ZADÁNÍ DIPLOMOVÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Milec** Jméno: **David** Osobní číslo: **439597**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra počítačů**

Studijní program: **Otevřená informatika**

Studijní obor: **Umělá inteligence**

II. ÚDAJE K DIPLOMOVÉ PRÁCI

Název diplomové práce:

Bounded Rationality Models in Counterfactual Regret Minimization

Název diplomové práce anglicky:

Bounded Rationality Models in Counterfactual Regret Minimization

Pokyny pro vypracování:

Counterfactual Regret Minimization (CFR) is an algorithmic framework that led to several advancements in computational game theory, such as outperforming human professionals in no-limit Texas Hold'em. Some key methods in this framework assume the players are perfectly rational, which is not the case in most real world applications. The student will:
1) review the usage of bounded rationality models in computational game theory;
2) review CFR variants CFR-D and continual resolving;
3) design or find an algorithm exploiting quantal response opponents in matrix games;
4) generalize the algorithms from 3) to extensive form games;
5) attempt to design a CFR-D-like algorithm for decomposition the computation of the strategy exploiting quantal response opponents.

Seznam doporučené literatury:

Moravčík, Matej, Martin Schmid, Neil Burch, Viliam Lisý, Dustin Morrill, Nolan Bard, Trevor Davis, Kevin Waugh, Michael Johanson, and Michael Bowling. "Deepstack: Expert-level artificial intelligence in heads-up no-limit poker." Science 356, no. 6337 (2017): 508-513.
Burch, Neil, Michael Johanson, and Michael Bowling. "Solving Imperfect Information Games Using Decomposition." In AAAI, pp. 602-608. 2014.
Fang, Fei, Thanh Hong Nguyen, Rob Pickles, Wai Y. Lam, Gopalasamy R. Clements, Bo An, Amandeep Singh, Milind Tambe, and Andrew Lemieux. "Deploying PAWS: Field Optimization of the Protection Assistant for Wildlife Security." In AAAI, pp. 3966-3973. 2016.
Farina, Gabriele, Christian Kroer, and Tuomas Sandholm. "Online convex optimization for sequential decision processes and extensive-form games." arXiv preprint arXiv:1809.03075 (2018).

Jméno a pracoviště vedoucí(ho) diplomové práce:

Mgr. Viliam Lisý, MSc., Ph.D., centrum umělé inteligence FEL

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) diplomové práce:

Datum zadání diplomové práce: **28.01.2019** Termín odevzdání diplomové práce: **24.05.2019**

Platnost zadání diplomové práce: **20.09.2020**

Mgr. Viliam Lisý, MSc., Ph.D.
podpis vedoucí(ho) práce

podpis vedoucí(ho) ústavu/katedry

prof. Ing. Pavel Ripka, CSc.
podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Diplomant bere na vědomí, že je povinen vypracovat diplomovou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v diplomové práci.

.

Datum převzetí zadání	Podpis studenta

Czech Technical University in Prague
Faculty of Electrical Engineering
Department of Computer Science

Diploma Thesis

Bounded Rationality Models in Counterfactual Regret Minimization

David Milec

Supervisor: Mgr. Viliam Lisý, Msc., Ph.D.

Study Programme: Open Informatics

Field of Study: Artificial Intelligence

May 23, 2019

Acknowledgements

I would first like to thank my thesis supervisor, Mgr. Viliam Lisý, Msc., Ph.D. for his professional guidance and support he has provided. Whenever I ran into a trouble spot, he was ready to provide valuable insights or steer me in the right direction.

I would also like to acknowledge David Svoboda as the second reader of this thesis, and I am gratefully indebted to him for his valuable comments on this thesis.

Finally, I must express my gratitude to my parents and my girlfriend for providing me with unfailing support and continuous encouragement throughout my years of study and through the process of researching and writing this thesis. This accomplishment would not have been possible without them.

Thank you.

Author statement for master's thesis

I declare that the presented work was developed independently and that I have listed all sources of information used within it in accordance with the methodical instructions for observing the ethical principles in the preparation of university theses.

Prague, date May 23, 2019 ....

Abstract

In my work, I focused on exploiting quantal response opponents in big imperfect information extensive form games. I defined two new solution concepts, quantal Nash equilibrium, and quantal Stackelberg equilibrium. I analyzed properties of defined equilibria and showed that they are not interchangeable even in a zero-sum scenario. The results showed that CFR-QR, which is an algorithm that I tested, could be used to get the strategy in quantal Nash equilibrium for both normal form games and extensive form games. Obtained results indicated that in both normal form games and extensive form games, there could be multiple quantal Stackelberg equilibria with different values.

I proposed a gradient descent algorithm to reach local quantal Stackelberg equilibrium in Normal form game and modified sequence form program to find quantal Stackelberg equilibrium in extensive form game. I compared both concepts in terms of how much they can exploit the quantal response adversary and how much they can be exploited by a rational opponent, and for both normal form games and extensive form games, quantal Stackelberg equilibrium is better in both aspects. Finally, I tried to apply decomposition to both algorithms, and I discussed problems that arise from a sequence program with decomposition. I proposed CFR-QR-D that can find quantal Nash equilibrium strategy, but in my tests, it converged in 99% of the games.

Keywords: game theory, efg, nfg, imperfect information, decomposition, CFR

Abstrakt

V mé práci jsem se soustředil na využívání soupeřů s modelem omezené racionality, kterým je například quantal response, ve velkých extenzivních hrách s omezenou informací. Definoval jsem dva nové koncepty řešení, quantal Nash equilibrium a quantal Stackelberg equilibrium. Analyzoval jsem vlastnosti definovaných konceptů a ukázal jsem, že i v zero-sum hrách jsou nezaměnitelné. Dále jsem ukázal, že CFR-QR, což je algoritmus, který jsem testoval, se dá použít na nalezení quantal Nash equilibria pro normální i extenzivní hry. Ukázal jsem pro normální i extenzivní hry, že v nich může být více quantal Stackelberg equilibrií s různými hodnotami.

Navrhl jsem algoritmus gradientního sestupu k nalezení lokálního quantal Stackelberg equilibria v normálních hrách a modifikovaný program sekvenční formy na nalezení quantal Stackelberg equilibria v extenzivních hrách. Porovnal jsem oba koncepty v tom, jak moc dokáží soupěře využít a jak moc by je dokázal využít racionální soupeř. Pro normální i extenzivní hry je quantal Stackelberg equilibrium lepší v obou aspektech. Jako poslední jsem se snažil použít dekompozici na oba algoritmy a ukázal jsem problémy, které vznikají při použití sekvenčního programu s dekompozicí. Navrhnul jsem algoritmus CFR-QR-D, který dokáže nalézt quantal Nash equulibrium strategii ale v mých testech zkonvergoval pro 99% her.

