Algorithm in Searching of Nash Equilibrium in Multi-player Games

by

Edward Manapov

Submitted to the Edward Manapov
Department of Computing and Mathematics
in partial fulfillment of the requirements for the degree of

Bachelor of Science in Computer Science

at the

UNIVERSITY OF DERBY

May 2016

© Edward Manapov, MMXVI. All rights reserved.

The author hereby grants to MIT permission to reproduce and to distribute publicly paper and electronic copies of this thesis document in whole or in part in any medium now known or hereafter created.

Author .....
Edward Manapov
Department of Computing and Mathematics
May 13, 2016

Certified by .....
Dr. Christophoros Mouratidis
Associate Professor
Thesis Supervisor

Accepted by .....
Dr. Tasos Stulianou
Department Committee on Graduate Thesis

Algorithm in Searching of Nash Equilibrium in Multi-player Games

by

Edward Manapov

Submitted to the Edward Manapov
Department of Computing and Mathematics
on May 13, 2016, in partial fulfillment of the
requirements for the degree of
Bachelor of Science in Computer Science

Abstract

In this dissertation, a research is made on algorithmic game theory. I designed and implemented a algorithm which performs a search of Nash equilibrium in two-player and multi-player games using brute-force technique; this involves a basic knowledge background, which is covered properly.

The source language of the algorithm is objective C, beside this pseudo-code is provided for user not in contact with object oriented languages. We test this algorithm in case of many game rounds. And, show how it performs based on the time and space complexity.

Thesis Supervisor: Dr. Christophoros Mouratidis
Title: Associate Professor

Acknowledgments

I would first like to thank my thesis advisor Dr. Christophoros Mouratidis of the department computing and mathematics at University of Derby. The door to Prof. Mouratidis office was always open whenever I ran into a trouble spot or had a question about my research or writing. He consistently allowed this paper to be my own work, but steered me in the right direction whenever he thought I needed it.

I would also like to acknowledge Dr. Tasos Stulianou of the department computing and mathematics at University of Derby as the second reader of this thesis, and I am gratefully indebted for his very valuable comments on this thesis.

Contents

THIS PAGE INTENTIONALLY LEFT BLANK

List of Figures

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 1

Introduction

1.1 Project Rationale

Game theory aims to model situations in which multiple participants interact or affect each others outcomes. This study is about to make a tool, in our occasion an algorithm, for searching Nash equilibrium in such situations. In other words, to make a helpful tool in order to minimize the time spent for searching it by hand. How much possible is to accomplish something like this; and which are the difficulties in this case. Doing such a study is worthwhile because of many reasons. Such as, an every day routine example of learning how many of our everyday actions can be calculated and modeled in a way where we can know our payoffs, wins and loses in many situations. Additionally, we can find out our best response in a situation accurately calculated. Terms like game, players, strategies and payoffs can fit in many of our daily routine acts.

1.2 Project Aims and Objectives

There are multiple methods of searching Nash equilibrium founded over the time, trivial and non-trivial.The aim of this study is to implement an algorithm in objective language C for searching Nash Equilibrium in multi-player games using a simple brute-force technique. The objectives of this study in order to achieve this aim are

to study in depth the whole theory of games and their strategies. Understand them in a very good level. Compare, and practice on. Finally construct and evaluate. Bibliographic collection, reading, evaluating problems, constructing algorithm is the main outline of study design and methods.

Chapter 2

Literature Review

2.1 General Game Theory

Game theory is "the investigation of mathematical models of conflict and cooperation between intelligent rational decision-makers (Myerson, Roger B., 1991)[1]. Game theory is mainly utilized in economics, political science, and psychology, as well as logic, computer science, biology and poker (Christopher Chabris, 2013)[2]. Originally, it tended to zero-sum games, in which one individual's increases result in losses for alternate participants. Today, game theory applies to an extensive variety of behavioral relations, and is presently an umbrella term for the investigation of logical decision making in humans, animals, and computers.

Modern game theory started with the thought in regards to the presence of mixed-strategy equilibrium in two-person zero-sum games and its proof by John Von Neumann (1928)[3]. Von Neumann's unique proof utilized Brouwer fixed-point theorem on consistent mappings into conservative convex sets, which turned into a standard method in game theory and mathematical economics. His paper was trailed by the 1944 book Theory of Games and Economic Behavior, co-composed with Oskar Morgenstern (1944)[4], which considered helpful recreations of a several players. The second release of this book gave a proverbial theory of expected utility, which permitted mathematical analysts and financial experts to treat decision-making under uncertainty.

This theory was created widely in the 1950s by numerous researchers. Game theory was later explicitly connected to biology in the 1970s, although comparative developments do a reversal at any rate similarly as the 1930s. Game theory has been generally perceived as an imperative tool in numerous fields. With the Nobel Memorial Prize in Economic Sciences going to game theorist Jean Tirole in 2014[5], eleven game-theorists have now won the economics Nobel Prize. John Maynard Smith (1999)[6] was honored the Crafoord Prize for his use of game theory to biology.

2.2 Games

In this game theory thesis we start the present part by giving some established games. First of all, we engage the reader's natural idea of a "game"; this notion is formally defined further. For a more inside and out examination of game theory we allude the readers to books on game theory such as Fudenberg and Tirole (1991)[7], Mas-Colell, Whinston, and Green (1995)[8], or Osborne furthermore, Rubinstein (1994)[9]. In simple words game theory intends to model situations in which various participants interact or influence each other's results. We begin by portraying what is maybe the most understood and very much contemplated game.

2.2.1 The Prisoners Dilemma

Two prisoners are on trial for a crime and every one faces a decision of admitting to the wrongdoing or remaining silent. On the off chance that they both remain silent, the authorities won't have the capacity to prove charges against them what's more, they will both serve a short jail term, say 2 years, for minor offenses. On the off chance that one of them admits, his term will be reduced to 1 year and he will be utilized as an observer against the other, who thus will get a sentence of 5 years. Finally, if they both admit, they both will get a little break for collaborating with the authorities and will need to serve jail sentences of 4 years each, instead of 5. Obviously, there are four aggregate results relying upon the decisions made by each of the two prisoners. We can compactly summarize the costs acquired in these four results through the

		Player 2	
		$Confess$	$Silent$
Player 1	$Confess$	$(4, 4)$	$(1, 5)$
	$Silent$	$(5, 1)$	$(2, 2)$

Figure 2-1: Prisoner's Dilemma's payoff matrix

accompanying two-by-two matrix.

Each of the two players "Player 1" and "Player 2" has two conceivable strategies(decisions) to "admit" or to remain "silent." The two strategies of Player 1 relate to the two rows and the two strategies of player 2 relate to the two columns of the matrix. Inside entries of the matrix are costs acquired by the players in every circumstance (left variable for the row player and the right for column). Such a matrix is known as a cost matrix since it contains the cost brought by the players for every decision of their strategies. The main stable arrangement in this diversion is that both prisoners confess; in each of the other three cases, no less than one of the players can change from "silent" to "confess" and improve his own particular result. Then again, a greatly better result for both players happens when neither of them admits. Unlike, this is most certainly not a stable solution, regardless of the fact that it is carefully planned out since each of the players would be enticed to defect and consequently serve less time. This game situation modeled above arises naturally in many different situations. See bellow for more games.

2.2.2 ISP Routing Game

Consider Internet Service Providers (ISPs) that need to send traffic each other. In routing that begins in one ISP with destination in an alternate ISP, the routing decision made by the first ISP additionally influences the heap at destination ISP. We will see here how this circumstance offers ascend to precisely the Prisoner's dilemma portrayed previously. Consider two ISPs, as in Figure 2.2, each having its own different network. The two networks can trade traffic by means of two transit points, called peering, which we will call A and B.

