Game Theory Complete Write-Up

Michael Quintin

Contents

1 Overall Introduction **2**
1.1 What is Game Theory? . . . 2

2 Strategic Form Games **3**
2.1 Introduction . . . 3
2.2 Payoff Matrices . . . 3
2.3 Nash Equilibrium . . . 4
2.4 The Maxmin Concept . . . 5

3 Extensive Form Games **6**
3.1 Introduction . . . 6
3.2 Game Trees . . . 6
3.3 Winning Strategies and Chomp . . . 8
3.4 Backwards Induction . . . 8
3.5 Formatting Equilibrium Answers and The Trembling Hand Theory . . . 10
3.6 Reduced Strategic Form . . . 10

4 Utility Theory **12**
4.1 Introduction . . . 12
4.2 The Utility Function . . . 12
4.3 Preference Relations . . . 12
4.4 Lotteries . . . 14
4.5 Risk . . . 15

5 Stable Matching **20**
5.1 Preference Relations, Extended . . . 20
5.2 Stable Matchings and Stable Matching . . . 21
5.3 The Gale-Shapley Algorithm . . . 21
5.4 Deriving the Bias . . . 22
5.5 Why is This Algorithm (and Proof) Useful? . . . 26

Chapter 1

Overall Introduction

1.1 What is Game Theory?

Game theory is a series of analytical frameworks that allow us to play with the outcomes of given scenarios. It is more succintly called the study of strategy, because the act of playing with a situation's outcome is the act of behaving strategically. It is an extremely young branch of economics, with infinitely more room for exploration.

Game theory has a reuptation for being immoral, but it is not. It is amoral. The key difference lies in motivation: immorality means going against morality, whereas amorality, and game theory, simply doesn't take morality into account at all. With how situational game theory can be, it is easy to see it as manipulative and evil, but its conclusions are simply the product of its inputs — meaning, if you wish game theory to be moral, you must explicitly alter it to be so.

Lastly, do not get bogged down by symbols. Comb through the explanations provided in these few chapters and you will see every symbol is far less complicated than it looks. Usually, game theory variables cannot be given a numerical value: a greek letter, for example, could be used to represent the real-world outcome of a scenario, like receiving a free car or successfully executing a backflip.

With all this in mind, enjoy the game theory content.

Chapter 2

Strategic Form Games

2.1 Introduction

Strategic-Form games involve players competing simultaneously, each with a set of posible moves, called **strategies**, with different combinations of strategies, called **vectors**, correlated with different possible outcomes of the game, called **payoffs**. Rock-paper-scissors is a commonly played strategic-form game, with the strategies of throwing rock, paper, or scissors, with the vectors of different combinations (e.g. rock versus paper, paper versus paper, rock versus scissors, etc.), and with the outcomes of these vectors being either player winning or a tie (in the case of both players choosing the same strategy).

2.2 Payoff Matrices

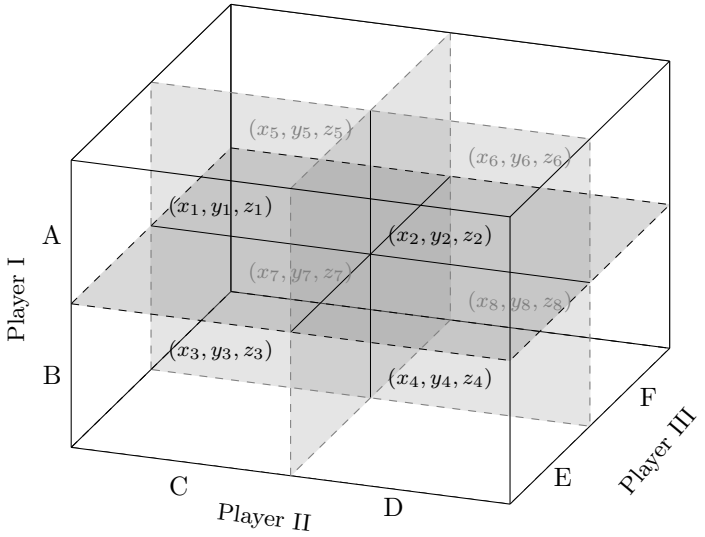
Strategic-form games can be modeled with **payoff matrices**. An example of a payoff matrix is below:

		Player II	
		A_{II}	B_{II}
Player I	A_{I}	(x_1, y_1)	(x_2, y_2)
	B_{I}	(x_3, y_3)	(x_4, y_4)

Player I and Player II are each allocated a dimensional axis of the matrix, with their strategies determining the number of rows or columns on that axis. In the above case, we find that there two players, so two dimensional axes are needed (resulting in a two-dimensional matrix), that each player has two strategies leading to four vectors, so each player gets two columns or rows (depending on what axis they're on) and the matrix has four cells to hold the four outcomes — each labeled with the payoffs x and y for Player I and Player II respectively. If Player I chooses action A_I and Player II chooses action $A_I I$, the outcome for Player I would be x_1 and the outcome for Player II would be y_1.

Payoff matrices usually appear as two-by-two tables, but, understanding that each dimensional axis represents a player and each row or column represents a given strategy, payoff matrices can be multi-dimensional and contain many rows or columns, depending simply on how many players there are and how many strategies each player has.

Games with more than two players are represented by matrices with more than two dimensional axes (e.g. yielding 3-dimensional or 4-dimensional matrices). Below is a three-player game modeled where each player has two strategies, and there are resultantly eight outcomes.



In the above payoff matrix, the players are Player I, Player II, and Player III. The strategies are A and B (for Player I), C and D (for Player II), and E and F (for Player III). The outcomes are all labeled with x, y, and z, representing the payoff to Player I, Player II, and Player III respectively.

With a strategic-form game modeled, we can begin to think about its outcome, if both players rationally decide which strategy is best to take. Finding this likely outcome is called finding the game's **equilibrium**. There are mutliple frameworks we can use to analyse potential outcomes of a game like this. This chapter will outline two of these frameworks.

2.3 Nash Equilibrium

One method of determining the outcome of a strategic-form game is to assume both players want to maximize their own payoff, and given their opponent's move, will alter their strategy to get the highest benefit among the possible options restricted to their opponent's action. Take the following strategic-form game, for example.

		Player II		
		A_{II}	B_{II}	C_{II}
Player I	A_{I}	$(0, 6)$	$(6, 0)$	$(4, 3)$
	B_{I}	$(6, 0)$	$(0, 6)$	$(4, 3)$
	C_{I}	$(3, 3)$	$(3, 3)$	$(5, 5)$

Understanding that the second number in a cell's brackets represents the payoff to Player II, if Player I chose strategy A_I, Player II would maximize his payoff by choosing strategy $A_I I$. But if Player II chose $A_I I$, Player I would maximize his payoff by changing his strategy to B_I. But if this were the case, Player II would change his strategy to $B_I I$, which would then lead Player I to change his strategy to A_I, and the two players would keep changing their strategy in a loop. John Nash, who won the Nobel Prize in economics for his work regarding strategy interactions and responses between players, discovered something called the **Nash Equilibrium**, which represents the opposite of the above scenario: *a Nash Equilibrium is a point in a given situation at which the two or more players involved cannot improve their own outcome by changing their strategy.*

Finding Nash Equilibrium is as simple as looking through all the cells in a given game and finding the cells where neither player can improve their payoff by changing their strategy. There can be multiple

Nash Equilibria, but in the above game there is only one: $(5, 5)$.