Klíčová slova: teorie her, efg, nfg, neúplná informace, dekompozice, CFR

Contents

1 Introduction **1**

2 Background **3**

2.1 Normal form game 3

2.2 Extensive form game 4

2.3 Sequence form 6

2.3.1 Sequence form representation 6

2.3.2 Sequence form linear program 7

2.4 Counterfactual regret minimization 8

2.4.1 ϵ-Nash equilibrium 8

2.4.2 Regret minimization 8

2.4.3 Counterfactual regret 9

3 Related work **11**

3.1 Deepstack 11

3.1.1 Introduction 11

3.1.2 Decomposition 11

3.1.3 CFR-D 12

3.1.4 Continual resolving 13

3.2 Opponent modeling 13

3.3 Quantal response equilibrium 14

3.3.1 Quantal response equilibrium in normal form games 14

3.3.2 Quantal response equilibrium in extensive form games 14

4 Problem specification **17**

4.1 Exploiting quantal response in normal form games 17

4.1.1 Quantal Nash equilibrium 17

4.1.2 Quantal Stackelberg equilibrium 17

4.1.3 Problem 18

4.2 Exploiting quantal response in extensive form games 18

4.2.1 Quantal response in extensive form games 18

4.2.2 Equilibria 18

4.2.3 Problem 19

4.3 Decomposition 19

5 Problem solution **21**

5.1 Equilibria 21

5.2 Normal form games 21

5.2.1 Quantal Nash equilibrium 21

5.2.2 Quantal Stackelberg equilibrium 22

5.3 Extensive form games . . . 24
5.3.1 Quantal Nash equilibrium . . . 24
5.3.2 Quantal Stackelberg equilibrium . . . 24
5.4 Decomposition . . . 25
5.4.1 Quantal Nash equilibrium . . . 25
5.4.2 Quantal Stackelberg equilibrium . . . 25

6 Evaluation 27
6.1 Equilibria analysis . . . 27
6.1.1 Shifting and scaling of payoffs in the game . . . 27
6.1.2 Example games . . . 28
6.1.3 Nash equilibria . . . 28
6.1.4 Quantal Nash equilibrium and quantal Stackelberg equilibrium examples . . . 28
6.1.5 Conclusion . . . 30
6.1.6 Game graphs . . . 31
6.1.7 Graph explanation . . . 31
6.2 Normal form games . . . 33
6.2.1 Multiple quantal Nash equilibria . . . 33
6.2.2 Quantal Nash equilibria results . . . 34
6.2.3 Quantal Stackelberg equilibria results . . . 34
6.2.4 Testing on random games . . . 36
6.2.5 Rectangle games . . . 38
6.2.6 Conclusion . . . 39
6.3 Extensive form games . . . 41
6.3.1 Quantal Nash equilibrium . . . 41
6.3.2 Quantal Stackelberg equilibrium . . . 42
6.3.3 Testing on random games . . . 43
6.3.4 Conclusion . . . 45
6.4 Decomposition . . . 46
6.4.1 Convergence . . . 46
6.4.2 Algorithm speed . . . 47
6.4.3 Solution quality . . . 47
6.4.4 Resolving . . . 48

7 Conclusion 49
7.1 Future work . . . 50

A User guide 53
A.1 Requirements . . . 53
A.2 Data . . . 53
A.3 Normal form games . . . 53
A.3.1 CFR-QR . . . 53
A.3.2 Gradient descent . . . 54
A.4 Extensive form games . . . 54
A.4.1 CFR-QR . . . 54
A.4.2 Sequence program . . . 54
A.4.3 CFR-QR-D . . . 55

B CD structure 57

List of Figures

2.1 Example zero-sum extensive form game. Circles represent states of the game. The number in the circle shows which player acts in that node. The chance player is denoted as N and chance is shown along with the action. Dashed lines join nodes that are in the same information set. Action labels for players are shown near the actions. Box nodes represent terminal nodes with the payoff for player 1 in the box, while the payoff for player 2 is negative. 5

2.2 Example extensive form game converted to it's normal form. 6

3.1 Left: game of rock-paper-scissors. Right: rock-paper-scissors split into trunk and subgame. 12

4.1 Example showing quantal responses in extensive form games. Left: Example game. Right: Same game as left but with strategies based on quantal response to strategy $X = 0.5, Y = 0.5$ using $\lambda = 1$. First value for actions A and B is quantal response along sequences, second value is quantal response in each information set. 19

5.1 Regularization used to redefine finding min-max problem of finding Nash equilibria to min-max problem of finding quantal Nash equilibria on game from Figure 5.1. Black lines are showing saddle-point. Quantal response player is on X axis and rational player on Y axis. Value on the axis is probability that the player will play his first action, in this case action **A** for rational player and action **X** for quantal response player. Left: payoff graph without regularization showing value of function u. Right: payoff graph with regularization showing value of function u'. 22

5.2 Objective function for rational player in game from Figure 5.2. Black line is horizontal line to show that there are really two local minima. . . 23

5.3 Simple game I used to test decomposition for quantal Stackelberg 26

6.1 Game 2 lambda 0.1 31

6.2 Game 2 lambda 2 32

6.3 Game 2 lambda 0.3402 32

6.4 Game 2 lambda 0.693 33

6.5 Convergence curves of CFR-QR on normal form game. Figure shows value after each iteration, best response to actual strategy and quantal Stackelberg equilibrium values for comparison. 35

6.6 Values after each iteration of CFR-QR for multiple normal form games. 35

6.7 Value of the game from Table 6.16 showing strategy in quantal Stackelberg equilibrium. 37

6.8 Values of different solution concepts in square games with rationality $\lambda = 1$. 37
6.9 Normalized values of different solution concepts in square games with rationality $\lambda = 1$. 38
6.10 Values of different solution concepts in square games with rationality $\lambda = 0.1$. 39
6.11 Values of different solution concepts in square games with rationality $\lambda = 10$. 39
6.12 Values of different solution concepts in all generated games with size 144x144 with rationality $\lambda = 1$. 40
6.13 Normalized average values for different solution concepts in rectangle games. 40
6.14 Extensive form game for testing of quantal Nash and quantal Stackelberg equilibria. 41
6.15 Convergence curves of CFR-QR on extensive form game. Figure shows value after each iteration, best response to actual strategy and quantal Stackelberg equilibrium values for comparison. 42
6.16 Values after each iteration of CFR-QR for multiple extensive form games. 43
6.17 Game value of the game from Figure 6.14 based on rational player strategy. 43
6.18 Average values of different solution concepts in randomly generated extensive form games with $\lambda = 1$. 44
6.19 Normalized average values of different solution concepts in randomly generated extensive form games with $\lambda = 1$. 45
6.20 Average values of different solution concepts in randomly generated extensive form games with $\lambda = 2$. 45
6.21 Average values of different solution concepts in randomly generated extensive form games with $\lambda = 0.5$. 46
6.22 Values after each iteration of CFR-QR-D for multiple games. 47

List of Tables

2.1 Simple example of Zero-sum normal form game 3
2.2 Example extensive form game from Figure 2.2 converted to it's sequence form. 7

5.1 Game used in regularization example in Figure 5.1. 22
5.2 Very small game which has two local minima in QSE objective function scenario. 23

