimal matchings are equivalent, then that matching set is the unique solution for that combination of preferences.

3.1.2. **Proof**. Gale (1962) and Roth (1992) show that the stable matchings induced by the male-proposal and female-proposal algorithms are optimal for the proposing side, and that matchings that are more advantageous for one side are less advantageous for the other. Suppose in a game with a unique solution there exists some other stable matching not obtained through the male- and female-proposal deferred-acceptance algorithms. This set is necessarily less beneficial for either males or females than the current stable set. But if it was less beneficial for one group, it would have been more beneficial for the other. If that were the case, one of the proposal algorithms would have arrived at that solution instead of the current one. Since this cannot be the case, the current solution is unique for this particular game.

3.2. The Monte Carlo algorithm

With strict preferences and equal men and women (n), there are $(n!)^{2n}$ possible preference combinations, represented by n x n matrices. This is not difficult to prove. Each row has n! permutations, so an n row matrix with n! permutations will have $(n!)^{n}$ unique results. Since there are two groups with the same number of permutations, we square this to obtain $(n!)^{2n}$. Even when n is relatively small, it is clear that the number of permutations balloons out of control quickly. There are only 16 possible combinations when $n = 2$ but 46,656 when $n = 3$. A 4x4 matrix contains over 110 billion unique games, and so on.

When calculating the number of game with unique stable matchings, the number of possible game permutations quickly balloons to levels that make it impractical to check every possible game. For our purposes, it will be useful to instead use a Monte Carlo simulation to make estimations. While the number of permutations skyrockets beyond even the double floating point range after a size 14 square matrix, this method can give useful results on the proportion of unique optimal games for significantly larger matrices.

1. Generate random vectors of strict preferences for each man and woman. It is enough for each value to be unique, and simpler to use a random permutation of integers $\iota \in [1, n]$ for a vector of size n.

2. Use the deferred-acceptance algorithm for both male-proposal and female-proposal games

3. Arrange the matchings into two matrices with rows $(m_i, \mu(m_i) = w_i)$, such that $m_i, w_i \in V(m, f)$. If these two matrices are equal, then the game has a unique solution that is both male- and female-optimal.

For each *n*, this Monte-Carlo algorithm should be run *m* times. Since the resulting random preferences are uniformly distributed over all possible preferences, a suitable estimate can be made even when *n* is many orders of magnitude smaller than $(n!)^{2n}$, such that it would take a less than infinite amount of time to calculate.

3.3. Results

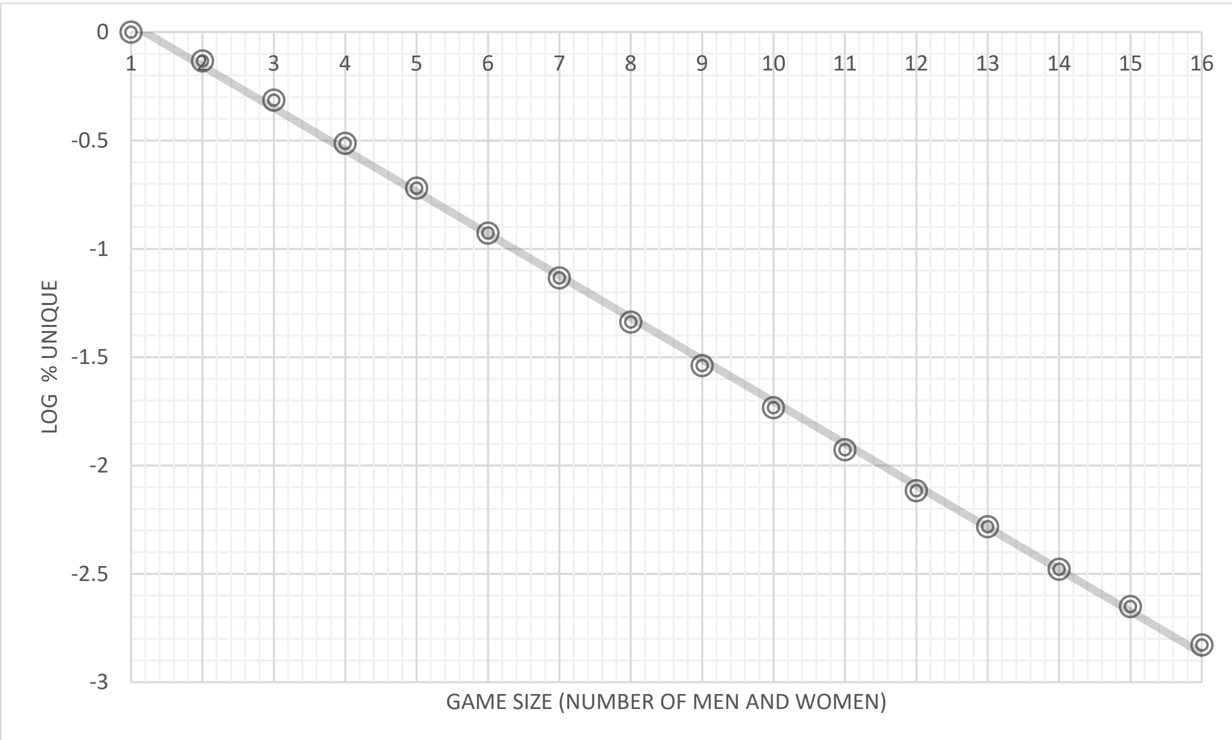
Using $m = 10{,}000{,}000$ iterations, we find the results for game sizes up to $n = 16$, where everyone prefers marriage to being single. At $n = 1$, all possible games are optimal for both men and women, since they both only have one choice. This number dwindles quickly, until at $n = 16$ fewer than 6% of possible games have such an optimal. From table 1, we can see that the relationship seems to be exponential, so we use the logarithm of the percentage to construct figure 1, which suggests a log-linear relationship.