Nash Equilibrium is not a perfect concept. Take the following strategic-form game, for example.

		Player II	
		L	R
Player I	T	$(2, 1)$	$(2, -20)$
	M	$(3, 0)$	$(-10, 1)$
	B	$(-100, 2)$	$(3, 3)$

Nash equilibrium technically exists at (B, R), but Player I would likely hesitate to choose B, since if Player II chose L instead of R by mistake, the change in payoff would be catastrophic for Player I. Considering the possibility that other players may make a mistake is called **accounting for a trembling hand**, calling to a common economics story where a man has a choice between two buttons, and his trembling hand leads him to accidentally click the wrong one.

But if there were potentially catastrophic consequences for either player on the board, is there a better way to analyze the likely outcome of the game than finding Nash Equilibrium?

2.4 The Maxmin Concept

If we assume that players' primary focus is on avoiding a potential loss instead of maximizing their payoff given their opponent's move, we should conduct **Maxmin Analysis** instead of finding Nash Equilibrium. Maxmin Analysis uses the minimum possible outcome for a given player at every strategy of their opponent to determine what strategy that player can use to minimize their potential loss, by maxmimizing their minimum possible outcome. Take the following strategic-form game:

		Player II	
		L	R
Player I	T	$(2, 1)$	$(2, -20)$
	M	$(3, 0)$	$(-10, 1)$
	B	$(-100, 2)$	$(3, 3)$

Since this game incurs a lot of loss and the maxmin concept is all about maximizing the losing outcome (or minimum payoff: cutting losses) of a game, we can use Maxmin Analysis here to find a reasonable outcome solution. To proceed with Maxmin Analysis, we can redraw our game as follows:

		Player II		
		L	R	*Min Player I*
Player I	T	$(2, 1)$	$(2, -20)$	2
	M	$(3, 0)$	$(-10, 1)$	-10
	B	$(-100, 2)$	$(3, 3)$	-100
	Min Player II	0	-20	

To find the equilibrium using Maxmin Analysis, we'll then find our **maxmin value**, which is a pair of values expressed like a coordinate: (highest minimum for Player 1, highest minimum for Player 2). For the minmax analysis above, our maxmin value would be $(2, 0)$. Using this minmax value, we then say that the strategy for a player that guarantees their maxmin value payoff (or includes their maxmin value payoff and greater) is the strategy they should choose, and the resultant strategy vector is the equilibrium outcome of the game. For the game above, strategy T guarantees Player I his maxmin value payoff, and strategy L guarantees Player II his maxmin value payoff or better. Therefore, T and L is the equilibrium strategy vector and $(2, 1)$ is the game's equilibrium.

Chapter 3

Extensive Form Games

3.1 Introduction

Extensive-form games are, in essence, two-player games where players take turns making moves. Chess is an extremely famous example of an extensive-form game. Formally, extensive-form games must have:

1. Clear rules of play established
2. A defined order of play
3. Information available or not available to players at each move
4. Rules for when and how the end of the game occurs
5. An outcome at each possible ending of the game.

3.2 Game Trees

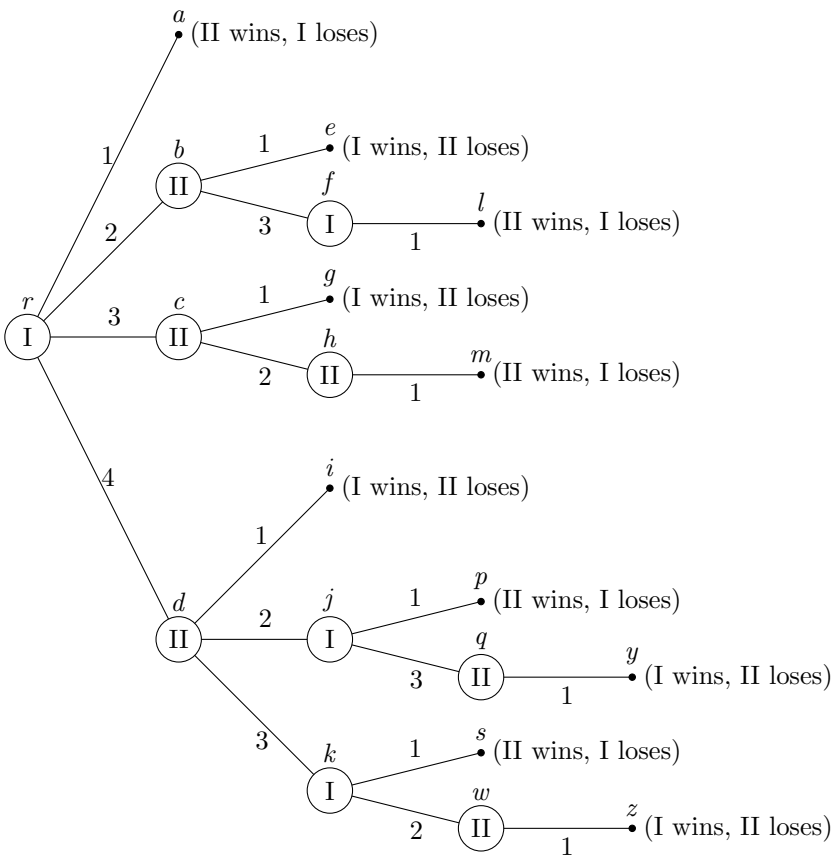
To analyze an extensive-form game, it is best to diagram it with a **game tree**. For example, take a game played on the following board:

2	4
1	3

Rules:

1. A square may be captured by a player if not captured before.
2. Square 4 may not be captured if square 2 or square 3 has been previously captured.
3. The game ends when square 1 is captured, with the player who captured square 1 being the loser.

The game above can be represented with the following tree diagram:



A **vertex** or **vertice** is a point along the game tree. The vertices in the diagram above are r, a, b, c, d, e, f, g, h, i, j, k, l, m, p, q, s, w, y, and z. Note that some letters are not used to label vertices; this is because they have other purposes. Also, note that the **root vertice** is always labeled with r. Vertices represent game states, and resultantly, each vertice can only have one previous path of moves leading to it.

An **edge** is a line connecting vertices. Please note that in typical game-theory talk, edges are not usually described as "connecting" vertices. Usually, edges are described to be **emanating from** vertices. Each edge is labeled with the action the player is taking to follow along that path. In the game tree above, all actions available to players are taking squares on the board. Therefore, for this game, edges are labeled with numbers 1-4, representing a respective player taking that number space on the board. Edges represent moves in the game.

A **leaf** is an endpoint of the tree. At each leaf, the respective game outcome is mentioned. Leaves of the tree above are a, e, l, m, i, p, y, s, and z.

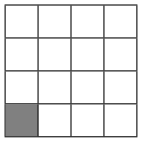
A vertex is represented notationally with x^n, where x^0 is the root vertex and x^1 refers to the vertex emanating from the root. However, when talking about vertices, like with the tree provided above, it is also common to use the vertice labels; i.e. for the game above, referring to the root vertex as r and all others by their respective letters.

A **path** in an extensive-form game is defined as the string of edges and vertices connecting an end vertex and start vertex. We represent path length with k, a numerical variable that describes how many vertices have been traversed to get to a given end vertex. If $k \geq 1$ and $x^1 = x^{k+1}$, the path is

called **cyclic**. Cyclic paths can be visualized in game trees like the one below:

3.3 Winning Strategies and Chomp

A **winning strategy** in an extensive form game is defined as a strategy that a player can implement to ensure that he wins no matter the moves of the other player. Take the following game, chomp, played on a four-by-four board.