6.1 Shift and scale example. From left to right: Original game, the game with shifted payoffs one up, the game with scaled payoffs by 2. 27
6.2 Strategies of both players in games from Table 6.1 in both quantal Nash equilibrium and quantal Stackelberg equilibrium. 27
6.3 Example games. From left to right: Game 1 - dominated action, Game 2 - fully mixed Nash equilibria, Game 3 - no action dominated but not fully mixed Nash equilibria, Game 4 - fully mixed Nash equilibria. . . . 28
6.4 Nash equilibria of the games. 28
6.5 Game 1 QNE 29
6.6 Game 1 QSE 29
6.7 Game 2 QNE 29
6.8 Game 2 QSE 29
6.9 Game 3 QNE 30
6.10 Game 3 QSE 30
6.11 Game 4 QNE 30
6.12 Game 4 QSE 31
6.13 Normal form game with two quantal Nash equilibria. 33
6.14 Example of game solved by CFR-QR. Left: The game. Center: Strategy and expected values for actions for the rational player. Right: Strategy, expected values for action and their exponentials. 34
6.15 Speed of CFR-QR algorithm on Normal Form games with 100 iterations. One number in Game size means that the game is square. 36
6.16 Game used for quantal Stackelberg evaluation. 36
6.17 Speed of gradient descent from Nash equilibrium on Normal Form games. One number in Game size means that the game is square. 36
6.18 Strategy found by CFR-QR with expected values for actions and also exponential values for quantal response adversary. 42
6.19 Speed of CFR-QR with 1000 iterations and sequence form with quantal response on random extensive form games. 44
6.20 Speed of CFR-QR-D algorithm using 100 trunk iterations and 1000 subgame iterations. 47

Chapter 1

Introduction

In recent years there has been a big breakthrough in solving large imperfect information games. The successful algorithm is called Deepstack [15] and it managed to produce more difficult to exploit strategies than any prior approach in the game of Texas Hold'em no limit Poker. Deepstack performed very well and defeated every human expert that finished all 3000 games against it, all except one with statistical significance. This success was possible only because of successful decomposition [2] of big imperfect information games and continual resolving, which together with counterfactual regret minimization [23] forms the hearth of the Deepstack algorithm.

Another rapidly developing field in computation game theory is security games [18, 22, 16]. Security games are deployed in the real world, and the deployment is very successful [3]. Most of the security games are created to face human adversaries in the real world, for example, poachers or smugglers. Therefore, opponent modeling is used in the majority of approaches that are used to solve security games. A model that is used the most is called quantal response, when used with subjective utility for the players it is called SUQR, and real-world experiments showed improvements over solutions without opponent models in security games [17].

Continual resolving framework could be possibly employed for many other instances of large imperfect information extensive form games. It is only logical to use opponent modeling for the successful deployment of these games to the real world. In this work, I analyzed the quantal response model in both normal form games and extensive form games.

I defined two solution concepts in the quantal response adversary scenario, quantal Nash equilibrium, and quantal Stackelberg equilibrium for both normal form games and extensive form games. For quantal Stackelberg equilibrium, I also defined its local version. Then I showed that CFR-QR could be used to find quantal Nash equilibrium in both normal form and extensive form games.

I created a new algorithm to find the local quantal Stackelberg equilibrium in normal form games using gradient descent from Nash equilibrium, and I also defined a mathematical program to solve quantal Stackelberg equilibrium in extensive form games. However, because of insufficient scalability, I can not solve it optimally for bigger games, and the result can be local quantal Stackelberg equilibrium. The last algorithm I proposed is CFR-QR-D, which is a decomposition version of CFR-QR. This algorithm can, in most cases, compute strategy in a trunk when the game is split to a trunk and subgames by using constant sized information from solved subgames.

In evaluation, I showed some properties of quantal response and newly defined equilibria. Further, I evaluated on randomly generated games, and it showed that for both Normal form and extensive form games, quantal Stackelberg equilibrium is better solution concept in measure of exploitation the opponent and how much it can be exploited by a rational opponent. However, proposed algorithms for computation of quantal Stackelberg equilibrium have insufficient scalability compared to the CFR-QR.

Additionally, I analyzed the problem with convergence of the CFR-QR-D algorithm and also how to resolve the subgame strategy after computing the trunk strategy by CFR-QR-D.

Chapter 2

Background

2.1 Normal form game

Normal form game [5] is a tuple (N, S, U). Where N is a set of players, which is a finite set $\{1, 2, ..., n\}$, where n is a number of players. S is a set of pure strategy spaces S_i for each player. U is a set of utility functions for each player. Utility function $u_i(s)$ assigns a payoff for each pure strategy profile $s = (s_1, s_2, ..., s_n)$. When referring to opponents of the player i, I mean all other players than the player i, I denote them $-i$.

Zero-sum game is a two player game such that $\forall s, \sum_{i=1}^{2} u_i(s) = 0$. The key feature is that the sum of utilities is a constant, and when I set this constant to the zero, it is called normalization. I will use zero-sum games in my work, e.g., games where players are truly opponents; thus, whenever one player wins, the other must lose.

Zero-sum games are depicted as matrices, as shown in the Table 2.1. In this game player 1 has pure strategies X, Y, Z and player 2 has pure strategies A, B, C. Payoffs for the player 1 are in the matrix and payoffs for the player 2 are numbers in the matrix but negative. So it can also be looked at in a way that the player 1 is maximizing payoffs in the matrix and the player 2 is minimizing.

Mixed strategy, denoted as σ_i is a probability distribution over pure strategies. Each player's randomization is statistically independent of strategies of its opponents. Payoff to a profile of mixed strategies are the expected values of the corresponding pure strategy payoffs. The space of mixed strategy profiles is denoted $\Sigma = \Sigma_1 \times \Sigma_2 \times ... \times \Sigma_I$ with element σ. Now I will overload utility function to add profile of mixed strategies as follows:

$$u_i(\sigma) = \sum_{s \in S} u_i(s) \prod_{j \in N} \sigma_j(s_j)$$

and also add $u_i(\sigma, s_i)$ as expected payoff for playing a pure strategy s_i when other

	A	**B**	**C**
X	1	3	5
Y	4	1	2
Z	2	5	1

Table 2.1: Simple example of Zero-sum normal form game

players play according to σ defined as

$$u_i(\sigma, s_i) = \sum_{s \in S, s_i \in s} u_i(s) \prod_{j \in N \setminus \{i\}} \sigma_j(s_j)$$

The *support* of a mixed strategy is the set of pure strategies to which the mixed strategy assigns a positive probability.

To discuss a varying strategy of a single player i while holding the strategies of his opponents fixed, I denote strategy selection for all players but i as $s_{-i} \in S_{-i}$ and write (s'_i, s_{-i}) for the strategy profile $(s_1, ..., s_{i-1}, s'_i, s_{i+1}, ..., s_I)$. And similarly for mixed strategies $(\sigma'_i, \sigma_{-i}) = (\sigma_1, ..., \sigma_{i-1}, \sigma'_i, \sigma_{i+1}, ..., \sigma_n)$.