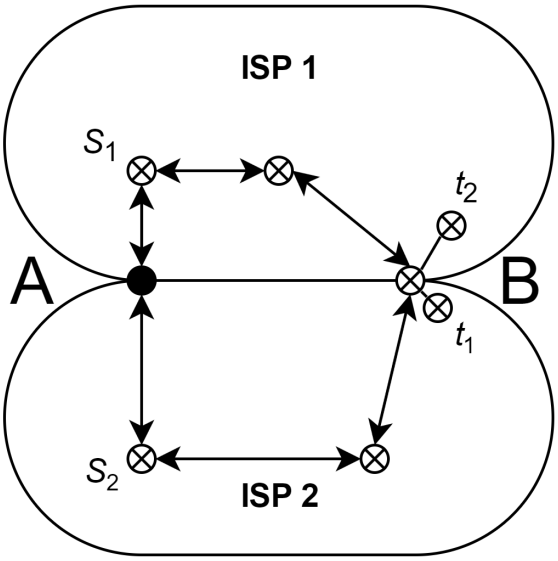

Figure 2-2: The ISP routing problem

In the figure we likewise have two origindestination sets s_i and t_i each crossing between the domains. Assume that ISP 1 needs to send traffic from point s_1 in his own domain to point t_1 in second ISP's domain. ISP 1 has two options for sending its traffic, relating to the two peering points. ISPs normally behave egotistically and attempt to minimize their own particular expenses, and send traffic to the nearest peering point, as the ISP with destination hub must route the traffic, regardless of where it enters its domain. Peering point A is nearer, utilizing this point ISP 1 acquires a cost of 1 unit (in sending traffic along 1 edge), while in the event that it utilizes the farther peering point B, it acquires a cost of 2. Note that the more distant peering point B is more directly on route to the destination t_1, and thus steering through B results in shorter general way. The length of way through A is 4 while B is 2, as the destination is near B. The situation depicted for ISP 1 routing traffic from s_1 to t_1 is as it were analogous to a prisoner's choices in the Dilemma: there are two, one is better from an egotistical viewpoint ("admit" or route through peering point A),but hurts the other player. To make our routing game identical to the Prisoner's Situation, expect that symmetrically the second ISP needs to send traffic from point s_2 in his domain to point t_2 in the first ISP's domain. The two decisions of the two ISPs lead to a game with cost matrix indistinguishable to the matrix above with A

relating to "admit" and B corresponding to remaining "silent."

2.3 Multi-player Versions

In this thesis we will be most concerned with situations where multiple participants interact with each other, furthermore, such situations are normally demonstrated by games that include numerous players: there are thousands of ISPs, and a huge number of traffic streams to be steered. We will give two case of such recreations, initial a multi-player form of the Prisoner's Dilemma that we will express as far as a pollution game. At that point we will examine the well-known game of Tragedy of the Commons.

2.3.1 Pollution Game

This game is the augmentation of Prisoner's Dilemma to the instance of many players. The issues displayed by this game emerge in numerous contexts; here we will talk about it in the context of pollution control. Assume that there are n countries in this game. For a straightforward model of this situation, suppose that every country confronts the decision of either passing legislation to control pollution or not. Accept that pollution control has an expense of 3 for the country, however every country that pollutes adds 1 to the cost of all countries. The cost of controlling pollution, which is 3, is impressively bigger than the expense of 1 a country pays for being socially irresponsible. Assume that k countries pick not to control pollution. Plainly, the cost acquired by each of these countries is k. Then again, the cost brought about by the remaining $n - k$ nations is $k + 3$ each, since they need to pay the additional taken a toll for their own pollution control. The main stable solution is the one in which no country controls pollution, having an expense of n for every country. Conversely, on the off chance that they all had controlled pollution, the cost would have been 3 for each country.

The games we have seen so far offer the feature that there is an unique ideal "selfish" methodology for every player, free of what other players do. Regardless what

strategy the opponent plays, every player is in an ideal situation playing his or her selfish strategy. Next, we will see a game where the players' ideal egotistical strategies depend on what alternate play.

2.3.2 Tragedy of the Commons

We will portray this game in the connection of sharing bandwidth. Suppose that n players each might want to have part of a shared asset. For instance, every player needs to send data along a shared channel of known maximum capacity, say 1. In this game every player will have a limitless sets of strategies, player i's strategy is to send x_i units of flow along the channel for some value $x_i \in [0,1]$.

Expect that each player might want to have an expansive fraction of the bandwidth, but suppose also that the quality of the channel deteriorates with the total bandwidth utilized. We will depict this game by a straightforward model, utilizing an benefit or payoff function for each set of strategies. If the total bandwidth $\sum_j x_j$ exceeds the channel limit, no player gets any advantage. On the off chance that $\sum_j x_j < 1$ then the quality for player i is $x_i(1 - \sum_j x_j)$. This displays precisely the sort of exchange we had at the top of the priority list: the benefit for a player deteriorates as the aggregate assigned bandwidth increases, but it increases with his own share, to a limited point.

To comprehend what stable strategies are for a player, let us focus on i, and assume that $t = \sum_{j \neq i} x_j < 1$ flow is sent by every single other player. Presently player i faces a straightforward optimization issue for selecting his flow sum: sending x flow results in an advantage of $x(1 - t - x)$. Utilizing basic analytics, we get that the ideal answer for player i is $x = (1 - t)/2$. An arrangement of strategies is stable on the off chance that all players are playing their ideal selfish strategy, given the strategies of every single other player. For this case, implies $x_i = (1 \sum_{j \neq i} x_j)/2$ for all i, which has an extraordinary solution in $x_i = 1/(n + 1)$ for all i.

Why is this solution a tragedy? The total value of the solution is extremely low. The value for player i is $x_i(1 \sum_{j \neq i} x_j) = 1/(n + 1)^2$, and the whole of the values over all payers is then $n/(n + 1)^2 \approx 1/n$. Conversely, if the aggregate bandwidth utilized is $\sum_i x_i = 1/2$ then the total value is $1/4$, around $n/4$ times greater. In this

game the n users sharing the common asset overuse it so that the total estimation of the shared asset diminishes drastically. The pollution game above has a comparable impact, where the common asset of the environment is overused by the n players expanding the cost from 3 to n for each player.

2.4 Coordination Games

In our next case, there will be various results that can be stable. This game is a case of supposed like "coordination game." A straightforward game includes two players picking between two alternatives, needing to pick the same.

2.4.1 Battle of the Sexes

Consider that two players, a man and a woman, are choosing how to spend their night. They both consider two conceivable outcomes: heading off to a football game or setting off to a basketball game. The man lean towards football and the woman lean basketball, yet they both might want to spend the night together instead of separately. Here we express the players' inclinations again through payoffs as follows.

		Man	
		$Football$	$Basketball$
Woman	$Football$	$(5, 6)$	$(1, 1)$
	$Basketball$	$(2, 2)$	$(6, 5)$

Figure 2-3: Battle of the Sexes payoff matrix

Clearly, the two solutions where the two players pick diverse games are not stable, for every situation, both of the two players can enhance their payoff by exchanging their activity. Then again, the two remaining choices, both going to the same game, whether it is football or basketball, are both stable solutions; the woman inclines toward the first and man lean towards the second.

Coordination games additionally emerge naturally in numerous contexts. Here we give an illustration of a coordination game in the context of routing to avoid a

congestion. The great outcomes in the Battle of Sexes were to go the same game. In contrast, in the routing game, great outcomes will require on various paths to stay away from congestion. Subsequently, this will be an "anti-coordination" diversion.

2.4.2 Routing Congestion Game

Assume that two traffic streams originate at proxy node O, and should be steered to the whole rest network, as appeared in Figure 2.4. Assume that node O is associated to the rest of network by means of connection points A and B, where A will be somewhat closer than B. Be that as it may, both connection points get effortlessly congested, so sending both streams through the same connection point causes additional delay. Great outcomes in this game will be for the two players to "coordinate" and send their traffic through various connection points.

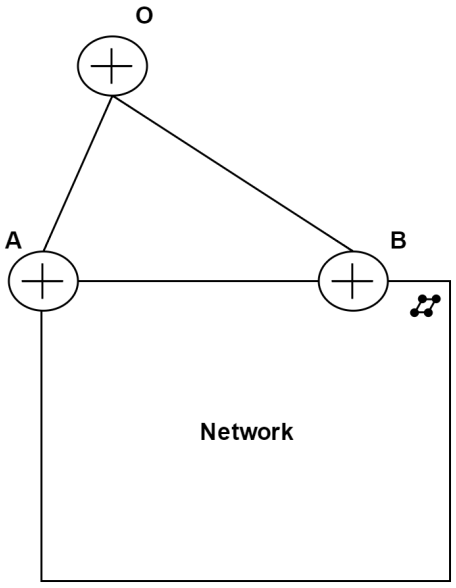

Figure 2-4: Routing Congestion Game Network Graph

		Traffic 1	
		A	B
Traffic 2	A	$(5,5)$	$(1,2)$
	B	$(2,1)$	$(6,6)$

We demonstrate this situation through a game with the two streams as players. Each player has two accessible strategies routing through A or steering B prompting

Figure 2-5: Routing Congestion Game Payoff Matrix

four total possibilities. The matrix of Figure 2.5 expresses the costs to players in terms of delays relying upon their routing decisions.