Rules:

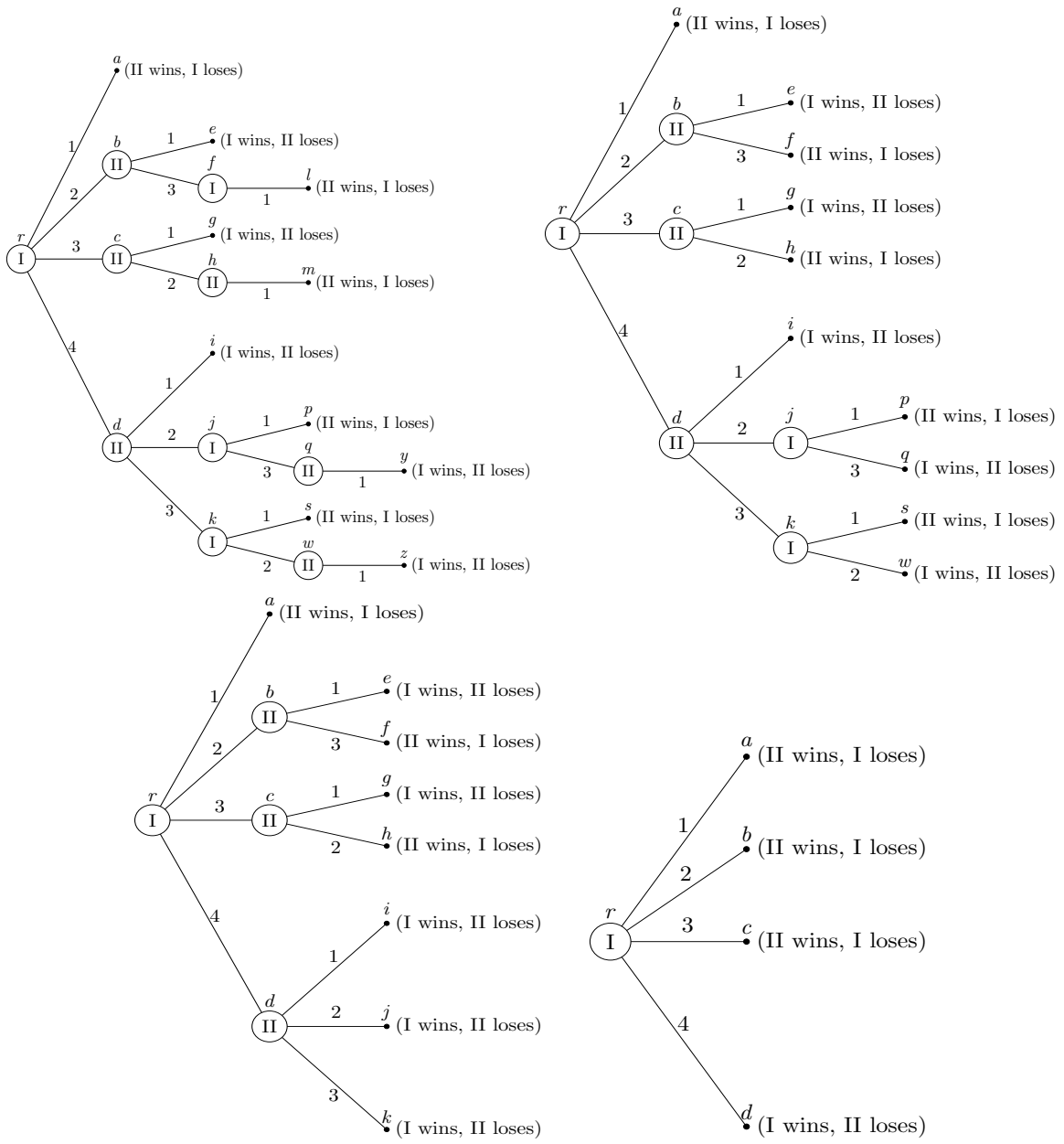
1. Player 1 acts first, and player 2 acts second. The players then take turns.
2. When a player captures a square, he captures all squares above, all squares to the right, and all squares in the area of those borders.
3. A square may be captured by a player if not captured before.
4. The entire board is captured until the gray square remains.
5. The game ends when a player captures the gray square and therefore loses.

Player 1 has a winning strategy for the game described above. Player 1's winning strategy in the game above is to capture the square diagonally adjacent to the gray square (and thus all squares above, to the left, and within those two borders), and then mirror all future movements of player two over the diagonal axis of the table. It is important to understand that player 1's winning strategy does not just apply to a board of this size; the board could be square but of infinite length and height, and player 1's strategy would still be a winning one. Diagramming out chomp with a tree, assuming player 1 takes the diagonally adjacent square first, we can see the winning strategy in action.

Winning strategies are a great way to determine the outcome of a game, because we can assume that if both players play rationally, the player with a winning strategy will simply play that strategy always. However, even if there are no winning strategies, or if a game is points-based instead of win/loss based, we can still find its outcome.

3.4 Backwards Induction

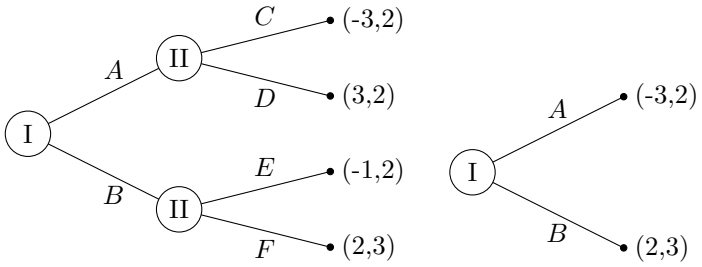
To find the outcome of an extensive-form game, we can use **backwards induction**. Using backwards induction is simple: imagine the last game move — right before the outcome — as its own little game, where one player is deciding between two choices. Obviously, this player will choose the better outcome for them (whether this means more points or winning the game). As a result, we can essentially say that the vertex where the player makes such a choice is equivalent to the outcome itself, because it's guaranteed that the player will maximize their own gain and choose such an outcome. Then, we can keep moving backwards, doing the same for the previous move, and the previous, and so forth until we reach the root vertex. For example, here is backwards induction conducted on the game tree representing the game defined in section 3.2:



Since this game, using backwards induction, can be simplified to Player I choosing between (II wins, I loses), (II wins, I loses), and (I wins, II loses), as seen in the final game tree diagrammed, we know that Player I will chose to win, and therefore that Player I winning will be the outcome of the game, assuming players both choose their best possible strategy. Also, knowing that the outcome of the game given rational play is that one player wins, means that player has a winning strategy. Backwards induction does not tell either player what the optimal strategy is, rather simply what the game's outcome will be given optimal play.

3.5 Formatting Equilibrium Answers and The Trembling Hand Theory

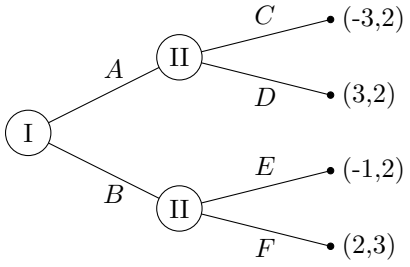
When writing the equilibrium of a game found using backwards induction, if asked to account for a trembling hand, the expressed outcome of a game must also feature the game's outcomes if all moves worse than the best were chosen, separated by semicolons.