There can be pure strategies with lower payoff than any other pure or mixed strategy independently on how opponents play. These strategies are called *dominated* and I will define two domination concepts. A pure strategy s_i is *strongly dominated* if there exists $\sigma'_i \in \Sigma_i$ such that $u_i(\sigma'_i, s_{-i}) > u_i(s_i, s_{-i}), \forall s_{-i} \in S_{-i}$. A pure strategy s_i is *weakly dominated* if there exists $\sigma'_i \in \Sigma_i$ such that $u_i(\sigma'_i, s_{-i}) \geq u_i(s_i, s_{-i}), \forall s_{-i} \in S_{-i}$ and there exists $\sigma'_i \in \Sigma_i$ and $\exists s_{-i} \in S_{-i}$ such that $u_i(\sigma'_i, s_{-i}) > u_i(s_i, s_{-i})$.

Best response is a strategy profile σ^*_i if $\forall \sigma_i \in \Sigma_i, u_i(\sigma^*_i, \sigma_{-i}) \geq u_i(\sigma_i, \sigma_{-i})$ and I will denote the set of all best responses for player i to strategy profile σ_{-i} as $BR_i(\sigma_{-i})$.

Nash equilibrium is a strategy profile where each player's strategy is best response to other player's strategies. Formally, mixed strategy profile σ^* is a *Nash equilibrium*, if $\forall i \in N, \sigma_i \in BR_i(\sigma_{-i})$.

Finally, I define *Stackelberg equilibrium.* There can be cases where my agent's strategy is known in advance and announced to other players. This holds, for example, in Security games where I have some plan, and the opponent can observe my strategy and then react to it. In this case, I want to optimize my payoff, knowing that the opponent knows my strategy and will play a best response to it. Commonly named, the agent that publicly commits to a strategy is called *leader* and all other agents are called *followers.* Formally when player i is leader *Stackelberg equilibrium* is defined as follows:

$$\arg\max_{\sigma \in \Sigma} u_i(\sigma), \quad \text{s.t.} \forall j \in N \setminus \{i\}, \sigma_j \in BR_j(\sigma_{-j})$$

With a Stackelberg equilibrium, there arises a question of how to break ties for followers in case of multiple best responses. There are two main options, *strong Stackelberg equilibrium* where followers select such strategies that maximize the outcome of the leader. When I use the term Stackelberg equilibrium, I mean a strong Stackelberg equilibrium. The opposite is *weak Stackelberg equilibrium* where followers select such strategies that minimize the outcome of the leader. Weak Stackelberg equilibrium is not guaranteed to exist and therefore is used very sparsely.

2.2 Extensive form game

If players choose their actions simultaneously, normal form game is enough as a representation. However, when I need to model dynamic structure, the game size of a normal form game would exponentially increase. The increase is caused by all possible situations I can encounter in the game, including these induced by opponent moves and stochastic events. Therefore, I would need action for each sequence of situations that can happen. To deal with these problems there is more compact representation called *extensive form game* [6].

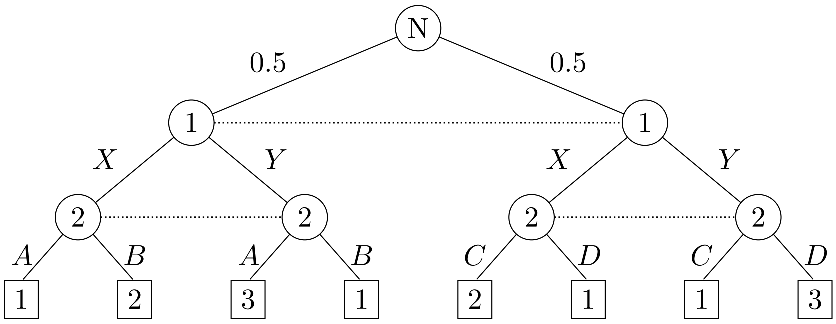


Figure 2.1: Example zero-sum extensive form game. Circles represent states of the game. The number in the circle shows which player acts in that node. The chance player is denoted as N and chance is shown along with the action. Dashed lines join nodes that are in the same information set. Action labels for players are shown near the actions. Box nodes represent terminal nodes with the payoff for player 1 in the box, while the payoff for player 2 is negative.

Perfect information extensive form game [6] is a tuple (N, A, H, T, ρ, χ, φ, $\mathbf{u}$). Where N is the set of players, which is a finite set $\{1, 2, ..., n\}$, where n is number of players. A is the set of actions, for player i $A_i \subseteq A$ denotes the set of his actions and $a \in A$ denotes a generic action. H denotes the set of decision nodes (histories), where $H_i \subseteq H$ denotes the set of decision nodes of player i. T is the set of terminal nodes and $H \cup T$ is the set of all nodes, with $w_0 \in H \cup T$ being root node. $\rho : H \rightarrow N$ is a player function which returns player that acts in a given decision node. χ is an action function, that returns actions available to player $\rho(w)$ at $w \in H$. $\varphi : H \times A \rightarrow H \cup T$ is a successor function that assigns next node $w \in H \cup T$ to pair (v, a) where $v \in H$ and $a \in \chi(v)$. $\mathbf{u} = (u_1, u_2, ..., u_n)$ is the set of players' utility functions $u_i : T \rightarrow \mathbb{R}$.

When I want to deal with opponent actions that I can not observe, for example, some secret bets, or stochastic events that can not be observed, for example, cards dealt for other players, some nodes in the game tree cannot be distinguished by some players. Games with such elements are called *imperfect information extensive form games* and nodes that are indistinguishable are in *information sets.*

Formally, *imperfect information extensive form game* is tuple ((N, A, H, T, ρ, χ, φ, $\mathbf{u}$), $\mathcal{I}$), where (N, A, H, T, ρ, χ, φ, $\mathbf{u}$) is a perfect information extensive form game and $\mathcal{I} = (\mathcal{I}_1, \mathcal{I}_2, ..., \mathcal{I}_n)$ is partition where $\mathcal{I}_i$ is a set of equivalence classes on decision nodes of a player i with the property that $\rho(h) = \rho(h') = i$ and $\chi(h) = \chi(h')$, whenever $h, h' \in I$ for some information set $I \in \mathcal{I}_i$. I will write $\chi(h)$ and $\chi(I)$ for $h \in I$ interchangeably.

I will visualize extensive form games as trees as shown in Figure 2.1.

Pure strategy s_i in a extensive form game for the player i is assignment of an action for each information set where the player i acts and S_i is the set of all pure strategies for the player i. Formally:

$$S_i := \prod_{I \in \mathcal{I}_i} \chi(I)$$

Mixed strategy in an extensive form game is again a probability distribution over pure strategies denoted as σ. An element of sigma corresponding to the player i is denoted σ_i and the space of all possible strategy profiles is denoted $\Sigma = \Sigma_1 \times \Sigma_2 \times ... \times \Sigma_n$.

In extensive form games there is one more strategy definition and it is called *behavioral strategies*, the set of behavioral strategies for player i is defined as $B_i = \prod_{I \in \mathcal{I}_i} \delta(\chi(I))$. That is the probability distribution over actions in each information sets. *Perfect recall games* are games where no player forgets any information it previously knew. In these games, behavioral and mixed strategy have the same expressiveness, and I will only be dealing with perfect recall games.