2.5 Mixed Strategies

In the games we considered in this way, there were outcomes that stable in the sense that none of players would need to exclusively deviate from such an outcome. Not all games have such stable solutions, as represented by the accompanying case.

2.5.1 Matching Pennies

Two payers, each having a penny, are inquired to look over among two strategies, heads (H) and tails (T). The row player wins if the two pennies match, while the column player wins on the off chance that they don't match, as appeared by the accompanying payoff matrix, where 1 shows win and 1 demonstrated misfortune.

		2	
		$Heads$	$Tails$
1	$Heads$	$(1,-1)$	$(-1,1)$
	$Tails$	$(-1,1)$	$(1,-1)$

Figure 2-6: Matching Pennies Game Payoff Matrix

One can see this game as a variation of the routing congestion game in which the column player is keen on getting good service, consequently might want the two players to pick diverse routes, while the row player is intrigued just in disturbing the column player's service by attempting to pick the same route. It is easy to see that this game has no stable solution. Rather, it appears to be best for the players to randomize so as to impede the strategy of other player.

2.6 Strategies, Costs and Payoffs

We have given case of games and talked about costs, payoffs, and strategies in an informal way. So, next those definitions will be discussed more formally. The games we considered above were every one of the *one-shot simultaneous move games*, in that all players at the same time picked an action from their set of conceivable strategies.

2.6.1 Terminology and Notation

Formally,such a game comprises of a set n of players, {1,2,...,n}. Each player i has his own set of conceivable strategies, say S_i. To play the game, every player i chooses a system $s_i \in S_i$. We will utilize $s = (s_1,...,s_n)$ to mean the vector of strategies chose by the players and $S = x_i S_i$ to indicate the set of all conceivable routes in which players can pick strategies.

The vector of strategies $s \in S$ chose by the players denote the result for every player; by and large, the outcome will be distinctive for various players. To indicate the game, we have to give, for every player, an preference ordering on these results by giving a complete, transitive, reflexive parallel connection on the set of all strategy vectors S; given two components of S, the connection for player i says which of these two results i feebly inclines toward; we say that i weakly prefers S_1 to S_2 in the event that i either prefers S_1 to S_2 or considers them as similarly great outcomes.

The least complex approach to indicate preferences is by assigning, for every player, a value to each outcome. In some games it will be natural to think about the values as the payoffs to players and in others as the costs caused by players. We will signify these functions by $u_i : S \to \mathbf{R}$ and $c_i : S \to \mathbf{R}$, respectively. Obviously, costs and payoffs can be utilized conversely, since $u_i(s) = -c_i(s)$.

On the off chance that we had defined, for each player i, u_i to be essentially a function of s_i , the strategy picked by player i, rather than s, the strategies picked by all n players, then we would have acquired n independent optimization issues. Watch the pivotal distinction among this and a game, in a game, the payoff of each player depends not just on his own strategy, additionally on the strategies picked by every

other player.

2.6.2 Standard Specified Form

To build up an algorithmic theory of games, we have to talk about how a game is specified. One choice is to expressly list all conceivable strategies, and the preferences or utilities of all players. Expressing games in this structure with a cost or utility function is called the standard form or matrix form of a game. It is extremely advantageous to define games in this way when there are just 2 players and the players have just a couple of strategies. We have utilized this structure as a part of the past section for defining the Prisoner's Dilemma and Battle of the Sexes.

Nonetheless, for most games we need to consider, this express representation is exponential estimated in the regular portrayal of the game, perhaps greater or even limitless. Most games we need to consider have numerous players, e.g., the movement streams or the numerous ISPs controlling such streams. As a precedent, consider the pollution games from Subsection 2.3.1, where we have n players, each with two possible strategies. There are 2^n strategy vectors, so the explicit representation of the game requires assigning values to each of these 2^n strategies. The extent of input expected to portray the game is much smaller than 2^n, thus this explicit representation is exponentially bigger than the description of game.

Another reason that explicit representation of the payoffs can be exponentially substantial is that players can have numerous strategies in the natural size of the game. This happens in routing games, since the strategy space of every player comprises of all possible paths from source to destination in the network. Form of the Tragedy Commons, we talked about in Section 2.3.2 players have infinite strategy sets, since any bandwidth $x \in [0, 1]$ is a conceivable strategy.

Such exponential and super descriptions can once in a while be avoided. For illustration, the payoff may rely on upon the quantity of players selecting a given strategy, as opposed to the exact subset, similar to the case for the pollution game. Another probability for conservative representation is the point at which the payoff of a player may rely on upon the strategies picked by just a couple of different players,

not all participants. Games with such locality properties can be find more analytically on the web.

2.7 Solution Concepts

In this segment we will present basic solution concepts that can be utilized to study the sorts of games we depicted in the previous segment. Specifically, we will formalize the idea of stability that we informally utilized as a part of solutions for some of the games.

2.7.1 Dominant Strategy Solution

The Prisoner's Dilemma and the Pollution Game share an extremely unique property: in each of these games, every player has a one of a kind best strategy, autonomous of the strategies played by alternate players. We say that a game has a dominant strategy solution in the case that it has this property.

All the more formally, for a strategy vector $s \in S$ we utilize s_i to mean the strategy played by player i and s_{-i} to mean the (n - 1) dimensional vector of the strategies played by all different players. Review that we utilized $u_i(s)$ to mean the utility caused by player i. We will likewise utilize the notation $u_i(s_i, s_{-i})$ when it is more helpful. Utilizing this notation,a strategy vector $s \in S$ is a dominant strategy vector, if for every player i, and each interchange strategy vector $s' \in S$, we have that

$$u_i(s_i, s'_{-i}) \geq u_i(s'_i, s'_{-i}).$$

Notice that a dominant strategy solution may not give an optimal payoff to any of the players. This was the situation in both the Prisoner's Dilemma and the Pollution Game, where it is conceivable to enhance the payoffs of all players at the same time.

2.7.2 Pure Nash Equilibrium

Since games once in a while have dominant strategy solutions, we have to look for a less stringent furthermore, all the more widely applicable solution concept. An

alluring game theoretic is one in which singular players act as per their motivations, boosting their own particular payoff. This idea is best caught by the notion of a Nash equilibrium, which, in spite of its short comings, mentioned below, has rose as the central solution concept in game theory, with amazingly diverse applications. The Nash equilibrium catches the idea of a stable solution, talked about in Section 2.2 and used in the Tragedy of Commons and the Battle of the Sexes a solution from which no single player can individually improve his or her welfare by deviating (Noam Nisan, 2007)[10].

A strategy vector $s \in S$ is said to be a Nash equilibrium if for all players i and each exchange strategy $s_i' \in S_i$, we have that

$$u_i(s_i,s_{-i}) \geq u_i(s_i',\, s_{-i}).$$

As such, no player i can change his picked strategy from s_i to s_i' furthermore, improve his payoff, expecting that every other player adhere to the strategies they have picked in s. Watch that such a solution is self-enforcing as in once the players are playing such a solution, it is in each player's best interest to adhere this or her strategy.

Clearly,a dominant strategy solution is a Nash equilibrium. Moreover,if the solution is entirely dominating (i.e., changing to it generally improves the outcome), it is likewise the unique Nash equilibrium. Be that as it may, Nash equilibrium may not be unique. For illustration, coordination games have numerous equilibrium.

We definitely realize that Nash equilibrium may not be optimal for the players, since dominant strategy solutions are Nash equilibrium. For games with numerous Nash equilibrium, distinctive can have many different payoffs for the players. For instance, by a little change to the payoff matrix, we can adjust the Battle of Sexes game so that despite everything it has two stable solutions, the ones in which both players go to the same movement, nonetheless, both determine a much higher utility from one of these solutions.