Table 1. Percentage and log-Percentage of unique solution games.

n	%	log %
1	100.00000%	0
2	87.54200%	-0.13305
3	73.03253%	-0.31427
4	59.83707%	-0.51354
5	48.67863%	-0.71993
6	39.54897%	-0.92763
7	32.16391%	-1.13433
8	26.24356%	-1.33775
9	21.46705%	-1.53865
10	17.67890%	-1.7328
11	14.55329%	-1.92735
12	12.05030%	-2.11608
13	10.10600%	-2.29204
14	8.38420%	-2.47882
15	7.05930%	-2.65082
16	5.91249%	-2.8281

The proportion of games with unique stable conditions declines, but the amount of total permutations rises faster by many orders of magnitude, so that when there are only 4 men and 4 women, there are over 65 *billion* preference combinations with only one stable set of matchings. With 12 of each gender there are 1.8×10^{207} games with unique solutions.[3]

Figure 1. Log % of unique games as n increases



3.4. Testing a log-linear percentage model

While the Monte Carlo method is time-saving relative to brute force, running a large number of iterations on increasing *n* still takes significant amount of computing time. From figure 1, we can see an apparent log-linear relationship between the percentage of unique solution games and the size of the matrix. We can try to find an alternative to the computational method using OLS on the log linear model $log\% = \beta_0 + \beta_1 n + \mu$.

---

[3] By comparison, there are only about 10^{81} atoms in the known universe.

Table 2. log-linear model of percentage of single solution games, with |t-value|.	
β_0	0.231391 (16.68)
β_1	-0.193731 (135.01)
$\bar{R}^2$	0.9992
F-statistic	18230

From table 1, the high $\bar{R}^2$ score suggests that this is a good prediction model, but this assumption does not stand up to scrutiny. For example, at $n = 20$, the estimation model predicts that 0.026167711 of games will have a single solution, while a Monte Carlo simulation with ten million iterations gives an estimate of 0.0299901. A difference of less than 0.004 appears to be minor, but when we consider that a size 20 game has $2.79\text{x}10^{735}$ possible variations, this small deviation results in a difference of $1.06\text{x}10^{733}$. Similarly, for $n = 21$, the regression and Monte Carlo methods yield 0.021559041 and 0.0257641, respectively. Due to the very large number of possible games, an $\bar{R}^2$ of 0.9992 is actually very low, and for estimation purposes it would be more useful to continue using the Monte Carlo algorithm.