Let $\pi^\sigma(h)$ be the probability of reaching node h if players choose actions according to σ. π_i^σ is contribution of player i for reaching h and π_{-i}^σ is product of all players' contributions except the player i. For $I \in \mathcal{I}$ define $\pi^\sigma(I) = \sum_{h \in I} \pi^\sigma(h)$ as the probability to reach particular information set given σ. $\pi_i^\sigma(I)$ and $\pi_{-i}^\sigma(I)$ is defined similarly.

To use payoff function with strategies I use $u_i(\sigma)$ for the expected payoff of player i if all players follow strategy σ defined as $u_i(\sigma) = \sum_{t \in T} \pi^\sigma(t) u_i(t)$ and I use $u_i(\sigma_i', \sigma_{-i})$ for expected payoff of player i if all players play according to σ and player i plays according to σ'. Formally $u_i(\sigma_i', \sigma_{-i}) = \sum_{t \in T} \pi_i^{\sigma'}(t) \pi_{-i}^\sigma(t) u_i(t)$.

Best response, *Nash equilibrium* and *Stackelberg equilibrium* are defined the same way as in normal form games. *Best response* is a strategy profile σ_i^* if $\forall \sigma_i \in \Sigma_i, u_i(\sigma_i^*, \sigma_{-i}) \geq u_i(\sigma_i, \sigma_{-i})$ and I denote set of all best responses for player i to strategy profile σ_{-i} as $BR_i(\sigma_{-i})$.

Nash equilibrium is a strategy profile such that each player's strategy is best response to other player's strategies. Formally, mixed strategy profile σ^* is a *Nash equilibrium*, if $\forall i \in N, \sigma_i \in BR_i(\sigma_{-i})$.

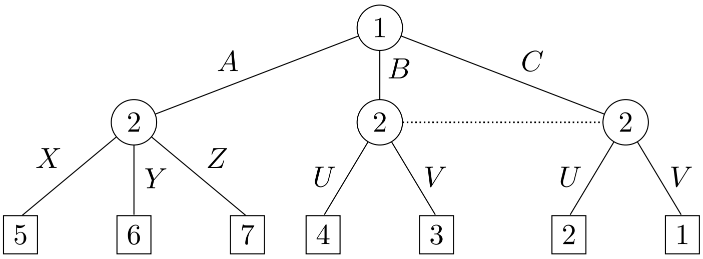
And *Stackelberg equilibrium* is defined as

$$\arg\max_{\sigma \in \Sigma} u_i(\sigma), \quad \text{s.t.} \forall j \in N \backslash \{i\}, \sigma_j \in BR_j(\sigma_{-j})$$

2.3 Sequence form

2.3.1 Sequence form representation

Converting games between forms is possible. However, as already mentioned, the size of a normal form game created from an extensive form game can grow exponentially. Figure 2.2 shows the extensive form game converted to its normal form, and it can be seen that the same payoff from one terminal node can appear three times in the matrix representing the normal form.



	(X,U)	(X,V)	(Y,U)	(Y,V)	(Z,U)	(Z,V)
A	5	5	6	6	7	7
B	4	3	4	3	4	3
C	2	1	2	1	2	1

Figure 2.2: Example extensive form game converted to it's normal form.

To deal with this problem *sequence form* [19] representation is used. It describes strategies in a new way, rather than planning a move for each information set player can look at the terminal nodes and consider choices he needs to make to reach that terminal node. These choices form a path from the root to the terminal node, and they represent a sequence that will be considered instead of a pure strategy. A *sequence* of choices for player i defined by node w, is set of labels D_i on the path from root to w. It is denoted as s_i. The sequence is defined as a set because all labels are distinct. S_i is set of all sequences for player i and S is set of all sequences $S_i \subseteq S$. In my game for player 1, there are sequences A, B, C, and empty sequence $\emptyset$. Sequences for player 2 are $\emptyset$, X, Y, Z, U and V. Sequences of chance player 0 are also considered to use only payoffs and not expected payoffs.

Payoff function $u : S_0 \times S_1 \times ... \times S_n \rightarrow \mathbb{R}^n$ in sequence form is defined by $u(s) = u(t)$ if s is in tuple $(s_0, s_1, ..., s_n)$ of sequences defined by the terminal node t and by $u(s) = (0, 0, ..., 0) \in \mathbb{R}^n$ otherwise.

In addition to payoffs, it is necessary to specify how the sequences are selected by a player. In normal form game, it is possible to select one pure strategy or using mixed strategy, use probability distribution to select one. In the sequence form, a player can no longer decide on a single sequence. In my example (Figure 2.2) player 2 has to decide between X, Y, Z and U, V. If he would choose X and U as in pure strategy (X,U) probability assigned to sequences ($\emptyset$, X, Y, Z, U, V) are $(1, 1, 0, 0, 1, 0)$. Sequence form matrix is shown in Table 2.2. The matrix is of similar size, but each payoff is there only once, so the matrix is very sparse and can be represented using far less memory.

Now if player i uses behavioral strategy β_i, sequence $s_i \in S_i$ is played with probability $r_i(s_i) = \prod_{c \in s_i} \beta_i(c)$ and the function $r_i : S_i \rightarrow \mathbb{R}$ is called *realization plan* of β_i.

	$\emptyset$	X	Y	Z	U	V
$\emptyset$						
A		5	6	7		
B					4	3
C					2	1

Table 2.2: Example extensive form game from Figure 2.2 converted to it's sequence form.

2.3.2 Sequence form linear program

Using sequence form leads to an optimization problem that can be used to solve the game and arrive at the Nash equilibrium of the game. I am solving two-player zero-sum games in this work, so I will show how the linear program looks for a two-player zero-sum extensive form game. Variables for one player are realization plans, and for the other player, it is the expected utility in his information sets; thus, the linear program will be as follows:

$$
\begin{aligned}
& \max_{r_1, v} v(root) \\
& s.t. \quad r_1(\emptyset) = 1 \\
& 0 \leq r_1(s_1) \leq 1 && \forall s_1 \in S_1 \\
& \sum_{a \in \chi(I_1)} r_1(s_1 a) = r_1(s_1) && \forall s_1 \in S_1, \forall I_1 \in inf_1(s_1) \\
\sum_{I' \in \mathcal{I}_2 : s_2 a = seq_2(I')} v(I') + & \sum_{s_1 \in S_1} u(s_1, s_2 a) r_1(s_1) \leq v(I) && \forall I \in \mathcal{I}_2, s_2 = seq_2(I), \forall a \in \chi(I)
\end{aligned}
$$

where $seq_i(I)$ is sequence of player i to information set $I \in \mathcal{I}_i$. $v(I)$ is expected utility at information set I. $inf_i(s_i)$ is an information set, where the last action of s_i has been executed. $s_i a$ denotes extension of a sequence s_i with action a.