The presence of numerous Nash equilibrium makes this solution idea less convincing as a forecast of what players will do: which equilibrium should we expect them to

play? What's more, with free play, by what method will they know which equilibrium they are expected to coordinate on? In any case, a Nash equilibrium is stable, once proposed, the players would prefer not to exclusively deviate.

2.7.3 Mixed Nash Equilibrium

The Nash equilibrium we have considered so far are called pure strategy equilibrium, since every player deterministically plays his picked strategy. As represented by the Matching Pennies game, a need not have any pure strategy Nash equilibrium. In any case, if in the matching pennies game, the players are permitted to randomize and every player picks each of his two strategies with likelihood 1/2, then we acquire a stable solution it might be said. The reason is that the normal payoff of every player now is 0 and not one or the other can enhance this by picking an alternate randomization.

At the point when players select strategies at random, we have to see how they assess the random outcome. Would a player incline toward a decision that prompts a little positive utility with high probability, however with a small probability prompts an extensive negative utility? On the other hand, is it better to have a little misfortune with high probability, and an expansive increase with small probability? For the idea of mixed Nash equilibrium, we will suppose that players are risk neutral, that is, they act to amplify the expected payoff.

To characterize such randomized strategies formally, let us upgrade the decisions of players so every one can pick a probability dispersion over his set of conceivable strategies. Such a decision is known as a mixed strategy. We expect that players freely choose strategies utilizing the probability dispersion. The free random decisions of players leads to a probability dispersion of strategy vectors s. Nash (1951)[11] demonstrated that under this extension, each game with a limited number of players, each having a limited set of strategies, has a Nash equilibrium.

Theorem 2.7.3: *Any game with a limited set of players and limited set of strategies has a Nash equilibrium of mixed strategies.*

2.7.4 Correlated Equilibrium

A further unwinding of the Nash equilibrium idea was presented by Aumann (1959)[14], called correlated equilibrium. The accompanying straightforward example pleasantly illustrates this idea.

Traffic lights

The game we consider is when two players drive up to the same crossing point in the same time. In the event that both endeavor to cross, the outcome is a deadly car crash. The game can be demonstrated by a payoff matrix where crossing effectively has a outcome of 1, not crossing pays 0, while a mishap costs -100.

		Player 2	
		$Cross$	$Stop$
Player 1	$Cross$	$(-100, -100)$	$(1, 0)$
	$Stop$	$(0, 1)$	$(0, 0)$

Figure 2-7: Traffic Lights Game Payoff Matrix

This game has three Nash equilibrium, two compare to letting one and only car cross, the third is a mixed equilibrium where both players cross with an greatly little likelihood $e = 1/101$, and with e^2 likelihood they crash. The initial two equilibrium have a payoff of 1. The last one is all the more reasonable, yet has low expected payoff ($\approx$0.0001), furthermore has a positive possibility of a car crash.

In a Nash equilibrium, players pick their strategies autonomously. In contrast, a correlated equilibrium facilitator can pick strategies for both players, nonetheless, the picked strategies must be stable: we require that the every player discover it in his or her enthusiasm to take after the prescribed strategy. For instance, in a correlated equilibrium the facilitator can randomly let one of the two players cross with any likelihood. The player who is advised to stop has 0 result, however he realizes that endeavoring to cross will bring about a traffic accident.

Formally, this idea accept an external correlation device, for example, a trusted game facilitator, or a few other physical source. A correlated equilibrium is a likelihood

conveyance over strategy vectors $s \in x_i S_i$. Let $p(s)$ mean the probability of strategy vectors, where we will likewise utilize the notation $p(s) = p(s_i, s_{-i})$ when discussing a player i. The circulation is a correlated equilibrium if for all player i and all strategies s_i,$s_i' \in S_i$ we have the imbalance.

$$\sum_{s_{-i}} p(s_i, s_{-i}) u_i(s_i, s_{-i}) \geq \sum_{s_{-i}} p(s_i, s_{-i}) u_i(s_i', s_{-i}).$$

In words, if player i gets a recommended strategy s_i, the normal benefit of the player can't be expanded by changing to an alternate strategy $s_i' \in S_i$. Nash equilibrium are exceptional instances of correlated equilibrium, where the conveyance over S is the product of autonomous circulations for every player. In any case, ion correlation permits a wealthier set of equilibrium.

2.7.5 Games with No Nash Equilibrium

Both suppositions in the theorem the limited set of players and limited strategy sets are critical: recreations with a boundless number of players, or games with a limited number of players who have admittance to an interminable strategy set might not have Nash equilibrium A straightforward case of this emerges in the accompanying pricing game.

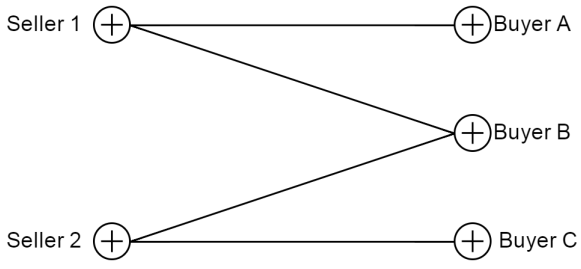

Figure 2-8: Pricing Game Graph

Pricing Game

Assume two players offer a product to three possible buyers, as appeared in Figure 2.8. Every buyers needs to purchase one unit of the product. Also, assume that seller 1 and 2 are selling identical to buyers A, B and C. So, Buyers A and C have access to one dealer just, in particular 1 and 2, separately. In spite of, buyer B can purchase the item from any of the two sellers. Each of the three buyers have a financial cost of 1, or have maximum worth 1 for the product, i.e., won't purchase the item if the cost is above 1. The sellers play a pricing game, they each name a price p_i in the interim $[0, 1]$. Buyers A and C buy from sellers 1 and 2, separately. Moreover, B buys from the less expensive point. To completely indicate the game, we need to set a rule for breaking ties. Give us a chance to say that if both sellers have the same price, B buys from point 1. For effortlessness, we accept no production costs, so the wage of a seller is the sum of the costs at which they sold.

Presently, one strategy for every dealer is to set a cost of $p_i = 1$, and ensure an income of 1 from the buyer who does not have a decision of buying point. Then again, they can additionally attempt to vie for buyer B. Nonetheless, by the rules of this game they are most certainly not permitted to price-discriminate; i.e., they can't sell the item to the two buyers at various costs. In this games, every player has uncountably numerous accessible strategies, i.e., all numbers in the interim $[0, 1]$. Concluding that this game does not have a Nash equilibrium, regardless of the possibility that players are permitted to utilize mixed strategies.

To see that no pure strategy equilibrium exists, take note of if $p_i > 1/2$, player 2 will marginally undermine the price, set it at $1/2 < p_2 < p_1$, and have income of something beyond than 1, and after that thusly player 1 will undermine player 2, and so forth. So we can't have $p_1 > 1/2$ in an equilibrium. If $p_1 \leq 1/2$, the unique best reaction for player 2 is to set $p_2 = 1$. Yet, then player 1 will increase his price, so $p_1 \leq 1/2$ likewise does not prompt an equilibrium. It is somewhat harder to contend that there is additionally no mixed strategy equilibrium in this game.

2.8 Finding Equilibrium in Games

Computing a Nash equilibrium, given an game in normal structure, is a principal issue for Algorithmic Game Theory. The issue is basically combinatorial, and on account of two or n-players it can be solved by many techniques. Such as, linear programming, backtracking technique and in general many algorithmic type techniques. More analyzing can find on book "Algorithmic Game Theory" (Noam Nisan, 2007)[10].

In this segment we consider two firmly related issues: how simple is it to find an equilibrium, and does "natural game play" lead the players to a equilibrium? In a perfect world, a immaculate solution concept is one which is computationally simple to discover, furthermore to find by players playing independently.

2.8.1 General Complexity

A fundamental problem on the active interface of algorithmic game theory is to define the computational complexity of computing Nash equilibrium. For example, the most known problem in some kind of games like non-cooperative games is the Nash equilibrium. In other words, a stable point among n strategic players from which none the choose to deviate. The brute-force algorithm is the main topic of complexity discussion in this thesis.