3.5. A large game with unique solutions.

An added benefit of the algorithm is that aside from the above tests, we can also use it to quickly discover large unique solution games, such as this 20 player game:

$$\begin{array}{c} \\ m_1 \\ m_2 \\ m_3 \\ m_4 \\ m_5 \\ m_6 \\ m_7 \\ m_8 \\ m_9 \\ m_{10} \end{array} \begin{array}{c} \begin{array}{cccccccccc} w_1 & w_2 & w_3 & w_4 & w_5 & w_6 & w_7 & w_8 & w_9 & w_{10} \end{array} \\ \begin{bmatrix} 1,6 & 8,8 & 2,6 & 6,7 & 4,8 & 10,3 & 9,9 & 7,3 & 5,10 & 3,5 \\ 6,7 & 3,3 & 9,5 & 2,4 & 8,10 & 1,9 & 5,6 & 10,9 & 7,9 & 4,6 \\ 7,8 & 4,1 & 8,9 & 5,2 & 10,4 & 3,6 & 6,8 & 9,1 & 2,3 & 1,7 \\ 5,4 & 4,2 & 10,8 & 6,10 & 8,5 & 2,7 & 7,1 & 9,8 & 1,1 & 3,8 \\ 5,2 & 7,10 & 3,4 & 8,8 & 9,7 & 4,5 & 1,5 & 2,2 & 10,8 & 6,9 \\ 1,9 & 10,9 & 3,1 & 4,6 & 2,6 & 7,4 & 9,2 & 5,6 & 8,5 & 6,2 \\ 4,1 & 6,4 & 8,10 & 9,3 & 7,2 & 5,10 & 10,7 & 1,7 & 3,6 & 2,10 \\ 2,3 & 4,7 & 10,2 & 9,1 & 8,1 & 1,8 & 3,4 & 6,4 & 7,2 & 5,3 \\ 3,10 & 4,5 & 9,3 & 10,5 & 7,3 & 2,1 & 6,5 & 1,5 & 5,4 & 8,1 \\ 7,5 & 4,6 & 10,7 & 6,9 & 1,9 & 9,2 & 5,10 & 3,10 & 8,7 & 2,4 \end{bmatrix} \end{array}$$

We can continually generate random games and use the deferred acceptance algorithm for both male and female proposals, until a game is found in which both the male and female optimal solutions are equivalent. For this game, the unique solution is optimal for both *m* and *w*:

$$V^T = \begin{bmatrix} m_1 & m_2 & m_3 & m_4 & m_5 & m_6 & m_7 & m_8 & m_9 & m_{10} \\ w_3 & w_4 & w_2 & w_9 & w_8 & w_5 & w_1 & w_7 & w_6 & w_{10} \end{bmatrix}$$

4. Stability in influenced games

So far we have shown the resulting matchings when all parties have only one effective list of preferences. We raise the question: what if one of the parties is influenced by a third party? Let us consider the reasonable possibility that when seeking a wife, one of the men might be influenced by his parents.

4.1. Man defers to parental preferences

	$\boldsymbol{w_1}$	$\boldsymbol{w_2}$	$\boldsymbol{w_3}$	$\boldsymbol{w_4}$
m_1	1,2	2,3	3,3	4,1
m_1'	4,2	3,3	2,3	1,1
m_2	2,1	1,2	3,1	4,3
m_3	4,3	2,1	1,2	3,2
m_4	3,4	1,4	2,4	4,4

In this female-proposal game, male preferences m_1 and parental preferences m_1' are different, while the various women keep preferences over m_1 himself. We can see, for example, that m_1 ranks w_1 first, but m_1' ranks her last; on the other hand, w_1 is really only ranking the man, not the parents. We can see that when m_1''s influence overpowers m_1 completely, he may be matched with a less desirable *w*, while conversely *w* will prefer him more than the match from his true preferences. At best, his match will not change.

Results from m_1's true preferences in a female proposal game:

Round	$\boldsymbol{m_1}$	$\boldsymbol{m_2}$	$\boldsymbol{m_3}$	$\boldsymbol{m_4}$
1	w_4(4,1)	w_1(1,2), w_3(3,1)	w_2(2,1)	
2	w_4(4,1)	w_1(2,1)	w_3(1,2), w_2(2,1)	
3	w_4(4,1)	w_1(2,1), w_2(1,2)	w_3(1,2)	
4	w_1(1,2), w_4(4,1)	w_2 (1,2)	w_3(1,2)	
5	w_1(1,2)	w_2 (1,2)	w_3(1,2)	w_4(4,4)

$$V^T = \begin{bmatrix} m_1 & m_2 & m_3 & m_4 \\ w_1 & w_2 & w_3 & w_4 \end{bmatrix}$$

From m_1''s preferences:

Round	$\boldsymbol{m_1'}$	$\boldsymbol{m_2}$	$\boldsymbol{m_3}$	$\boldsymbol{m_4}$
1	w_4(1,1)	w_1(2,1)	w_2(2,1)	
2	w_4(1,1)	w_1(2,1)	w_2(2,1), w_3(1,2)	
3	w_4(1,1)	w_1(2,1), w_2(1,2)	w_3(1,2)	
4	w_4(1,1)	w_2(1,2)	w_3(1,2)	w_1(3,4)

This has actually slightly improved the situation for m_4, who now has his third choice, even though his new wife also ranked him last. While $\mu(m_1')$ is a great match for m_1', it is actually the worst possible match for m_1, since $p = (4,1)$. At least his new wife really likes him.