2.4 Counterfactual regret minimization

If not stated otherwise, this whole section is based on [23]. When I use abbreviation CFR, I refer to Counterfactual regret minimization.

2.4.1 ϵ-Nash equilibrium

Mostly in iterative algorithms I do not have guarantee to reach a Nash equilibrium in finite number of iterations. However I can have guarantee that after finite number of iterations I am close to a Nash equilibrium. To denote this fact I will define *ϵ-best response* of player i as a strategy profile σ_i^* if $\forall \sigma_i \in \Sigma_i, u_i(\sigma_i^*, \sigma_{-i}) + \epsilon \geq u_i(\sigma_i, \sigma_{-i})$ and I will denote the set of all ϵ-best responses for the player i to the strategy profile σ_{-i} as ϵ-$BR_i(\sigma_{-i})$.

Now *ϵ-Nash equilibrium* is a strategy profile such that each player's strategy is ϵ-best response to other player's strategies. Formally, mixed strategy profile σ^* is a *ϵ-Nash equilibrium*, if $\forall i \in N, \sigma_i \in \epsilon$-$BR_i(\sigma_{-i})$.

2.4.2 Regret minimization

Regret minimization considers playing extensive form game repeatedly. Let σ_i^t be the strategy used by player i on round t. The *average overall regret* at time T is defined as:

$$R_i^T = \frac{1}{T} \max_{\sigma_i^* \in \Sigma_i} \sum_{t=1}^{T} (u_i(\sigma_i^*, \sigma_{-i}^t) - u_i(\sigma^t))$$

Now for each information set $I \in \mathcal{I}$ and for each action $a \in \chi(I)$, define:

$$\overline{\sigma}_i^t(I)(a) = \frac{\sum_{t=1}^{T} \pi_i^{\sigma^t}(I) \sigma^t(I)(a)}{\sum_{t=1}^{T} \pi_i^{\sigma^t}(I)}$$

An algorithm for selecting σ_i^t is regret minimizing, if player's i average overall regret goes to zero as $t \to \infty$. Theorem 2.4.1 shows that regret minimizing algorithm can be used in the self play to compute ϵ-Nash equilibrium.

Theorem 2.4.1. *In a zero-sum game at time T, if both player's average overall regret is less than ϵ, then $\bar{\sigma}^T$ is a 2ϵ-Nash equilibrium.*

Proof. See [20] □

2.4.3 Counterfactual regret

The idea behind counterfactual regret is to decompose overall regret into additive regret terms, which can then be minimized independently. Counterfactual regret is defined on information set, and overall regret is bounded by the sum of counterfactual regrets.

Consider information set $I \in \mathcal{I}_i$ and player i's choices made in that information set. Now *counterfactual value* $u_i(\sigma, I)$ is expected utility given that information set I is reached and all players play using strategy σ while player i plays to reach I. If I define $\pi^\sigma(h, h')$ as probability of going from h to h', then formally:

$$u_i(\sigma, I) = \frac{\sum_{h \in I, h' \in T} \pi^\sigma_{-i}(h) \pi^\sigma(h, h') u_i(h')}{\pi^\sigma_{-i}(I)}$$

Let $\sigma|_{I \to a}$ be a strategy profile identical to σ except that the player i always chooses action a when in information set I. The *immediate counterfactual regret* is:

$$R^T_{i,imm}(I) = \frac{1}{T} \max_{a \in \chi(I)} \sum_{t=1}^{T} \pi^{\sigma^t}_{-i}(I)(u_i(\sigma^t|_{I \to a}, I) - u_i(\sigma^t, I))$$

Because positive part of the regret is mostly what is needed, let $R^{T,+}_{i,imm}(I) = max(R^T_{i,imm}(I), 0)$ be the positive portion of immediate counterfactual regret.

Proof that overall regret is bounded by sum of immediate counterfactual regret is in original paper [23]. And the result is formally:

$$R^T_i \leq \sum_{I \in \mathcal{I}_i} R^{T,+}_{i,imm}(I)$$

This enables finding an approximate Nash equilibrium by only minimizing immediate counterfactual regret.

For all $I \in \mathcal{I}$, for all $a \in \chi(I)$:

$$R^T_i(I, a) = \frac{1}{T} \sum_{t=1}^{T} \pi^{\sigma^t}_{-i}(I)(u_i(\sigma^t|_{I \to a}, I) - u_i(\sigma^t, I))$$

is player i's counterfactual regret of taking action a in information set I. As for immediate regret $R^{T,+}_i(I, a) = max(R^T_i(I, a), 0)$, then strategy for next iteration is computed using *Regret matching*. Which selects actions in proportion to the amount of positive counterfactual regret for not playing this action. Formally:

$$\sigma^{T+1}_i(I)(a) = \begin{cases} \frac{R^{T,+}_i(I,a)}{\sum_{a \in \chi(I)} R^{T,+}_i(I,a)} & \text{if } \sum_{a \in \chi(I)} R^{T,+}_i(I, a) > 0 \\ \frac{1}{|\chi(I)|} & \text{otherwise} \end{cases}$$

Now it is possible to use Regret matching in self-play to compute approximate Nash equilibrium. The proof is in the original paper [23].

Chapter 3

Related work

3.1 Deepstack

Deepstack is a general purpose algorithm for a large scale class of imperfect information games [15]. Deepstack combines abstraction to reduce the dimensionality of the state and action spaces with continual resolving to minimize the amount of information that need to be remembered and a look-ahead heuristic. In my work, I focus to use counterfactual regret minimization with decomposition. Therefore I will deal with CFR-D in details. Nevertheless, I will skip the abstractions and neural networks used in Deepstack.

3.1.1 Introduction

When using CFR to solve very large games, I would need to store the whole strategy of the game. This is not possible for most real-world games, for example, HUNL (Heads-Up No-Limit Texas Hold'em poker). Deepstack avoids this by using *continual resolving*, the process of forgetting the strategy used to reach the actual game state and then reconstructing a part of the strategy needed for selecting the next action. In HUNL this can be done from the information of constant size.

3.1.2 Decomposition

In [2] authors show first imperfect information game decomposition as described in this subsection.

In games of perfect information, a strategy can be computed from the actual game state alone. However, in imperfect information games, finding a subgame is not trivial, because game history can provide valuable information, thus significantly changing the strategy [2].

Definition of subgame in perfect information games is subtree rooted at any node. This definition is impossible in imperfect information games as it could exclude part of the same information set from the subgame. The definition of an augmented information set is needed to define subgame in imperfect information games.

Let $h \in H$ be a history, the player $i = \rho(h)$. Let player $j \neq i$ and let $H_j(h)$ be the sequence of player j information sets reached by j in path to h and the actions taken by j. Then, for two states $h, h' \in H, I_j(h) = I_j(h') \iff H_j(h) = H_j(h')$. $I_j(h)$ is called *Augmented information set.*

The *imperfect information subgame* is a forest of trees, closed under both the descendant relation and membership within augmented information sets of any player.