At this point, an overview of what is complexity provided. The complexity of finding a Nash equilibrium in will be discussed further(see chapter 4). We then analyze a known scheme of a two-player zero-sum game. Also, show how can a Nash equilibrium can be found using linear programming. The outcome is, that even two-player zero-sum games have different character approach in contrast with game of three and more players. As a precedent, two-player games where payoffs are rational numbers always come to solution with rational probabilities, and this not right for games with more than two players.

NP-completeness, the standard way of establishing interactivity of individual problems, does not seen the right tool for studying the complexity of Nash equilibrium. Christos Papadimitriou(2005)[12] gave an overview talk at the NIPS* conference in

2005 and mentioned the new result by Chen and Deng[13] showing that, given the payoff matrix, the complexity of computing a 2-Player Nash Equilibrium is PPAD-complete.

**NIPS is an AI(Artificial Intelligence) conference.*

We will concentrate on pure Nash equilibrium searching complexity. And here is only provided a short overview of what is mentioned on discussion section. In our occasion, the complexity refers to the algorithm constructed for solving this problem. Precisely, step by analysis will be on the execution time and storage of the algorithm. Some specific "could be" and "could not be" will discussed based on the efficiency of algorithm. Also providing a mind map of how difficult is to accomplish a simple brute-force algorithm and it's complexity over the big O notation.

2.8.2 Two-Person Zero-Sum Games

Here we consider two-player zero-sum games in more detail. A two-player game is a zero-sum game if the total of payoffs of the two players is zero for any decision of strategies. For such games it is sufficient to give the payoffs of the line player. Let A be the matrix of these payoffs, representing the winnings of the line player and the loss of column player (Noam Nisan, 2007)[10].

Review from Theorem 2.7.3 that a Nash equilibrium of mixed strategies dependably exists. We will utilize this to demonstrate that a equilibrium can be discovered using linear programming. Consider a couple of likelihood distributions p^* and p^* for the row and column players that shape a Nash equilibrium. The expected value paid by the column player to the row player can be assumed as $u^* = p^* A q^*$, on the off chance that we think of p^* as a row vector and q^* as a column vector.

A Nash equilibrium has the property that regardless of the possibility that players know the strategies played by alternate, they can't be in an ideal situation by deviating. In light of this, consider a strategy p for the row player. Normal payoffs for various strategies of the column player will be pA. When p is known, the column player will need to minimize his misfortune, and play strategies that relate to the minimum entries in pA. So the best freely declared strategy for the row player is to maximize this minimum value. Best public strategy can be found by solving the accompanying

linear program:

$$u_{r} = max\ u$$
$$p \geq 0$$

$$\sum_{i} p_{i} = 1$$

$$(pA)_{j} \geq u \text{ for all } j,$$

We assume that $(pA)_{j}$ to mean the jth entry of the vector pA. The value u_{r} is the maximum safe value for the row player. Although, the maximum value player can guarantee victory by playing a mixed strategy p that will be known to the column player

How does u_r and the Nash value u^* compare? Clearly $u_r \leq u^*$, since the row player, can promise to win u_r, so should win at least this much in any equilibrium. On the other hand, an equilibrium is a procedure that is stable regardless of the fact that known to the other player, so it must be the situation that the column player is selecting the columns with minimum value p^* A, so we should have $u^* \leq u_r$, and hence $u_r = u^*$.

Essentially, we can set up the undifferentiated from linear program to get the value u_c,the section player's minimum safe esteem, the minimum misfortune column player can guarantee by playing a mixed strategy q that will be known to the row player:

$$u_{c} = min\ u$$
$$q \geq 0$$

$$\sum_{j} q_{j} = 1$$

$$(Aq)_{i} \leq u \text{ for all } i,$$

where we utilize $(Aq)_i$ to signify the ith point of the vector Aq. We can contend that $u^* = u_c$ additionally holds. Consequently we get that $u_c = u_r$, the row players' most extreme ensured win is the same as the column players' minimum ensured misfortune. This will suggest that the ideal solution for this pair of linear programs shape a Nash equilibrium.

Theorem 2.8.2 *Optimum solution for the above pair of linear programs give likelihood distributions that frame a Nash equilibrium of the two-person zero-sum game.*

Proof: Let p and q mean optimum solutions for the two linear projects. We contended over that $u_c = u_r$. On the off chance that the players play this pair of strategies, then the row player can't build his win, as the column is ensured by his system not to lose more than u_c. Essentially, the column player can't diminish his misfortune, as the row is ensured to win u_r by his strategy. So the set of strategies is at equilibrium.

User more acquainted with linear programming will see that the two linear programs above are duals of each other. We set up that $u_r = u_c$ utilizing the presence of a Nash equilibrium from Theorem 2.7.3 Linear programming duality additionally suggests that the two qualities u_r and u_c are equivalent. When we know the qualities are equivalent, the confirmation of Theorem 2.8.2 demonstrates that the ideal solution shape a Nash equilibrium, so linear programming duality yields a proof that Nash equilibrium exists in the exceptional case of zero-sum two-person diversions.

2.8.3 Best Response in Games

It would be attractive for a solution idea to fulfill a more grounded condition than essentially being polynomial processable: the reality of situation should prove that normal game playing procedures rapidly lead players to either discover the equilibrium or if nothing else converge to an equilibrium in the limit.

Perhaps the most characteristic "game playing" strategy is the accompanying "best response." Consider a vector s, and a player i. Utilizing the strategy vector s player i

gets the value or utility $u_i(s)$. Changing the technique s_i to some other strategy s'_i $\in S_i$ the player can change his utility to $u_i(s'_i, s_{-i})$, accepting that every single other player adhere to their techniques in $_{s_{-i}}$. We say that a change from system s_i to s'_i is an enhancing response for player i if $u_i(s'_i, s_{-i}) > u_i(s)$ and best response if s'_i boosts the players' utility $max_{s'_i \in S_i}\ u_i(s'_i, s_{-i})$. Playing a game by over and again permitting some player to make an enhancing or a best response move is maybe the most regular game play.

In some recreations, for example, the Prisoner's Dilemma or the Coordination Game, this dynamic leads the players to a Nash equilibrium in a couple steps. In the Tragedy of Commons the players won't achieve the equilibrium in a limited number of steps, be that as it may, the strategy vector will converge to the equilibrium. In other games, the play may cycle, and not converge. A straightforward illustration is matching pennies, where the payers will cycle through the 4 conceivable strategy vectors on the off chance that they exchange making best response moves. While this game play does not locate a pure equilibrium, as none exists in a few sense we can in any case say that best response converges to the equilibrium. Average payoff for the two players unites to 0, which is the result at equilibrium; and even the frequencies at which the 4 conceivable strategy vectors are played converge to the probabilities in equilibrium, 1/4 each.

2.8.4 Conclusion

Finally, theoretically the whole concept of game theory is analyzed. On the other hand, practically, the "whole" Stheory of games is enormous. This literature review is conducted after a research based on what is enough for someone not in contact with the algorithmic game theory to get easily in and understand, then evaluate something similar to our aim. Which is the evaluation of an algorithm searching the now, "known" Nash equilibrium.

Some of the sections are a crucial component of the discussion that will follow. Providing paradigms where necessary, readers might already have in mind random games and solutions. Pure Nash equilibrium is for sure the basic component of the

algorithm. After that, the solution of a simple exercise by your hand could be very helpful, looking at how you personally can define a finite algorithm for searching Nash equilibrium in a two-player game using a matrix. Additionally, if it seems unknown or difficult, use the notation we mentioned after looking on how other people did it.

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 3

Research Methodology

3.1 Overview

In this chapter, a series of the major components of the methodologies used to build the algorithm for searching Nash equilibrium is provided. According to the hypothesis we mentioned in project rationale. Including research strategy, design, methods, sampling strategies and data analysis techniques. Which all of these components may have ethical implications. After search of the implication in each component. Ethics section is providing a general discussion and identically in which cases we have or do not have ethical issues in our case.