For the tree above, if we account for a trembling hand on the opening move, we are accounting for Player I not only choosing his best move, but his second-best (which, in this case, is also his worst) and any outcome worse than that (if Player I had more than two possible actions on his opening move), listed in decreasing sequential order of the outcome to Player I. The outcome answer would therefore be written as $(2, 3); (-3, 2)$.

3.6 Reduced Strategic Form

Another way to find the equilibrium of an extensive-form game is to convert it to a strategic-form game and then find the equilibrium of that strategic-form game using the techniques defined in Chapter 1. We convert all the posible combinations of player actions along the game tree into the different strategic-form game strategies, and use the extensive-form game outcomes as the strategic-form game payoffs. Accordingly, the following tree would correspond to the following table.



		Player II			
		C, E	C, F	D, E	D, F
Player I	A	$(-3, 2)$	$(-3, 2)$	$(3, 2)$	$(3, 2)$
	B	$(-1, 2)$	$(2, 3)$	$(-1, 2)$	$(2, 3)$

We know that Player I's strategies are A or B. But since Player II does not know whether Player I will choose A or B, Player II's strategies have to operate independently of Player I's first move — meaning Player II's denoted strategies have to account for the two possible moves of Player I by choosing two possible responses for each possible move. Player II's strategies will be combinations of the moves he could make on the second turn of play, namely: C, E; C, F; D, E; or D, F. In this way, Player II doesn't necessarily know what Player I will choose, but has chosen a move for each possible Player I action, by choosing a response to each possible Player I action within one single strategy. If Player II

chose strategy C, E and Player I chose strategy B, the outcome of the game would be $(-1, 2)$, because Player II's strategy essentially says that *if Player I chooses A, then I will take action C, but if Player I chooses B, then I will take action E.*

Chapter 4

Utility Theory

4.1 Introduction

Utility theory lays the groundworks for game theory and most of economics as a whole — it is the study of the evaluation of outcomes in terms of a measured, numerical utility, and how that utility relates to player action.

4.2 The Utility Function

The utility function, central to the application of numerical utility is defined as follows:

$$u : O \to \mathbb{R}$$

This leaves the utility function to be defined by any mathematical formula. For example, the utility for a set of given outcomes could be represented by a parabolic graph, or a cubic one, or any other type of function — as long as it converts the input O, an outcome, into a real number $\mathbb{R}$, representing a level of utility to the respective individual.

It is important to note that as a result of this definition, there is no "intensity of benefit" described by the utility function; that is, an outcome with a numerical utility of 20 is not twice as beneficial as an outcome with a numerical utility of 10. Instead, the utility function only finds value in comparison — so for the outcome of 20 utility and 10 utility described before, the only thing we could conclude is that the 20 utility outcome is simply more useful than the 10 utility one.

Please note, when talking about utility functions, it is common to call them **expected utility functions**. While both are correct, technically because the utility functions are the result of player estimations of their own utiility from a certain outcome, it is more correct to call them *expected* utility functions instead.

4.3 Preference Relations

To be able to analyze utility relations, it is fundamental to be able to understand what ouctomes players prefer more than others. Therefore, preference relation notation was developed, which plays out as follows.

For the outcomes x and y and for $i = 1, 2...k$ where k is the number of players...

$$x \approx_i y$$

Means "player i prefers outcome x and y equally."

$$x \succ_i y$$

Means "player i prefers outcome x to outcome y."

$$x \prec_i y$$

Means "player i prefers outcome y to outcome x."

$$x \succsim_i y$$

Means "player i prefers outcome x to outcome y or prefers them equally."

$$x \precsim_i y$$

Means "player i prefers outcome y to outcome x or prefers them equally."

The $\succsim_i$ or $\precsim_i$ symbol is also called "weak preference," and the $\succ_i$ or $\prec_i$ symbol is also called "strong preference."

With these abstract preference relations, we can bring them into the numerical world by using the utility function. To do so, we translate preference relations as follows:

$$x \approx_i y \iff u_i(x) = u_i(y)$$

$$x \succ_i y \iff u_i(x) > u_i(y)$$

$$x \prec_i y \iff u_i(x) < u_i(y)$$

$$x \succsim_i y \iff u_i(x) \geq u_i(y)$$

$$x \precsim_i y \iff u_i(x) \leq u_i(y)$$

Finally, in order to make sure our system of preference relations has structure and can be manipulated while holding itself consistent, we make three assumptions about preference relations themselves:

1. The assumption of **Completeness**
2. The assumption of **Reflexivity**
3. The assumption of **Transitivity**

These assumptions are not incredibly complicated, and yet we still can come up with real-world examples where they don't hold true. This is to say: these assumptions are one of many oversimplifications in utility theory, but we work with them regardless.

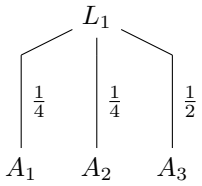
The assumption of **Completeness** assumes that for any pair of outcomes x and y in the set of possible outcomes O, either $x \succsim_i y$, $y \succsim_i x$, or both (meaning $x \approx_i y$). Essentially, we are just assuming players are able to actually analyze which ouctome or event they prefer between two options. As simple as this assumption may seem, it does not always hold true: in the case of two divorcing parents, for example, their child will likely not be able to determine which parent he prefers, and likely wouldn't even be able to say he preferred them equally.

The assumption of **Reflexivity** assumes that for every $x \in O$, $x \approx_i x$. This one is pretty obvious and is basically impossible to break, as it is the assumption that player will prefer a given outcome x from the set of all outcomes O equally to itself.

The assumption of **Transitivity** assumes that if $x \succsim_i y$ and $y \succsim_i z$, then $x \succsim_i z$. This allows us to make conclusions about preference relations, because otherwise we wouldn't be able to navigate a system of variables with the preference operation $\succsim_i$, although this assumption is just as frequently broken as the assumption of Completeness. An incredibly intriguing example of this assumption being broken is the **Condorcet Paradox**, which is a great rabbit hole to dive down.

4.4 Lotteries

Lottery outcomes describe outcomes down to chance. We cannot meaningfully analyze a player's preference for each individual outcome of a lottery without at least weighing those outcomes by their probabilities. For example, if for the following diagram of lottery L_1:



And if, for player θ:

$$u_\theta(A_1) = 6,\ u_\theta(A_2) = 3,\ u_\theta(A_3) = 4$$

To calculate the **expected utility of the lottery**, we can apply the utility function to each outcome of the lottery and sum those utilities weighted by the respective outcome's probabilities, notated as $u_\theta(\mathbb{E}[L_1])$. This would be calculated as follows:

$$O = \{A_1, A_2, A_3\}$$

$$u_\theta(\mathbb{E}[L]) = \sum \left(u_\theta(\forall_x \in O)\,(p(x))\right)$$

$$u_\theta(\mathbb{E}[L_1]) = u_\theta(A_1)\left(\frac{1}{4}\right) + u_\theta(A_2)\left(\frac{1}{4}\right) + u_\theta(A_3)\left(\frac{1}{2}\right)$$

$$u_\theta(\mathbb{E}[L_1]) = 6\left(\frac{1}{4}\right) + 3\left(\frac{1}{4}\right) + 4\left(\frac{1}{2}\right) = 4.25$$

It is extremely important to understand though that the expected utility of the lottery is not calculated in the same way as the utility of the expected outcome of the lottery. The **expected outcome** of the lottery is defined as follows, denoted with μ_L:

$$\mu_L = \sum_{k=1}^{K} (p_k x_k)$$

The expected outcome of a lottery is therefore just the individual outcomes weighted by their probabilities, or a weighted average (μ) of the lottery. We can then find the **utility of the expected outcome of the lottery** with the following expression:

$$u(\mu_L) = u\left(\sum_{k=1}^{K} (p_k x_k)\right)$$

As stated earlier, finding a player's utility for this average outcome is not the same as finding the expected utility of the lottery. The difference in calculation of these two values allows us to assess a player's propensity to risk, because it impacts how much a player likes to "gamble" their winnings.