If m_1 and $m_1'$are both in play, they can each be a member of blocking pairs. In this example, their preferences are the exact reverse of each other, and preferences for m_1 include both (m_1, m_1').

Round	$(\boldsymbol{m_1}, \boldsymbol{m_1'})$	$\boldsymbol{m_2}$	$\boldsymbol{m_3}$	$\boldsymbol{m_4}$
1	w_4((4,1),1)	w_1(2,1), w_3(3,1)	w_2(2,1)	
2	w_4((4,1),1)	w_1(2,1)	w_2(2,1), w_3(1,2)	
3	w_4((4,1),1)	w_1(2,1), w_2(1,2)	w_3(1,2)	
4	w_1((1,4),2) w_4((4,1),1)	w_2(1,2)	w_3(1,2)	

At this point, the choice is between w_1, w_4 for (m_1, m_1') is not very clear. w_1 is m_1's favourite and m_1''s least favourite, and the opposite is true for w_4. The first three rounds play out the same way, but while in the two original games, m_1 could make a definitive choice between w_1 and w_4, the addition of another set of preferences has made this a trickier proposition. w_2 and w_3 are now matched with their first choices, and they are not affected by the conflict between m_1 and m_1'. m_4 prefers w_1 to w_4, but w_4 ranks m_1 first. In fact, (m_1', w_4) is the best matching in the entire game. So this is a situation where w_4 favours m_1, and m_1' favours w_4, but m_1 himself does not care for her at all. The only options are for either w_1 or w_4 to move over to m_4. Since m4 prefers $w_1$to w_4, (m_4, w_4) is the top candidate for a tie-breaking match.

This matching is the same as if only m_1''s preferences were considered, which favours the females. However, we still need to check whether it is stable. It is immediately apparent that (m_2, w_2) and (m_3, w_3) will not be blocked by any pair since no other males rank those two women as highly. The (m_4, w_1) matching is the man's third choice and woman's last and was instituted only as a tiebreaker for m_1's indecision. Once the tie is broken, the resulting solution is stable.

Round	$(\boldsymbol{m_1}, \boldsymbol{m_1'})$	$\boldsymbol{m_2}$	$\boldsymbol{m_3}$	$\boldsymbol{m_4}$
4	w_1((1,4),2) w_4((4,1),1)	w_2(1,2)	w_3(1,2)	
5	w_4((4,1),1)	w_2(1,2)	w_3(1,2)	w_1(3,4)

4.2. Influence-induced stability.

It is also possible for a completely new solution to emerge. Consider the following game where m_1 is influenced by m_1':

	$\boldsymbol{w_1}$	$\boldsymbol{w_2}$	$\boldsymbol{w_3}$
m_1	1,1	2,1	3,1
m_1'	3,1	2,1	1,1
m_2	1,2	2,2	3,2
m_3	1,3	2,3	3,3

We first find the solutions for m_1 and m_1' separately using both male and female proposal games. The deferred acceptance solution for the male proposal game with m_1's preferences:

Stage	$\boldsymbol{w_1}$	$\boldsymbol{w_2}$	$\boldsymbol{w_3}$
1	m_1(1,1)		
2	m_1(1,1)	m_2(2,2)	
3	m_1(1,1)	m_2(2,2)	m_3(3,3)

The solution to the male proposal game is $\begin{bmatrix} m_1 & m_2 & m_3 \\ w_2 & w_1 & w_3 \end{bmatrix}$, and the female proposal game yields the same solution. In this, one couple is a best-case scenario where both partners are each other's first choice, one pair is a second-best solution, and the final pair must settle on their last choice. Using m_1''s preferences, the solution for both proposal games is $\begin{bmatrix} m_1 & m_2 & m_3 \\ w_3 & w_1 & w_2 \end{bmatrix}$. Compared to the first game, the results are more mixed, with *p* values of final matchings of (3,1), (1,2), and (3,3), where for the first pair the wife is the parents' favourite but the son's least favourite. When m_1 is influenced by m_1', an entirely new stable solution, $\begin{bmatrix} (m_1, \mathrm{m}_1') & m_2 & m_3 \\ w_2 & w_1 & w_3 \end{bmatrix}$, emerges.