3.2 Research Strategy

The research strategy followed in this study consists of a mixed methods research design. Both qualitative and quantitative research methods are used. I could design and develop the algorithm using many different techniques. Like, linear programming, back tracking or even simple tracking technique. However, i used brute-force technique because it consists of a simple mechanism for searching. Simple implementation. Logical overview. And much more because it is running in quadratic $O(n^2)$ complexity. Moreover, in the case of small number of game plays. Likewise, the algorithm can reach high accuracy and execution time even in a games played n times. Design

thinking follows the conditions of: empathize, define, ideate, prototype and finally test. I chose to develop the algorithm in objective c because of the fast binary search. Although, this is an additional option that i did not implemented. I can test the searching algorithm in many ways, like different conditions. For example, in many cases with different game types, strategies and payoffs but the algorithm is constructed in a way that this is impossible as some of these setting are randomly generated and others just can not be defined Although, I will test it in a way of time complexity. Beside this, i will use a third party software to compare the results in cases of multiple games played. The whole set up will be analyzed and discussed in chapter 4 and 5 accordingly.

3.2.1 Quantitative Research Design

In contrast with qualitative research design, the more well structured and defined utilities of quantitative research design gives the chance to researchers plan the biggest part of the process before it even starts. In the case that has started it gives you the chance to stay much closely to these plans. From the point of view the ethics part, it is kind of easy job to define the challenges you may face. And step by step plan the overcome of this challenge. In this study quantitative research design is used because of the fact that algorithmic game theory uses quantitative data. For example, notation and terminology mentioned in chapter 2 can be counted and moreover calculated in a quantitative way. Jargons used like outcomes, strategies and payoffs could be qualitative assuming them like jargons and theory's standards. But assuming them as a number gives us the option to call them quantitative. Most of the data, is. Because we have to match with math calculations. And algorithm complexity. Which is tested over quantitative again components.

3.2.2 Qualitative Research Design

On the other hand, qualitative research design seems to be more evolutionary when compared to quantitative design. As a precedent, data gathering during the research

can easily influence the choice of what methods should the researcher use. As a result, the ethical issues that could be discovered is only after all this data collection and research. This strategy's cost is that it makes harder to understand the ethical issues and of course to overcome them. Beside this, it is much more time spend in contrast to quantitative research design. Concluding that using both mixed strategies are a time saving strategy in the view of ethical formality. In this study, having in mind that game set up is quality set up, concludes that it can be refereed as qualitative design. Above all, we are talking about a social science that can arise sometimes in natural.

3.3 Data Generation and Analysis Methods

The data generation method used in this study was a combination of experiments, analysis, simulation in a way and correlations.

Experiments are evaluated in this case, as a precedent, when an exploration technique is applied something similar to try get our head around an issue. Where, in the case of an algorithm you have to solve the game first and moreover understand the issues. Also, a comparison is made between the implemented algorithm's execution time which we will analyze in discussion section, with the execution and complexity of an external software named Gambit[15] more explanation will be provided in discussion as well. Method also extends experiments in the field where an explanation on how and why some properties work in a specific way defined. Furthermore, demonstration and theory validation is used. Demonstrating a Nash balance(equilibrium) which is actually a proof about a fact in a multi-player game. Finally, the expected outcomes converted into theoretical results based on the literature review's background.

The data analysis is also a crucial part, which not solving the main problem and other side issues by answering question, but moreover it gives you the direction for future data collection. In simple words, opening cases where you can again and collect more data. There are many tools like (DAP) which helps you with the procedure to arrive your data. In the next step by evaluating this, easily can be helpful for

testing the hypothesis you started. The provided utilities are, converting data into information and knowledge. Explore the relationship between variables. Which in our case is indeed needed for the algorithm metrics.

3.3.1 Simulation Method

As mentioned in the begging, there is used a simulation method. As far we consider the as it refers there must be a physical simulation. In other words, there must be something build with material components. Actually, in our case there is no but the defined method "simulation" is used. I am trying to simulate first of all a game which could arise in natural. By converting a natural procedure like, crossing the street in a formal game using notations and terminology come to the conclusion that there is actually simulation. Of the mathematical point of view, this simulation is precisely calculated using numbers.

3.3.2 Correlative Method

Another method used in, is correlative. Correlation stands for association, more analytically it is a measure of the extend to which two variables are related. For example, the fact that one variable tends to increase itself and it is associated with the variable two is known as a positive correlation. In the case of game "Traffic Lights", that we have the strategies picked by both players we have a correlation equilibrium on the payoff as analyzed above. Beside this example of correlated equilibrium, actually there is correlation in every mixed strategy game and not only. Row players payoffs could be changed analogous to the column player's picked strategy. Prediction, validity and reliability is some of the correlation method's uses.

3.4 Sampling

Sampling used in this study, is non-probability sampling. This means that i decide by my self what units should I use for theoretical or practical part. Without asking

someone for permission to use some data. In example, if i could have to deal with people and names. That is not happening in this point. References are provided over the literature part. From which we will utilize the terminologies and notations. The sampling is not over-sized neither under sized. As the used units comes specifically.

3.5 Ethics

There are a number of ethical principles that should cared when writing an undergraduate thesis even a master one. As it is a crucial component of a dissertation. The main outline of principal ethics is; beneficence and non-malfeasance. Moreover, in practice it has to do with the participants and their anonymity, confidentiality. Where in our case, there are no physical participants. So, there is no ethics principal issue to challenge.

3.6 Limitations

All researchers suffers from limitations, whether in theoretical or practical part. In part, there is clearly some notations that are extra. And some other section that could be analyzed in more depth. Among this, is not something big as you can avoid or search over the missing parts. Finally, in the practical part, is also clear that there could have been developed a much higher quality algorithm. Having in mind, the efficiency and complexity. However, its good for someone who wants the job done no matter the quality.

3.7 Conclusions

The mixed strategy methodologies followed in this study, are both important in a research with of computer science. Especially what have to do with economics and mathematics where use both In this chapter, a brief but specific representation of what will follow introduces gently to the reader how these method came up to table

on this specific research. How they are used is mentioned as well. But have in mind that more analyzing and evidence will be provided for the hypothesis in chapter 4.

Chapter 4

Findings and Analysis Discussion

4.1 Introduction

Coming back to the objectives of this research, means to be conducted in order to implement an algorithm running in quadratic time of searching Nash equilibrium in multi-player games. To be able gather the necessary data to achieve this, the researcher used a combination of different methods, using both qualitative and quantitative approaches. Moreover, a reliable plan about how this objectives will be accomplished using the current methods based on the theoretical background is analyzed. Which on their turn, will achieve the aim. Furthermore, providing practical evidence for the hypothesis.

4.2 Definitions

To start up, lets define a mind map on how thing could defined together to get our expected outcome. Having in mind the notation from chapter 2 (2.6.1 Terminology and Notation) we have a struct of n-player {1, 2, 3, ... n} where each player i has his own set of possible strategies S_i which means another struct for the possible strategies S_i where $s_i \in S_i$. To start the game we must have: Each player i selects a possible strategy from the given range. There are many ways that could store the strategies for example, as vectors, as struct, but in our occasion are stored in an array. All these

according to the $S = x_i S_i$ possible ways in which player can pick strategies.

To implement the brute-force algorithm. The overall thinking is much easier than it seems to be. First of all we will define a two dimensional array for each player which will stand for the payoff matrix we already know. And will show the payoffs for each player separately. So, we would need a second two dimensional array for this job. Defined for the second player. And so on, for three and more. Furthermore, in the game set up we test our algorithm the payoffs are not gathered from input of the user. But, adds randomly the payoff values to the player. Using the brute-force or exhaustive search technique which is also known as generate and test, which is a kind of very general problem-solving technique that consists from systematically enumerating all possible candidate in our occasion payoffs, for the solution and checks if each candidate satisfies the problem's statement[see brute-from search] we are searching in each player's payoff matrix for pure equilibrium according to the math theorem in subsection 2.7.2. More formally explanation is provided below.

	a1	a2
b1	Random Number Generated	Random Number Generated
b2	Random Number Generated	Random Number Generated

Figure 4-1: Payoff matrix of Player i

In figure 4-1 is defined a payoff matrix for player i where the random generated numbers for the player are the costs. Additionally, the $a1$ and $a2$ are the costs for player i (column player). Exactly one another same matrix have to be defined for the row player i. Where the costs now would be the $b1$ and $b2$. If we assume that each player i has a set of strategies $S_1 = \{a1, a2\}$ and $S_2 = \{b1, b2\}$ then if we have $S_1 = \{1, 2\}$ and $S_2 = \{3, 4\}$. Concluding to the A x $B = \{$(x,y) / x $\in$ A, y $\in$ B$\} \equiv$ $\{$(1,3), (2,4)$\}$.