To clarify: we are differentiating between (for a given player) the utility of the expected outcome of a lottery versus the expected utility of the lottery.

4.5 Risk

To assess a player's propensity to risk, we can evaluate what that player's preference relation is with respect to the expected outcome of a lottery versus the lottery itself. A player is described as:

Risk-averse if:

$$\mu_L \succ_i \mathbb{E}[L] \text{ and therefore } u_i(\mu_L) > u_i(\mathbb{E}[L])$$

Risk-seeking if:

$$\mu_L \prec_i \mathbb{E}[L] \text{ and therefore } u_i(\mu_L) < u_i(\mathbb{E}[L])$$

Risk-neutral if:

$$\mu_L \approx_i \mathbb{E}[L] \text{ and therefore } u_i(\mu_L) = u_i(\mathbb{E}[L])$$

But why?

Understanding that μ, the expected outcome of a lottery, essentially represents the "average" or "typical" outcome of that lottery, it can be interpreted to also represent a practically guaranteed result. If a player is risk-averse, they will prefer this guaranteed result to the lottery itself, even though the lottery could potentially offer greater returns (just with a lower probability). This means the player will find less utility in gambling with the lottery, despite the potential for higher returns, than just getting an average outcome. If a player is risk-seeking, they're likely looking for those higher returns the lottery could offer, and find it less useful to go for a guaranteed result, in favor of the gamble and risk. If a player is risk-neutral, they don't mind either taking the lottery or the guaranteed outcome, because they simply don't care about risk.

Imagine we defined a lottery with two outcomes: ω occuring with probability α, and γ occuring with probability $(1 - \alpha)$. The lottery would be defined as follows:

$$L_{\omega\gamma} = [\alpha(\omega), (1 - \alpha)(\gamma)]$$

The expected utility of the lottery would be calculated as follows:

$$u_i(\mathbb{E}[L_{\omega\gamma}]) = \alpha(u_i(\omega)) + (1 - \alpha)(u_i(\gamma))$$

And the utility of the expected outcome of the lottery would be calculated as follows:

$$u_i(\mu_{L_{\omega\gamma}}) = u_i(\alpha(\omega) + (1 - \alpha)(\gamma))$$

We know that the utility function can be represented with any kind of linear function, so if we graph utility over outcome, and then a player's utility function, we can compare the utility (y-values) of the expected utility of the lottery of ouctomes and the utility of the expected outcome of the lottery of outcomes. Doing so many times over, we'd find that the concavity of the graph indicates the risk attitude of the player.

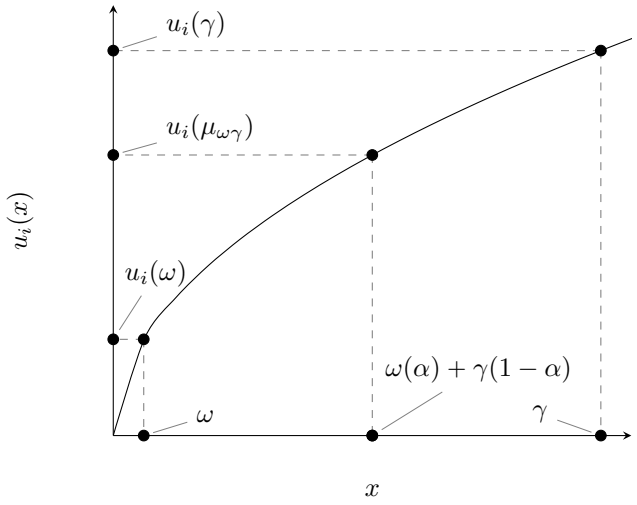
Concave expected utility graphs indicate the player is risk-averse, because $u_i(\mu_L) > u_i(\mathbb{E}[L])$.

Convex expected utility graphs indicate the player is risk-seeking, because $u_i(\mu_L) < u_i(\mathbb{E}[L])$.

Straight expected utility graphs indicate the player is risk-averse, because $u_i(\mu_L) = u_i(\mathbb{E}[L])$.

Let's graph a concave utility function, a convex utility function, and then a straight utility function to visually see the differences between $u_i(\mu_L)$ and $u_i(\mathbb{E}[L])$.

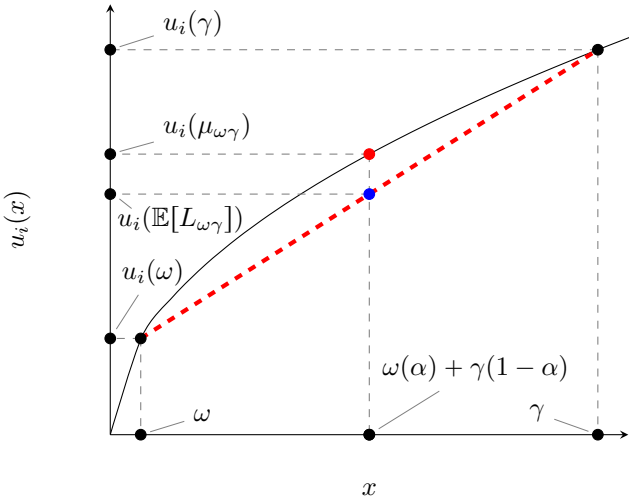
For concave utility functions:



In the graph above, we have plotted the point on the x-axis between ω and γ where $\alpha = 0.5$, since the point is directly between ω and γ. If we were to move this point closer to ω, we'd simply raise the value of α, and vice versa for γ. Recall that $\omega(\alpha) + \gamma(1-\alpha) = \mu_{L_{\omega\gamma}}$.

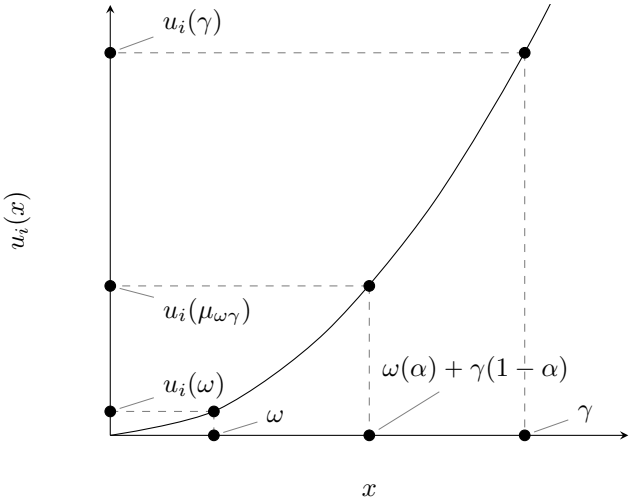
The black line, which represents this player's utility function per outcome, is concave, because it is increasing at a decreasing rate. A good test for concavity is the "water test," where you imagine what water would do if you poured it on top of this graph. Because the water would slide off, this graph is concave.