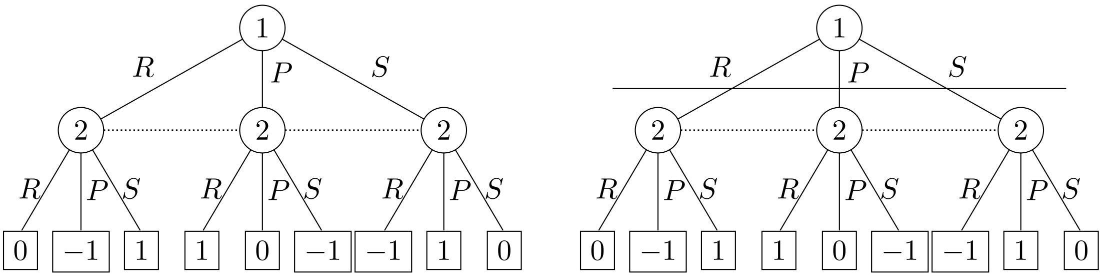


Figure 3.1: Left: game of rock-paper-scissors. Right: rock-paper-scissors split into trunk and subgame.

To illustrate issue with subgame decomposition, consider game of rock-paper-scissors in Figure 3.1. The game is split to trunk and subgame in this example. There is one information set for player 2 $I_2 = \{R, P, S\}$ and three augmented information sets for player 1 $I_1^R = \{R\}, I_1^P = \{P\}, I_1^S = \{S\}$ in the subgame.

Assume I start with a Nash equilibrium for the rock-paper-scissors game. In the trunk the player 1 plays his actions uniformly, in the subgame player 1 cannot take any action. To find the Nash equilibrium of the subgame, player two must pick a strategy that is the best response to no action of player 1 given payoffs induced by the trunk strategy. All his actions have expected utility 0. Thus any strategy is the best response. However, if player 1 switches his strategy in the trunk, the value of the game can substantially change.

To deal with this problem, authors in [2] present method of summarizing a subgame strategy with opponent's counterfactual values $v_{opp}(I)$ for all information sets I at the root of the subgame. These values can be described as values the opponent would receive if he reached the subgame through the information set I, and changed its strategy so that $\pi_{opp}(I) = 1$. Moreover, by generating subgame strategy where the opponent's best response counterfactual values are no higher than the opponent's best response counterfactual values for the original strategy, the exploitability of the combined trunk and subgame strategy is no higher than the original strategy. The exploitability of strategy σ_i is how much player loses if he switches from a Nash equilibrium strategy to σ_i.

To resolve strategy in the subgame, a special gadget game is constructed. A new node is put into the game for the opponent, corresponding to each state in each of his information set. In these nodes, he has a choice to follow the game, which continues as in the original subgame or to terminate and receive previous best response counterfactual value for this state as a reward. This gadget ensures the condition that an opponent's best response counterfactual values in the subgame will not be higher than when using the original strategy [15].

3.1.3 CFR-D

CFR-D is an algorithm that arises from the subgame decomposition. It is inspired by CFR-BR [8] which proceeds as follows. Game is split into trunk and subgames. At each iteration, CFR-BR uses the standard counterfactual regret minimization update for both players in the trunk and one player in the subgames. For the other player, CFR-BR constructs and uses the best response to current CFR player strategy in each subgame.

CFR-D works as follows. First, the game is split to trunk and subgames, then the trunk strategy for both players is initialized uniformly. The algorithm then performs updates that solve one subgame by using current trunk strategy and then updates counterfactual values at the root of the subgame for both players. Next, it uses these values to update the trunk strategy for both players, using CFR update in the trunk. This is performed iteratively one subgame at the time, and after solving the subgame, the corresponding strategy is discarded, and the next subgame is used. The average strategy is then an approximation of Nash equilibrium [2].

If I want to resolve subgame after convergence of the algorithm, I have to keep average counterfactual values at the root of the subgame and reach probabilities for both players. Then I use the gadget game described in the previous section.

3.1.4 Continual resolving

The basic idea of continual resolving is to go one step further in the direction of never storing a strategy. It is done by reconstructing strategy every time Deepstack needs to act, and as soon as it samples the action from the strategy, it forgets the strategy. The public state is defined by the information available to both players. In the case of poker, these are cards on the table face up and betting history. To be able to resolve at any public state, I need two pieces of information, first $\pi_1(I_1)$ for all player 1 augmented information sets in the root of the subgame, second I need opponent's counterfactual values in all opponents augmented information sets in the root of the subgame.

In case of poker, the Deepstack initializes its range at the start of the game to uniform and opponent counterfactual values are initialized to values of being dealt each private hand. When it is time to act the Deepstack re-solves the subtree at current public state using stored values and acts according to computed strategy, after playing the action, the strategy is discarded. When playing, the range and opponent's counterfactual values change according to the following rules. At Deepstack's action opponent's counterfactual values are replaced by the new ones computed in the subgame and range is updated based on the played strategy. In the chance event, Deepstack replaces opponent's counterfactual values with those computed for this chance action in the last resolve and in the range it zeros the hands that are impossible given new information. Moreover, on the opponent's action, no change is required.

These updates ensure that counterfactual values of the opponent meet required conditions and the procedure produces a close approximation of Nash equilibrium [15].

3.2 Opponent modeling

Opponent modeling has been mostly used in Security games [18, 22, 16] because Security games are often used against human adversaries in real world. Opponent model that showed to perform very well is quantal response and it's variation. In security games one player is protecting targets and other player is attacking the targets. In this sense there is some expected utility for attacking the target k based on defender's strategy $\mathbf{x}$ which is a coverage vector where at position i is probability that target i will be protected. Now expected utility for attacking target k is $U_i^a(x_k)$. Then probability of attacking target k, given defender strategy $\mathbf{x}$ according to quantal response with parameter λ is:

$$q_k(\mathbf{x}) = \frac{e^{\lambda U_k^a(x_k)}}{e^{\sum_{j \in \mathcal{T}} \lambda U_j^a(x_j)}}$$

where $\mathcal{T}$ is the set of all targets.

Interesting extension is the subjective utility quantal response (SUQR) [17]. This extension uses subjective utility in the quantal response model which is defined as $\hat{U}_t^a = w_1 x_t + w_2 R_t^a + w_3 P_t^a$ where attacker's utility is split into penalty P_t^a and reward R_t^a. w_1 to w_3 are constants defining the model.

State of the art algorithm to solve optimal strategy against quantal response adversary or even SUQR is called PASAQ, and it uses the binary search with piecewise linearization of constraints. For details see [22].

3.3 Quantal response equilibrium

In my work, I do not directly use quantal response equilibrium, but solution concepts that I use are close to it. There are two very recent studies [4, 11] that use different techniques to solve quantal response equilibria. Quantal response equilibrium was firstly defined in [13] as a noisy alternative to Nash equilibrium.

3.3.1 Quantal response equilibrium in normal form games

Quantal response in normal form game is mixed strategy σ_i in the form:

$$\forall s_i \in S_i, \sigma_i(s_i) = \frac{e^{\lambda u_i(\sigma, s_i)}}{\sum_{s_i' \in S_i} e^{\lambda u_i(\sigma, s_i')}}$$

Where λ is constant setting rationality of the quantal response. Intuitively meaning that in the numerator I have expected utility for action j in exponent with constant *lambda* and in the denominator I have a sum of expected utilities for all actions quantal response player can play. If I put it into contrast with the best response, the best response can be viewed as maximum, whereas the quantal response as softmax.