4.2.1 Analysis

More formally now, to get this done. We need to define a payoff matrix as 2D array where the dimensions would be n x n for each player i. In our occasion there are two for two player. Beside this, a maximum indicator for each player i is needed. To store

the maximum outcome after the brute force search done his job. So we could later show it. Another crucial component is if you want to give the payoffs for each player hard coded by input or in our case it does not matter because we are just testing the algorithm to see if it performs well. So, we construct a method to assign values to the two matrices as payoffs randomly. Just with a simple for loop every row and column. At this point, the struct is kind of ready. The only think is to figure out how to find the equilibrium in those matrices. We define two more method each for one matrix for player i. That performs our technique according to the conclusion in previous paragraph. Where A x B can do the job. So, simple search with a for loop in rows and columns for each player. And a local variable to store the maximum value found on each matrix. The maximum payoff. And assign it to the matrix we defined in the begging that hold the row and column position of the matrix where now we assigned the maximum value. All you need a main method to call the function and print the outcome. That is up to everyone make it by his own. In spite of, i present my version of source that you can evaluate and test on your own. To change the game set to more the two player logic is quite simple. As an example, just define an array of $[n][n][n]$ for three players. All arrays must have the same aspects as if one of the arrays would be two dimensional the some other more than two there would be a parsing error. Note that, you have to change the methods by adding an additional for loop each dimension.

The orientation could be easily converted to pure objected oriented. With classes, headers and source files. Pseudo code is also provided for people who are not in contact with programming language like C. Source code in C can is provided in Appendix chapter. Beside this, NassiShneiderman diagram (NSD) is available.

Brute-force algorithm

In computer science the, analysis of algorithms is the determination of amount resources such as storage and time necessary to execute them. Most algorithms are designed to work with an input in our case a generated input. Most of the times, efficiency or running time of an algorithm is conducted as a function relating the input to number of steps for time complexity and storage locations for space complexity. In

```
Define:
    payoff_matrix[n][n]
    maximum_indicator[n]

Algorithm:
    Sub Nash equilibrium()
        For i = 0 to n
            max = 0
            ind = -1
        For j = 0 to n
            If payoff_matrix[i][j] > max
                max = payoff_matrix[i][j]
                ind = j
                maximum_indicator[i] = ind
                return i, maximum_indicator[i]
    End Sub
```

Figure 4-2: Brute-force algorithm in pseudo-code

```
int payoff_matrix[n][n];
int maximum_indicator[n];

void findMax() {
    int i, j, max, ind;
    for(i=0; i<n; i++) {
        max = 0, ind = -1;
        for(j=0; j<n; j++)
            if (payoff_matrix[i][j] > max)
                { max = payoff_matrix[i][j]; ind = j; }
        maximum_indicator[i] = ind; }
    return; }
```

Figure 4-3: Brute-force algorithm in objective C

our case, according to the well known Big O notation the algorithms complexity is $O(n^2)$. Which is quadratic, means multiplying two n-digit number by the algorithm.

Next, lets assume the whole game set played more than one times. Followed the simple rules we assumed previously. On two-player game where the payoffs are generated randomly in the range of $[0-9]$. The execution time according to "GNU GCC Compiler" of an open-source IDE differs on the number of games played. Some test have been done. Playing the same game 1 time, 10 times, 100 and 1000 with random generated numbers every time. The result are as follows:

1. 1 round. Execution time: $[0.010 - 0.0020]$ seconds

2. 10 rounds. Execution time: $[0.020 - 0.0022]$ seconds

3. 100 rounds. Execution time: $[0.122 - 0.0175]$ seconds

4. 1000 rounds. Execution time: $[1.220 - 1.275]$ seconds

To sum up, you can try the execution in much round you want. Approximately, this algorithm can find Nash equilibrium in around 800 rounds played with random generated number as payoffs less than 1 second.

Using an external software we mentioned in chapter 3. Named Gambit, we can test the accuracy of this algorithm by using it's so simple interface to create an extensive game showed like tree, or a strategic game as table. Defining the moves, strategies and payoffs as you wish. After that, just by pressing a button you can compute one Nash equilibrium or as many ass possible in a game. Using gambit's recommended method for searching, or by solving linear programming or simplicity subdivision even tracing lo-git equilibrium. There is know output for the execution time in gambit. But, you can check re results on accuracy in many games with your inputs. There could be comparison between this software and our algorithm but there is not because of the use of different methods in finding equilibrium. Pure, mixed or correlated. Execution outputs available for both at appendix B section.

4.3 Conclusions

To sum up, there are many condition along these that the algorithm could designed, implemented and tested. By choosing this technique of brute force we ensure the reader that there is an easy start for going deeper. A simple mechanism like this can be implemented for first time by only reading the literature and some exercises. So, finding an equilibrium is not that difficult as set in the hypothesis. We answered, on how possible is to accomplish the aim set in the rational section.

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 5

Conclusions and Recommendations

5.1 Conclusions

Finally, lets end up with this study by concluding the statements of my main study findings. Always related to the aims and objectives. So, having in mind all the previous related work, we have accomplished the goal of constructing a tool for finding payoffs in our every day life actions and decisions. This is why paper useful. Beside this, you were gently introduced in maybe something new, the whole concept of algorithmic game theory and calculations in many situations arise Specifically talking, we are in a situation where we can know that our action of (i.e. going to sleep now) is the best response according to the situation's payoff.

In this thesis was successfully covered and supported the hypothesis after findings i have accomplished. On how much possible is to do something like implementing an algorithm for this job. Moreover, the difficulties of achieving this was covered successfully. The main outline of the project was followed precisely, using the right methodology which we explained the reason in each occasion.

5.2 Recommendations

Furthermore, there is always space for improvements. Even in the highest quality works. There could be spent much more time, to generate another algorithmic logic,

that will follow the improvement of technique and the overall complexity. Generally, the study of game theory have still today the known as unsolved "one million dollars" problems. I am recommending readers not try to solve these problems, but at least make their own research on how to get close understand these problems. A good procedure could be solving problems with small complexity. Always talking after a deep research. Games and solution as we denote them are always there, anytime.

Chapter 6

Personal Reflection

6.1 Introduction

At this, i would like to make a discussion of my personal point of view. On the emotional effects i got when first get in contact with the theory of games. Also, on what i could easily and with passion get related to in this theory. And finally, i will speak honestly about my thoughts on this thesis. Generally issues i noticed and i could not get over in certain aspects.

6.2 Get Personal

Game theory, already known by numerous of people all over the world. Starting from economics, to biology, computer science and much more. I was first heard of game theory from political people. So, my first idea was that is something boring that have to do with political studies and methodologies. Without knowing what is game theory i just ignored it. This year, i had the pleasure of meeting someone who explained me what is all about in brief. I was so enthusiastic. From the first minute general ideas about this theory was getting in my mind. For example, i was thinking how people can rule the world accurately calculated knowing what are the costs of their actions in every situation. That made me go and study in little more depth what is all about. Then over thinking about what could happen if this, that.

I would like also someday to make a study more near to economic part of game theory. Again without knowing what exactly is about in depth. But only the idea of predicting something drives me emotional grateful. Maybe some studies will be on my plans for future work.

And the last part, where i can honestly say that there could be implemented much more efficient algorithm calculating more aspects. Running maybe in some exponential time. The reasons that this was not accomplished was the time and some theory knowledge. On the other side, my main goal was to implement something that people with not much knowledge of the subject and maybe algorithmic skills could follow my pattern and do the same or even better work. Simplicity was a crucial part of my work. As, there is evidence that someone only getting study the literature part could make the solving possible.

Although, I learned a few things. In the beginning I always enter freak-out mode. First, it paralyzes me. Then I spent the ninety percent of the whole time estimated to find all the task that need to be accomplished and how this can be done though. Then finally after all recourses needed gathered. I just make a collage. And my laziness creativity mode is turned on.

Bibliography

[1] Myerson, Roger B. (1991). Game Theory: Analysis of Conflict, Harvard University Press, p. 1. Chapter-preview links, pp. viixi.