Because we know α has been set to 0.5, we know that when calculating $u_i(\mathbb{E}[L_{\omega\gamma}])$ we are essentially taking the average of the two outcomes' utility values on the y-axis (because, recall, the utility of a lottery is the utility of each outcome multiplied by their probability, which can be viewed as the calculation for expected value, or average, of the utility of every lottery outcome), so the y-coordinate — just like the x-coordinate — for $u_i(\mathbb{E}[L_{\omega\gamma}])$ will sit in the middle of the utilty for ω and the utility for γ. We can now redraw the graph, incorporating a red line to more easily demonstrate this middle characteristic.



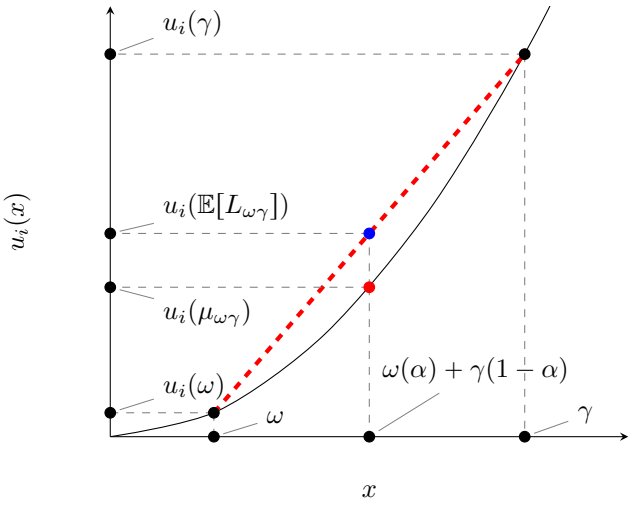
As you can see, the y-value for $u_i(\mu_{L_{\omega\gamma}})$ sits above the value for $u_i(\mathbb{E}[L_{\omega\gamma}])$, because the utility function (black line) curves above the red dotted line, which has been drawn in to help demonstrate that $u_i(\mathbb{E}[L_{\omega\gamma}])$ sits directly in between the y-values for ω and γ. Understanding, as well, that changing the value of α would mean moving the blue and red points along their respective lines at the same x-coordinate (more towards γ if α increased and more towards ω if α decreased), we can see that the red point will always remain above the blue point, and as a result, $u_i(\mu_{L_{\omega\gamma}})$ will always have a higher value than $u_i(\mathbb{E}[L_{\omega\gamma}])$. This, as a result, shows that this player will always prefer the guaranteed outcome of this lottery over the lottery itself, so they are risk-averse.

For convex utility functions:



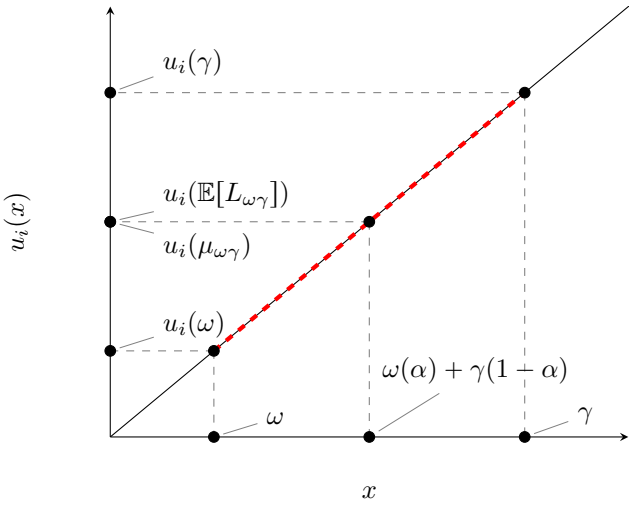
The opposite is true to the graph above. This utility funciton is convex because it is increasing at an increasing rate. A good test for concavity is the "water test," where you imagine what water would do if you poured it on top of a graph. Because water would be held by this graph like a cup, this graph is convex.

We've also labeled on this graph the two outcomes ω and γ as well as the expected outcome calculation $\omega(\alpha) + \gamma(1-\alpha)$, which is equivalent to $\mu_{L_{\omega\gamma}}$.



We can see, just like above, that no matter what we set α to be, the blue point will always be above the red point, meaning $u_i(\mathbb{E}[L_{\omega\gamma}]) > u_i(\mu_{L_{\omega\gamma}})$. This means that this player finds more utility (i.e. prefers) taking the chance of the lottery over its average result, meaning this player is risk-seeking.

For straight utility functions:



The black line represents a straight utility function. It is straight because it is increasing at a constant rate. The "water test" does not apply here because this graph does not have concavity.

We can see that there is no need for a red and blue dot, because no matter the value of α the utility (y-coordinate) of $\mathbb{E}[L_{\omega\gamma}]$ and $\mu_{L_{\omega\gamma}}$ will always be equal. Since the utility of both the lottery and guaranteed outcome are identical, this player must be making his choice based on a different comparison. The only comparison between an average outcome and a lottery outcome is the difference in risk incurred, so thus risk makes no difference in utility to this player, so this player does not incorporate risk into their decision: they are risk-neutral.

What is the utility in understanding risk profiles like so? Well, in the case of a lottery of two outcomes, if we survey people on how much utility they get from being able to receive either outcome at different probabilities (i.e. change α), we can graph their utility function and determine whether or not they

like taking risks based on the curvature of the function. The curvature of a player's utility function, which we determine by asking him how much he likes different types of outcomes, tells us whether he likes to gamble or not.

Chapter 5

Stable Matching

5.1 Preference Relations, Extended

A **preference profile** is a player's ranking (or communication of preference) along a set of multiple outcomes. For example, if a given situation's outcomes can be represented with x, y, z and the players in such a situation can be represented with A, B, C, an example set of preference profiles for the three players (one each) could look as follows:

$$x \succ_A y \succ_A z$$

$$x \approx_B y \succ_B z$$

$$z \approx_C x \prec_C y$$

We can see in player A's preference profile that he prefers outcome x over outcome y over outcome z. We can conclude, therefore, that if A were to choose which of the three outcomes in the given scenario he could receive, x would be his first choice, y his second, and z his third.

We can see in player B's preference profile that he is indifferent between outcomes x and y, but prefers outcome y to outcome z. Since outcomes x and y are preferentially equivalent for player B, we can conclude that player B also prefers outcome x to outcome z. B therefore ranks x and y first and then z second.

We can see in player C's preference profile that he is indifferent between outcomes z and x, but prefers outcome y to outcome x. Like above, because outcomes z and x are preferentially equivalent to player C, we can conclude he also prefers outcome y to outcome z. C therefore ranks y first, then z and x.

Not only can we have three players rank three outcomes, but we can have three players rank three other players, and vice versa. Take the following case, where three high-school students each rank three colleges and those three colleges each rank the three students. The six preference profiles involved could look like so:

Jack, Lily, Bob (students)	Harvard, Princeton, Yale (colleges)
$Harvard \succ_{Jack} Princeton \succ_{Jack} Yale$	$Jack \succ_{Harvard} Lily \succ_{Harvard} Bob$
$Yale \succ_{Lily} Princeton \succ_{Lily} Harvard$	$Bob \succ_{Princeton} Jack \succ_{Princeton} Lily$
$Yale \succ_{Bob} Harvard \succ_{Bob} Princeton$	$Bob \succ_{Yale} Lily \succ_{Yale} Jack$

Now, imagine we had to pair students to colleges, knowing where each student ranked each college and where each college ranked each student. We could do it algorithmically, or build the matching ourselves. How could we pair students to colleges so that everyone was happy?