	$\boldsymbol{w_1}$	$\boldsymbol{w_2}$	$\boldsymbol{w_3}$
m_1	1,1	***2,1***	3,1
m_1'	3,1	***2,1***	1,1
m_2	***1,2***	2,2	3,2
m_3	1,3	2,3	***3,3***

We can verify that this is stable. m_1 and m_1' are indecisive between w_1, w_3, so they will stick with w_2. w_2's first choice is m_1, so she will not seek another m, and so the pair $\left((m_1, m_1'), w_2\right)$ is stable. The other two pairs can be verified in the same manner. These pairings are not stable if we consider m_1 and m_1' separately. For m_1's preferences, w_2 will deviate to m_1, which is an unchallengeable solution since (m_1, w_1) is a mutual first-choice pairing. The same is true for (m_1', w_3) in m_1''s preferences.

The solution $\begin{bmatrix} (m_1, m_1') & m_2 & m_3 \\ w_2 & w_1 & w_3 \end{bmatrix}$ is only stable because the combined preferences are indecisive between w_1, w_3, which causes m_1 to settle on a second-best choice, changing the stable marriages for everybody else as well. Compared to the solutions from the first two games, the results are ambiguous: some players are paired with a higher-ranked member of the opposite sex, some are paired with a worse partner, and some make lateral moves or end up with the same person.

5. Non-strict preferences

5.1. Tiebreaking

We return to the tied male-proposal game in section 2 where:

$P(m_1) = (w_1 \sim w_2, w_3, m_1),$ $\qquad$ $P(w_1) = (m_2, m_1 \sim m_3, w_1),$

$P(m_2) = (w_2, w_3, w_1, m_2),$ $\qquad$ $P(w_2) = (m_3, m_1, m_2, w_2),$

$P(m_3) = (w_1, w_3, w_2, m_3),$ $\qquad$ $P(w_3) = (m_1, m_2, m_3, w_3).$

At stage 2, m_1's indecisiveness resulted in an impasse:

Stage	$\boldsymbol{w_1}$	$\boldsymbol{w_2}$	$\boldsymbol{w_3}$
2	$m_1(1,2)$, $m_3(1,2)$	$m_1(1,2)$	$m_2(2,2)$

The responsible parties are the two with indecisive preferences, m_1 and w_1. As long as some mechanism exists to break a tie, a stable equilibrium can be found.

e.g. w_1 pulls a name out of a hat, but gets m_1, who is also indecisive. That is not a concern, since she can keep pulling one of two names out of a hat until he gets m_3, and there is now a stable matching since every *m* is matched with a unique *f*.

Let us make the reasonable assumption that people would prefer not to choose their mates by lot, and have another set of preferences based on some secondary attribute of the opposite sex that can help them resolve their indecision. Let us further assume that both of the indecisive parties have such a strict preference set in place.

e.g. The secondary preferences produce the following rankings:

$P(m_1') = (w_3, w_1, w_2, m_1)$, $\qquad P(f_1') = (m_1, m_3, m_2, w_1)$,

The game can now continue:

Stage	$\boldsymbol{w_1}$	$\boldsymbol{w_2}$	$\boldsymbol{w_3}$
2	m_1(1,2), m_3(1,2)	m_1(1,2)	m_2(2,2)
3	m_1(1,2)		m_2(2,2), m_3(2,3)
4	m_1(1,2)	m_3(3,1)	m_2(2,2)

With the second set of preferences acting as a tiebreaker, m_1, f_1 a stable solution $V^T = \begin{bmatrix} m_1 & m_2 & m_3 \\ w_1 & w_3 & w_2 \end{bmatrix}$ has been produced. However, the results from breaking the tie at this point is not the same as if the additional preferences were used to resolve the indifference before any matchings were made.

e.g. If we use $P(m_1'), P(w_1')$ to obtain

$P(m_1) = (w_1, w_2, w_3, m_1)$, $\qquad P(f_1) = (m_2, m_1, m_3, w_1)$,

$$V'^T = \begin{bmatrix} m_1 & m_2 & m_3 \\ w_1 & w_2 & w_3 \end{bmatrix} \neq V^T$$

While the results are not the same, both are valid tiebreaking methods.