[2] Christopher Chabris (26 July 2013). "The Science of Winning Poker". WSJ.

[3] Janet Chen, Su-I Lu, Dan Vekhter. (Undefined). John Von Neumann. Available: $https://cs.stanford.edu/people/eroberts/courses/soco/projects/1998-99/game-theory/neumann.html$. Last accessed 13th May 2016.

[4] John von Neumann and Oskar Morgenstern / Princeton University Press (2004) Theory of Games and Economic Behavior, Princeton University: Princeton University Press.

[5] C.R. (2014) 'The Nobel prize goes to Jean Tirole', The economist, (), pp. [Online]. Available at: $http://www.economist.com/blogs/freeexchange/2014/10/economics$ (Accessed: 13th May 2016).

[6] New World Encyclopedia (4 July 2013) John Maynard Smith, Available at: $http://www.newworldencyclopedia.org/entry/John_Maynard_Smith$ (Accessed: 13th May 2016).

[7] D. Fudenberg and J. Tirole. Game Theory, MIT Press, 1991.

[8] A. Mas-Colell, M. Whinston, and J. Green. Microeconomic Theory, Oxford Press, 1995.

[9] M. Osborne and A. Rubinstein. A Course in Game Theory, MIT Press, 1994

[10] Noam Nisan, Tim Roughgarden, Eva Tardos and Vijay V. Vazirani (2007) Algorithmic Game Theory, ISBN 978-0-691-13061-3 edn., 32 Avenue of the Americas, New York, NY 10013-2473, USA: Cambridge University Press.

[11] J. Nash. Non cooperative games. Annals of Mathematics, 54:289295,

1951.

[12] Lance Fortnow (2005) What is PPAD?, Available at: *http* : *//blog.computationalcomplexity.org*/2005/12/*what* − *is* − *ppad.html* (Accessed: 13th May 2016).

[13] Xi Chen, Xiaotie Deng and Shang-Hua Teng, Setlling the Complexity of Computing Two-Player Nash Equilibrium, Journal of the ACM 56 (3), 2009. [Journal version of the two FOCS 06 papers]

[14] R.J. Aumann. Acceptable points in general cooperative n-person games. In: Contributions to the Theory of Games IV, Princeton University Press, 1959.

[15] The Gambit Project (2014) Gambit: Software Tools for Game Theory, Available at: *http* : *//gambit.sourceforge.net/gambit*14/*gui.html* (Accessed: 13th May 2016).

Appendix A

Source Code in Objective C

```
#include <stdio.h>
#include <stdlib.h>

#define n 10

time_t t;
int payoff_p1[n][n], payoff_p2[n][n];
int maxInd_p1[n], maxInd_p2[n];

void assignValues() {
    int i, j;
    for(i=0; i<n; i++)
        for(j=0; j<n; j++)
        { payoff_p1[i][j] = rand()%100; payoff_p2[i][j] = rand()%100; }
    return; }

void findMax_p1() {
    int i, j, max, ind;
    for(i=0; i<n; i++) {
```

```
        max = 0, ind = -1;
        for(j=0; j<n; j++)
            if (payoff_p1[i][j] > max)
                { max = payoff_p1[i][j]; ind = j; }
        maxInd_p1[i] = ind; }
    return; }

void findMax_p2() {
    int i, j, max, ind;
    for(i=0; i<n; i++) {
        max = 0; ind = -1;
        for(j=0; j<n; j++)
            if (payoff_p2[j][i] > max)
                { max = payoff_p2[j][i]; ind = j; }
        maxInd_p2[i] = ind; }
    return; }



int main(int argc, const char * argv[]) {
    srand((unsigned) time(&t));
    int count = 0;

    while(count < 500) {
        assignValues();
        findMax_p1();
        findMax_p2();

        int found = 0;
        int i;
```

```
        for(i=0; i<n; i++) {
            if (maxInd_p2[maxInd_p1[i]] == i) {
                printf("Equilibrium found in: %d,%d\n", i, maxInd_p1[i]);
                found = 1; } }

    if (!found) printf("There is no pure Nash equilibrium\n");
    printf("-----------------------------------\n");

    count++ ;
    }

    return 0;
}
```

THIS PAGE INTENTIONALLY LEFT BLANK

Appendix B

Diagrams

Figure B-1: Nassi-Shneiderman Diagram Part 1

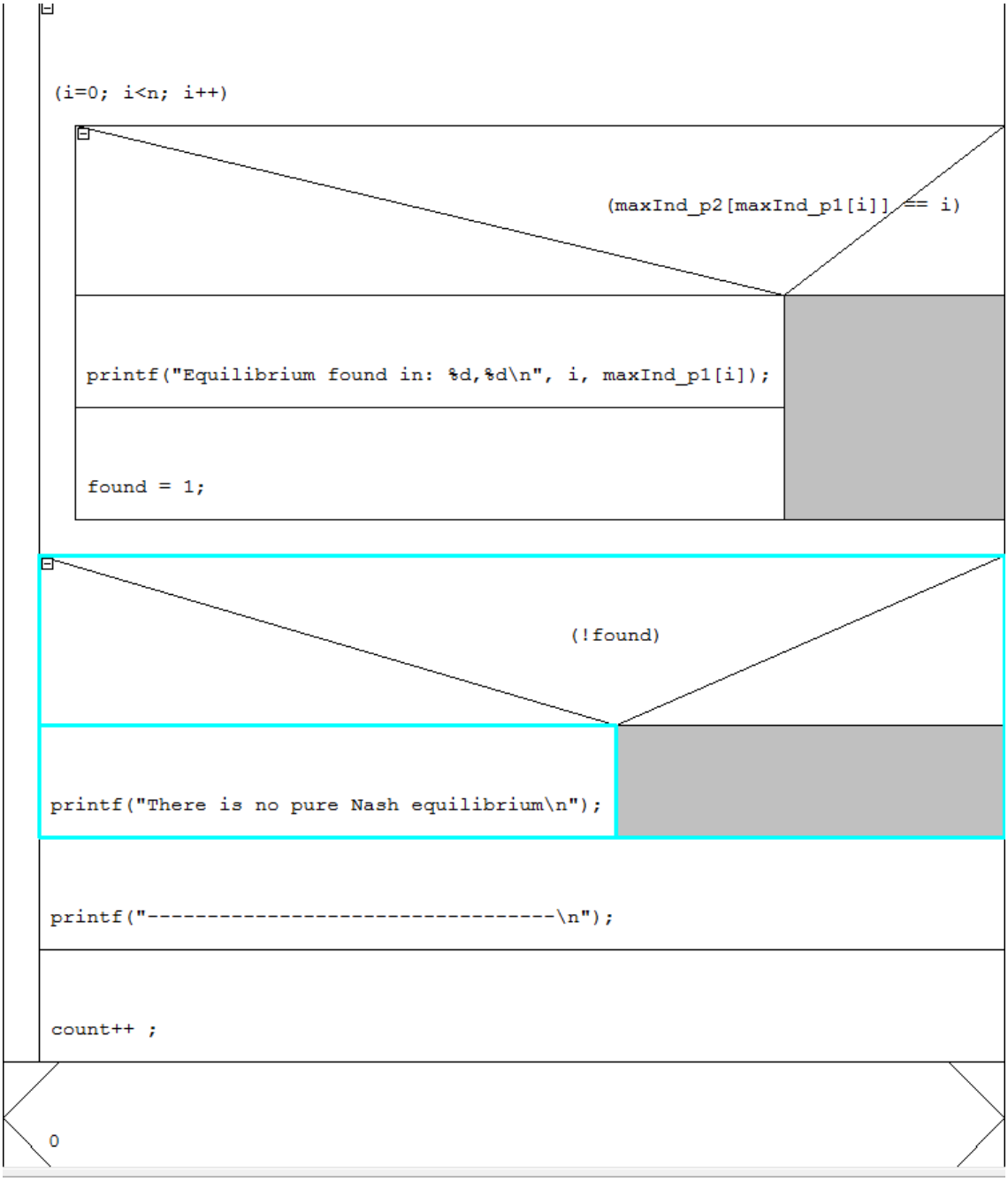

Figure B-2: Nassi-Shneiderman Diagram Part 2