5.2 Stable Matchings and Stable Matching

This question, of how can we form pairs from two different groups knowing each individual's ranking of the members in the other group, is where we introduce the concept of **stable matching**. A **matching** is a series of pairings between individuals in two separate groups. A **stable matching** is a series of pairings between individuals in two separate groups such that after being paired, *no two individuals in different pairs prefer each other over their assigned partner.* If two individuals in different pairs preferred each other to their assigned partner, they would abandon their partners to instead pair together, ruining the stability of the overall matching. Our goal in creating stable matchings is not necessarily to pair people such that everyone gets their highest choice or number-one ranked partner, but rather to ensure only that no two individuals prefer each other over their assigned partner. We call this type of cross-pair preference between two people, specifically with regard to the fact that it screws up stability, a **blocking pair**. To create stabiility, we must avoid creating a matching that leads to a blocking pair. A matching without any blocking pairs is stable.

For example, take the following matching of students to colleges for the situation defined with the preference profiles above.

$$\begin{array}{c} students \longrightarrow colleges \\ \hline (Jack, Princeton) \\ (Lily, Harvard) \\ (Bob, Yale) \end{array}$$

This matching is not stable. As we can see in the preference profiles provided for the students and colleges, Jack prefers Harvard to Princeton ($Harvard \succ_{Jack} Princeton$) and Harvard prefers Jack to Lily ($Jack \succ_{Harvard} Lily$); both Jack and Harvard prefer each other to their assigned partner, meaning they are a blocking pair in this matching. Jack and Harvard would reject the matching, and instead pair together, making the matching unstable.

But how can we think about this algorithmically?

5.3 The Gale-Shapley Algorithm

David Gale and Lloyd Shapley created an algorithm to solve this issue, aptly named the **Gale-Shapley Algorithm**. It works as follows:

Recall the general scenario: two distinct groups have each ranked all individuals in the opposite group. Now imagine that we called one group the "applicant group" and the other group the "selector group." Using these two denominations, we can run the algorithm, which abides by the following steps:

1. Each member in the applicant group finds their top-ranking individual in the selector group, and applies for a partnership with them.
2. If any member in the selector group has multiple applicants to choose from, they pick their highest-preferred applicant and reject the rest.
3. Any rejected individual in the applicant group finds their next-highest-preferred individual in the selector group and applies for a partnership with them.
4. Repeat steps 2-3 until all applicants have been matched to a selector.

Imagine the scenario where we are trying to match men and women for marriage. If we decided men would be the applicants — meaning they would propose to the women — and we decided women would be the selectors — meaning they'd choose from the men who proposed to them — the algorithm would look something like this:

1. Each man finds his top-ranking woman and proposes to her.
2. If any woman has multiple men to choose from, she picks the highest-ranking man of those who have proposed to her, and rejects the rest.
3. Any rejected man finds his next-highest-ranking woman and proposes to her.
4. Repeat steps 2-3 until all men have been matched to a woman.

This algorithm always leads to stable matchings. Why? By the very nature of the fact that men are rejected by the women, we know that any man who is not with his top choice has already been rejected by the woman or women above who he is paired to — but the woman or women would only have rejected him to be with another man they prefer even more, and will therefore not want to abandon the man they are paired with to be with the man they rejected. The only man or men a given woman in the Gale-Shapley Algorithm can prefer more than her assigned partner is a man she has not rejected, meaning a man that does not reciprocally prefer her highly and thus did not propose to her. The Gale-Shapley Algorithm will always produce stable matchings.

The Gale-Shapley Algorithm seems to have been taken from real life; when written in the context of marriage, the algorithm roughly parallels the real-world marriage structure where men propose to women. It seems, then, that we have naturally found a fantastic way to sort ourselves into stable matchings (i.e. marriages). But can we go further?

Of course we can. What Gale and Shapley discovered, that ended up winning them the nobel prize, is that this simple algorithm is actually biased.

5.4 Deriving the Bias

The Gale-Shapley Algorithm is quite unique in that it takes the two groups of individuals involved, and denominates a "seeker" group and a "sought" group so to speak. But how does the mechanics of seeking actually work? Well, baked into the algorithm, we see that seekers are working down from their top choice, whereas the sought are choosing the best of the options they have. The seekers control the act of proposal and thus can propose to anyone, whereas the sought only have as options the individuals who have proposed to them. We can use this disparity in options and access to prove that *in the Gale-Shapley Algorithm, the seekers (the group of individuals who apply or propose out to the other group) will always get their best possible partner amongst all possible stable matchings, and the sought (the group of individuals who are applied or proposed to) will always get their worst possible option amongst all possible stable matchings.* What does this mean in the marriage context? Well, when men propose to women, they always get their best possible wife, and women always get their worst possible husband — and we can mathematically prove it.

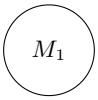
The proof of this claim is simple but subtle. First, let's establish a few key facts about the Gale-Shapley algorithm. We know that the algorithm always terminates, because the individuals in both groups always have finite preference rankings, since there is always a finite number of individuals in the other group to rank. We also know that each seeker in the Gale-Shapley algorithm will always move down their preference list as they are rejected, proposing to a lower-ranked individual after each rejection, and that each sought individual in the Gale-Shapley algorithm will always move up their preference list as they reject seekers for higher-preferred ones. Finally, we know that the Gale-Shapley algorithm will always produce stable matchings (as proved above).

We have been using the language "seeker" and "sought" to describe the denominations of the two groups, but let's go back to men and women in the marriage context, for clarity in the next few proofs. It's going to get a little complicated. Remember, the two groups are men and women now, and the

men are proposing to the women.

First, we can prove that each man will get his best possible choice by the Gale-Shapley Algorithm. We can do so by proving the impossibility of a given man getting anything better than the partner he was assigned by the algorithm.

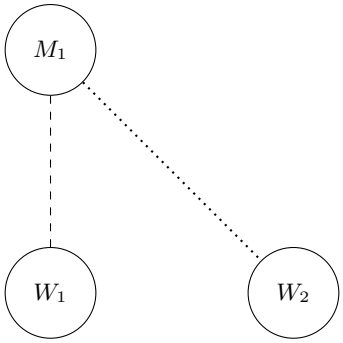
Imagine if this were the case, where a man did not get his best possible choice by the Gale-Shapley Algorithm, meaning the stable matching produced by the algorithm didn't pair this man with the highest-ranked woman he could possibly be paired with among all stable matchings for the given scenario. Let's call this man M_1.



Let's say that M_1 was matched to W_1 by the Gale-Shapley Algorithm.

Dashed lines: pairings assigned by the Gale-Shapley Algorithm as assumed by this proof.

Now imagine there exists a separate stable matching where M_1 is matched to W_2, and $W_2 \succ_{M_1} W_1$.

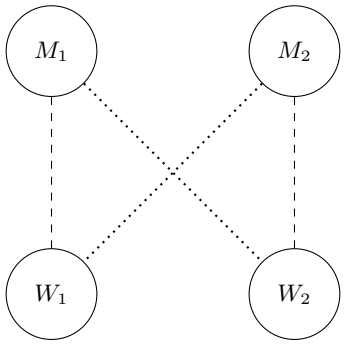


Dashed lines: pairings assigned by the Gale-Shapley Algorithm as assumed by this proof.
Dotted lines: pairings assigned in a separate matching as assumed by this proof.