5.2. Tiebreaking with lexicographic preferences

The fact that any tiebreaking method effectively can turn non-strict preferences into strict ones can be used to construct a more expanded version of the marriage game where all individuals judge the opposite sex by multiple attributes rather than just a single ranked list. Since it is safe to assume that marriage proposals or college applications are not judged by only one factor, this is a slightly more realistic than a single ranking. However, to keep things relatively simple we assume that preferences are lexicographic. This is a way of formally incorporating the additional preferences of section 5.1.

Assumption L1. All $m \in M, f \in F$ have lexicographic preferences for the opposite sex based on multiple attributes $A_f = (a_1, a_2, \dots, a_k)$, such that $w(A_f) = w(a_1, a_2, \dots, a_k)$ represents all characteristics of a particular female for which males have preference, and vice versa for $m(A)$. Preference set $L_m(a_1) = (l_{11}, l_{12}, \dots, l_{1n})$ represents male rankings of females based on a_1. The complete lexicographic preferences can be presented in one column vector (not a matrix):

$$L_m(a_{1\dots k}) = \begin{pmatrix} (l_{11}, & \cdots & , l_{1n}) \\ \vdots & \ddots & \vdots \\ (l_{k1}, & \cdots & , l_{kn}) \end{pmatrix}$$

e.g. $\quad w(a_1 = 3, a_2 = 10, \dots) >_m w'(a_1 = 4, a_2 = 1, \dots)$ since $a_1 >_m a_2 >_m \dots$

It does not matter that in a_2, *f'* was vastly preferred over *f*, since the choice had already been made at a_1. It can also be seen that if a_j represents the first attribute for which the opposite sex has strict preferences, then any a_k where $k > j$, are irrelevant to the stable matchings. Since the first attribute judged with strict preferences will resolve any remaining ties, further ones, whether strict or not, are no longer necessary.

e.g. If preferences for m_i still have unresolved ties at a_j, and he has strict preferences over a_{j+1}, then he will have effectively strict overall preferences once a_{j+1} has been considered, for $j \in [1, k-1]$.

5.4. Other implications.

It can be inferred that in any lexicographic preference game with *k* attributes where the strictness of preferences is stochastic, the probability of at least one stable solution approaches 1 as $k \to \infty$. Another corollary of this lexicographic model is that the basic Gale-Shapley problem can be represented with all *m*, *f* having strict preferences over a_1 and where $a_{2 \ldots k}$ are unrevealed and irrelevant to finding stable matchings.

6. Conclusion

Within the context of Gale-Shapley matching, we have shown that there can be such a thing as a fair marriage, where there are unique solutions that are optimal for both men and women. While the relative number of these single solution games dwindles as *n* grows, the absolute number still grows astronomically from the large number of permutations. It was also demonstrated that larger sets of preferences can be boiled down to an equivalent single set of strict preferences, allowing the Gale-Shapley procedure to find stability in more complex games. This created a more “realistic” scenario, in which each side of a two-sided matching problem considered more than a single attribute when choosing from the other side. In influenced games, we showed that the addition of another player acting in tandem with an existing one can change the stable set of marriages in an unpredictable way.

It would be interesting in the future to determine the mathematical properties, if any, of the unique optimal games, and if some method exists to generate such a matrix without resorting to computational testing. There is also the question of stable matchings that are not optimal for either side. When they exist, they are necessarily “between” the male and female optimum matchings and not optimal for either, and it would be interesting to find some unique characteristics of these kinds of matchings. We have shown some aspects of game theoretical matching in this paper, but there is still much potential for further research.

Appendix 1. MATLAB code and output for Monte Carlo simulations

The MATLAB code is highly modular, so that every part can be run separately if needed. For any $n > 15$, the total permutations and estimated number of permutations with generally optimal solutions will no longer be displayed, since it surpasses the double floating point range. The *parfor* function was especially useful for saving time, taking advantage of multiple processors to run several iterations of a for loop at once. *runiteration.m* can easily be modified to display the *womPropose* and *menPropose* results when there is a need to evaluate an individual game for a unique matching.

All tests used 10,000,000 iterations, except for $n = 1$ and $n = 2$, where the results were plain to see with much fewer tests, and they were in any case verifiable by hand.