In order for the algorithm matching to have been a genuine result of the Gale-Shapley Algorithm, we know that M_1 must have proposed to W_2 (because she is his top choice), yet was rejected, leaving him to propose to W_1, who he was then matched to. We therefore can make the following observation:

$$W_2 \succ_{M_1} W_1$$

We also know that in order for M_1 to have been rejected by W_2, there must have been another man she liked more, that she rejected M_1 to pair with instead. Let's call this man M_2.



Dashed lines: pairings assigned by the Gale-Shapley Algorithm ("algorithm matching") as assumed by this proof.
Dotted lines: pairings assigned in a separate matching ("hypothetical matching") as assumed by this proof.

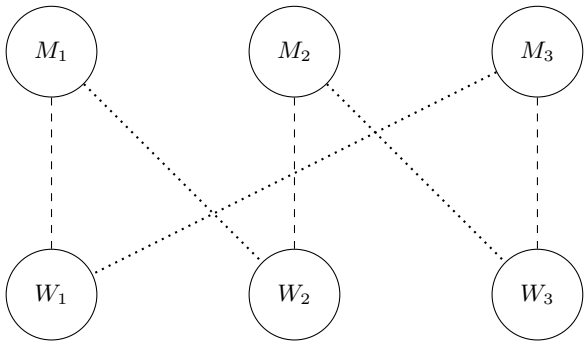
We can now reiterate our two observations from above, and add two more:

$$W_2 \succ_{M_1} W_1$$
$$M_2 \succ_{W_2} M_1$$
$$W_2 \succ_{M_2} W_1$$

We can derive the last observation by remembering that men always propose to their top choice first. If $W_1 \succ_{M_2} W_2$, then in the algorithm matching M_2 would have proposed to W_1 and M_1 would have proposed to W_2. If this were the case, the two men would have been matched to their top choices, and the resulting algorithm matching wouldn't have been the one we need: M_2 would have been paired with W_1 and M_1 would have been paired with W_2, instead of M_2 with W_2 and M_1 with W_1.

So far, we have assumed that these four individuals are the only individuals in the situation. If this were the case, in the separate matching M_2 would have to be assigned to W_1, and W_2 would have to be assigned to M_1. Because both M_2 and W_2 prefer each other over their assigned partners in this separate matching, it would not be stable, and therefore could not be the output of the Gale-Shapley Algorithm. To be paired with a better woman than she who was assigned by the algorithm, a man has to be assigned her in a way that "takes" her from another man and leaves this other man with a worse choice, making stability impossible. It is important to remember this principle as we move forward and make this situation more complex.

If there were more individuals in the situation than the four we have currently described, the same conclusion of instability can be reached but must be derived on a larger scale, since we introduce the possibility that not only can M_1 get a better partner, but M_2 can as well. If M_2 also got a better partner, we'd have to add two more individuals to our pool of subjects in this thought experiment: W_3, where $W_3 \succ_{M_2} W_2$, and M_3, her partner assigned by the algorithm.



Dashed lines: pairings assigned by the Gale-Shapley Algorithm ("dashed matching") as assumed by this proof.
Dotted lines: pairings assigned in a separate matching ("dotted matching") as assumed by this proof.

We see that M_3 must be paired with W_1 in the separate matching for W_3 to be paired with M_2. However, we know that $W_3 \succ_{M_3} W_1$ because if instead $W_1 \succ_{M_3} W_3$, M_3 would have proposed to W_1 at the start of the algorithm, M_1 would have proposed to W_2 at the start of the algorithm, and M_2 would have proposed to W_3 at the start of the algorithm – and, with each woman having only one option, all three men would have been paired with their top choice, and the algorithm would not have yielded a matching as we said it did (so the preconditions of our situation would not have been met). Therefore, in order to meet the preconditions of our situation, we must assume that $W_3 \succ_{M_3} W_1$. We also know that $M_3 \succ_{W_3} M_2$, because in order for M_2 to have proposed to his lower choice W_2 he must have been rejected by W_3, which she would have done only if she preferred M_3 over him. To recap, we now have the following observations about this setup:

$$W_3 \succ_{M_2} W_2$$
$$M_3 \succ_{W_3} M_2$$
$$W_3 \succ_{M_3} W_1$$

We can now see that M_3 and W_3 form a blocking pair in the separate matching, and it is therefore not stable. Again, the principle from above applies: that because a man can only receive a better woman than she who was assigned by the algorithm if she is "taken" from another man who prefers her more than who he will be left with, and the fact that such a woman would have had to prefer the man she is taken from, it is impossible for a matching where a man receives a higher-preferred woman than who he got from the Gale-Shapley Algorithm to be stable. Men can't do better than the woman they were assigned by the algorithm.

It is important to clarify that the preferences of M_1 and W_2 do not matter beyond how we know them to be from our previous scenario (assuming they hold constant), because they are not involved in the two pairs that create instability in the separate matching. The only reason they (as individuals) can't be truly ignored is that they are "shuffled in" with the more pertinent individuals in the algorithm matching.

Because it is impossible for a matching where a man receives a higher-preferred woman than who he got from the Gale-Shapley Algorithm to be stable, it is therefore impossible for a stable matching to exist where a man receives a higher-preferred woman than who he got from the Gale-Shapley Algorithm. This means that any given man will get his highest possible choice by the Gale-Shapley Algorithm amongst all possible stable matchings.

But there's more. Not only have we proved that a matching where a given man gets a higher-preferred woman (than the algorithm gave) cannot be stable, but we have proved that a matching where a given woman gets a lower-preferred man (than the algorithm gave) cannot be stable as well. What this means is that as well as giving men their best possible choice, **the Gale-Shapley Algorithm gives women their worst possible choice**. Looking at the four-individual example, we can see that in the separate stable matching W_2 is getting a worse choice than she was assigned by the algorithm. In the six-individual example, we can see that both W_2 and W_3 are getting worse choices than they were assigned by the algorithm. Like with the men, it is impossible for a given woman to get a worse choice than who she was assigned by the algorithm because that means one woman's man will be moved to another woman that he will prefer lower than her (M_2 in the four-person case and M_3 in the six-person case) and therefore she (W_2 in the four-person case and W_3 in the six-person case) will be left with a man she prefers less — meaning both the man and woman would prefer each other over their assigned partners, creating instability. Since it is impossible for a matching where a woman gets a lower-preferred partner (than she was assigned by the algorithm) to be stable, the Gale-Shapley Algorithm gives women their worst possible choice amongst all stable matchings.

5.5 Why is This Algorithm (and Proof) Useful?

The Gale-Shapley Algorithm appears far more often than one would expect. The most notable usage of the Gale-Shapley Algorithm and proof above was in the redesigining of the US' algorithm for assigning medical students to their hospital residencies. With both students ranking hospitals and hospitals ranking students, Gale and Shapley realized that the old system, which had hospitals proposing to students, could be flipped to give better chances to the new graduates. This specific restructuring at the hands of Gale and Shapley's conclusions is a large part of what won them the nobel prize.

This math gives us a philosophical life lesson, too: those who take the risk of seeking out, benefit more than those who comfortably wait for options to come to them. If you wish to maximize your outcome, you should be the one to seek.