A1.1. Code

runmain.m:

```
% Function for multiple runs, leave it overnight

function runmain(n,N)

parfor i=3:N            % parallel computing!
    main(n,i);
end
```

main.m:

```
% Finds the percentage of strict preference marriage matching games where
% male- and female- optimal results are the same.

function main(n,N)

counter = 0;
progressUpdate = n/100;         % split progress into 100 chunks

totalperms = factorial(N).^(2*N);

p = ProgressBar(100);           % show progress only for full percentages
for i=1:n
```

```
    test = runiteration(N);
    if test==1
        counter = counter + 1;
    end
    if mod(i,progressUpdate)==0 % only update when full percentage
        p.progress;
    end
end
p.stop;

proportion=counter/n;
bothOptimal = proportion * totalperms;

fprintf('Results for %dx%d matrix with %d iterations\n', N,N,n);
fprintf('%.10f of %.6E matrices have results that are optimal for
both\n',proportion,totalperms);
fprintf('This is roughly %.6E unique matrices\n', bothOptimal);

end
```

randmatrix.m:

```
% random matrix with specified dimensions, with unique row values
function [matrix1]=randmatrix(N)

matrix1=zeros(N,N);

for i=1:N
    matrix1(i,:)=randperm(N,N);
end

end
```

convertmatrix.m:

```
% Converts ordered preference matrix to standard form
% e.g. A(i,:)=[4 2 1 3] represents preference order with 4 highest, 3 lowest,
%      M(i.:)=[3 2 4 1] represents std form matrix with 1 in column 4
representing
%      preference for w4.

function [stdForm] = convertmatrix(A,N)

%A=[1 2 3 4;3 4 1 2;4 2 1 3;3 1 4 3]

for i=1:N
    for j=1:N
        M(i,A(i,j))=j;
    end
end
stdForm=M;
end
```

stdtopref.m:

```
% Converts ordered standard form matrix to preference order matrix
% S(i.:)=[3 2 4 1] represents std form matrix with 1 in column 4 representing
%       preference for w4.
% P(i,:)=[4 2 1 3] represents preference order with 4 highest, 3 lowest

function [prefForm] = stdtopref(S,N)

for i=1:N
    for j=1:N
        P(i,S(i,j))=j;
    end
end
prefForm=P;
end
```

runiteration.m:

```
% The meat of the program, the Gale-Shapley deferred-acceptance algorithm
% The ith element of the output is the man who will be matched to the ith
% woman.

function isEqual = runiteration(N)

isEqual=0;

men_pref = zeros(N,N);      % Preference order for the men
women_pref = zeros(N,N);    % Preference order for the women
menPropose=zeros(N,2);
womPropose=zeros(N,2);

men_pref = randmatrix(N);
women_pref = randmatrix(N);
menforwomen = galeshapley(N, men_pref, women_pref);  % men propose
womenformen = galeshapley(N, women_pref, men_pref);  % women propose

for i=1:N
    menPropose(i,2)=i;
    menPropose(i,1)=menforwomen(i);
    womPropose(i,1)=i;
    womPropose(i,2)=womenformen(i);
end

menPropose=sortrows(menPropose);        % need same male order for both

if menPropose==womPropose
    isEqual=1;
    M=convertmatrix(men_pref,N);        % remove ;s to display these
    W=transpose(convertmatrix(women_pref,N));
end

end
```

galeshapley.m:

```
%---------GALE - SHAPLEY ALGORITHM (Men propose) --------%
function stablematch = galeshapley(N, men_pref, women_pref)

men_free = zeros(N,1);
women_suitor = zeros(N,N);
women_partner = zeros(N,1);
rank = zeros(N,N);


for i = 1:N
    for j = 1:N
        for k = 1:N
        if(women_pref(i,k) == j)
            rank(i,j) = k;
        end
        end
    end
end

while (min(women_partner) == 0)
    for i = 1:N
        if (men_free(i,1) == 0)
            next = find(men_pref(i,:) > 0, 1);
            women_suitor(men_pref(i,next),i) = i;
            men_pref(i,next) = 0;
        end
    end
    for i = 1:N
        for j = 1:N
            if(women_suitor(i,j) ~= 0)
                if(women_partner(i,1) == 0)
                    women_partner(i,1) = women_suitor(i,j);
                    men_free(j,1) = 1;
                end
                if(women_partner(i,1) ~= 0)
                if(rank(i,women_suitor(i,j)) < rank(i,women_partner(i,1)))
                    men_free(women_partner(i,1),1) = 0;
                    women_partner(i,1) = women_suitor(i,j);
                    men_free(j,1) = 1;

                end
                end
            end
        end
    end
end

stablematch = women_partner;
```

BSD License for the Gale Shapley algorithm code:

A1.2. Output

```
Results for 1x1 matrix with 10000 iterations
1.0000000000 of 1.000000E+00 matrices have results that are optimal for both
This is roughly 1.000000E+00 unique matrices

Results for 2x2 matrix with 100000 iterations
0.8754200000 of 1.600000E+01 matrices have results that are optimal for both
This is roughly 1.400672E+01 unique matrices

Results for 3x3 matrix with 10000000 iterations
0.7303253000 of 4.665600E+04 matrices have results that are optimal for both
This is roughly 3.407406E+04 unique matrices

Results for 4x4 matrix with 10000000 iterations
0.5983707000 of 1.100753E+11 matrices have results that are optimal for both
This is roughly 6.586584E+10 unique matrices

Results for 5x5 matrix with 10000000 iterations
0.4867863000 of 6.191736E+20 matrices have results that are optimal for both
This is roughly 3.014052E+20 unique matrices

Results for 6x6 matrix with 10000000 iterations
0.3954897000 of 1.940841E+34 matrices have results that are opt
This is roughly7.6758262E+33 unique matrices

Results for 7x7 matrix with 10000000 iterations
0.3216391000 of 6.823819E+51 matrices have results that are optimal for both
This is roughly 2.194807E+51 unique matrices
```

```
Results for 8x8 matrix with 10000000 iterations
0.2624356000 of 4.878973E+73 matrices have results that are optimal for both
This is roughly 1.280416E+73 unique matrices

Results for 9x9 matrix with 10000000 iterations
0.2146705000 of 1.190514E+100 matrices have results that are opt
This is roughly 2.555682E+99 unique matrices

Results for 10x10 matrix with 10000000 iterations
0.1767890000 of 1.567692E+131 matrices have results that are optimal for both
This is roughly 2.771506E+130 unique matrices

Results for 11x11 matrix with 10000000 iterations
0.1455329000 of 1.680451E+167 matrices have results that are optimal for both
This is roughly 2.445609E+166 unique matrices

Results for 12x12 matrix with 10000000 iterations
0.1205030000 of 2.128566E+208 matrices have results that are optimal for both
This is roughly 2.564986E+207 unique matrices

Results for 13x13 matrix with 100000 iterations
0.1010600000 of 4.480105E+254 matrices have results that are optimal for both
This is roughly 4.527594E+253 unique matrices

Results for 14x14 matrix with 1000000 iterations
0.0838420000 of 2.145032E+306 matrices have results that are optimal for both
This is roughly 1.798438E+305 unique matrices

Results for 15x15 matrix with 1000000 iterations
0.0705930000 of Inf matrices have results that are optimal for both
This is roughly Inf unique matrices

Results for 16x16 matrix with 10000000 iterations
0.0591249000 of Inf matrices have results that are optimal for both
This is roughly Inf unique matrices

Results for 20x20 matrix with 10000000 iterations
0.0299901000 of Inf matrices have results that are optimal for both
This is roughly Inf unique matrices

Results for 21x21 matrix with 10000000 iterations
0.2576410000 of Inf matrices have results that are optimal for both
This is roughly Inf unique matrices
```

Appendix 2. R code and output for models

A2.1. Code

```
test=read.csv("e:/gdrive/!thesis/algorithmtesting/results.csv");
pctModel=lm(logPct~n,test);

summary(pctModel);
```

A2.2. Output

Graphs of percentage and log of the total possible games with generally optimal stable matchings.

Summary of the percentage of games model:

```
Call:
lm(formula = logPct ~ n, data = test)

Residuals:
      Min        1Q    Median        3Q       Max 
-0.037660 -0.023640 -0.001452  0.023202  0.040203 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept)  0.231391   0.013875   16.68 1.24e-10 ***
n           -0.193731   0.001435 -135.01  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.02646 on 14 degrees of freedom
Multiple R-squared:  0.9992,	Adjusted R-squared:  0.9992 
F-statistic: 1.823e+04 on 1 and 14 DF,  p-value: < 2.2e-16
```

References

Adachi, H. (2000). On a characterization of stable matchings. *Economic Letters, 68*, 43-49.

Gale, D., & Shapley, L. S. (1962, January). College Admissions and the Stability of Marriage. *The American Mathematical Monthly*, *69*(1), 9-15.

Roth, A. E., & Sotomayor, M. (1992). Two-Sided Matching. In S. Hart, & R. J. Aumann (Eds.), *Handbook of Game Theory, Volume 1* (pp. 485-541). Elsevier Science Publishers. Retrieved from http://web.stanford.edu/~alroth/papers/92_HGT_Two-SidedMatching.pdf