Then strategy profile (σ^*) is quantal response equilibrium of a normal form game if

$$\forall i \in N, \forall s_i \in S_i, \sigma_i(s_i) = \frac{e^{\lambda u_i(\sigma, s_i)}}{\sum_{s_i' \in S_i} e^{\lambda u_i(\sigma, s_i')}}$$

3.3.2 Quantal response equilibrium in extensive form games

Quantal response in extensive form games is defined in [14]. I need to define the utility for player i in information set I and so far I have only defined counterfactual value. Value $\bar{u}_i(\sigma, I)$ is value in information set I when all players play according to σ. Formally:

$$\bar{u}_i(\sigma, I) = \frac{\sum_{h \in I, h' \in T} \pi^{\sigma}(h) \pi^{\sigma}(h, h') u_i(h')}{\pi^{\sigma}(I)}$$

The quantal response is not directly defined but authors in [14] say choice probabilities in quantal response equilibrium follows the distribution

$$\sigma_i(a) = \frac{e^{\lambda \bar{u}_i(\sigma|_{I \to a}, I)}}{\sum_{b \in \chi(I)} e^{\lambda \bar{u}_i(\sigma|_{I \to b}, I)}}$$

where I is information set for player i, $I \in \mathcal{I}_i$, a is action available in I, $a \in \chi(I)$ and $\sigma_i(a)$ is probability that player i plays action a in information set I following strategy

σ. This means that the strategy is also softmax but in this definition the softmax is used in each information set of the player amongst the actions.

Strategy profile σ^* is quantal response equilibrium if

$$\forall i \in N, \sigma_i^*(a) = \frac{e^{\lambda \bar{u}_i(\sigma^*|_{I \to a}, I)}}{\sum_{b \in \chi(I)} e^{\lambda \bar{u}_i(\sigma^*|_{I \to b}, I)}}$$

which means that each player plays a quantal response to strategies of the other players.

Chapter 4

Problem specification

4.1 Exploiting quantal response in normal form games

The first part aims to design an algorithm that would exploit a quantal response opponent in zero-sum normal form games. In the normal form game, there are two main concepts, Nash equilibrium, and Stackelberg equilibrium. In the zero-sum case with perfectly rational players, these two concepts are interchangeable. I will define solution concepts for quantal response adversary based on these two equilibria.

4.1.1 Quantal Nash equilibrium

Quantal Nash equilibrium is first (could be also referred as QNE), generally this is stable point in the game where all players except player i are playing quantal response to each other and player i's strategies while player i has no incentive to deviate. This means that all action player i plays with non-zero probability are best responses. Formally:

$$\forall \sigma_i \in \Sigma_i \quad u_i(\sigma_i^*, \sigma_{-i}) \geq u_i(\sigma_i, \sigma_{-i})$$
$$\forall j \in N \backslash \{i\}, \forall s_j \in S_j \quad \sigma_j(s_j) = \frac{e^{\lambda u_j(\sigma, s_j)}}{\sum_{s_i' \in S_i} e^{\lambda u_j(\sigma, s_j')}}$$

We can also look at quantal Nash equilibrium from the side of quantal response equilibrium. Then, it is quantal response equilibrium where for one player, I raise his lambda so high that he eventually becomes rational. That would be player i here.

4.1.2 Quantal Stackelberg equilibrium

Quantal Stackelberg equilibrium is second (could be also referred as QSE), generally this is point in the game where all players except player i are playing quantal response to each other and player i's strategies when player i announced his strategy before the game and tries to find such a strategy to maximize his expected utility. Formally:

$$\sigma_i^* = \arg\max_{\sigma_i \in \Sigma_i} u_i(\sigma_i, \sigma_{-i}) \quad \text{s.t.} \quad \forall j \in N \backslash \{i\}, \forall s_j \in S_j \quad \sigma_j(s_j) = \frac{e^{\lambda u_j(\sigma, s_j)}}{\sum_{s_i' \in S_i} e^{\lambda u_j(\sigma, s_j')}}$$

There may exist more local optima for the optimization problem, as I show in evaluation. Because of this fact, I define local quantal Stackelberg equilibrium as a local optimum of the shown optimization problem.

4.1.3 Problem

In the next Chapter, I will analyze the differences and properties of both quantal Nash equilibrium and quantal Stackelberg equilibrium. Then I will find and create algorithms to solve the game for these equilibria.

4.2 Exploiting quantal response in extensive form games

Here, I focus on using the algorithm from the first part to extensive form games or find a new one.

4.2.1 Quantal response in extensive form games

Apart from the definition showed before there is another possible definition of quantal response in extensive form games. It takes the expected reward along all the sequences to terminal nodes and performs softmax on all of them as if they were actions in normal form game. Formally let $s_{i,t}$ be a sequence ending in terminal node from set of terminal sequences of player i S_i^t and $u_i(s_{i,t}, \sigma_{-i})$ is expected payoff for player i when playing actions along sequence $s_{i,t}$ when other players follow fixed strategy defined by σ. Then probability of playing this sequence $q_i(s_{i,t})$, for each sequence from a set of terminal sequences S_i^t is:

$$q_i(s_{i,t}) = \frac{e^{\lambda u_i(s_{i,t},\sigma_{-i})}}{\sum_{s_{i,j} \in S_i^t} e^{\lambda u_i(s_{i,j},\sigma_{-i})}}$$

Since I use regret minimization to solve the problem, I will use the representation from Chapter 3 with one change. In the original representation I use values in information sets. Therefore, in sets where my reach probability is 0 quantal response generates uniform strategy because all of the values will be 0. I will change this by using counterfactual values and I define counterfactual quantal response for player i:

$$\forall I \in \mathcal{I}_i, \forall a \in \chi(I) \quad \sigma_i(a) = \frac{e^{\lambda u_i(\sigma|_{I\to a}, I)}}{\sum_{a' \in \chi(I)} e^{\lambda u_i(\sigma|_{I\to a'}, I)}}$$

This version expects that the quantal response player knows how he will play in the information sets closer to the terminal nodes when calculating strategy for information sets further along the sequence to the root. This assumption is not a very realistic assumption, but because I will handle with counterfactual regret minimization to solve this problem, I will use this representation.

Figure 4.1 gives an example of how both versions of quantal response play. In lowest information sets in the game tree, both versions play the same but in the root, where counterfactual version has more information than sequence version the strategy differs, sequence version having expected utility 2.54 while counterfactual version 2.56. From now on, when I refer to quantal response in extensive from the game scenario, I mean the counterfactual version.

4.2.2 Equilibria

Using quantal response I can define quantal versions of equilibria for extensive form game. Intuitively, it is the same as for normal form games. In quantal Nash equilibrium rational player i plays such strategy that has the highest payoff assuming all strategies

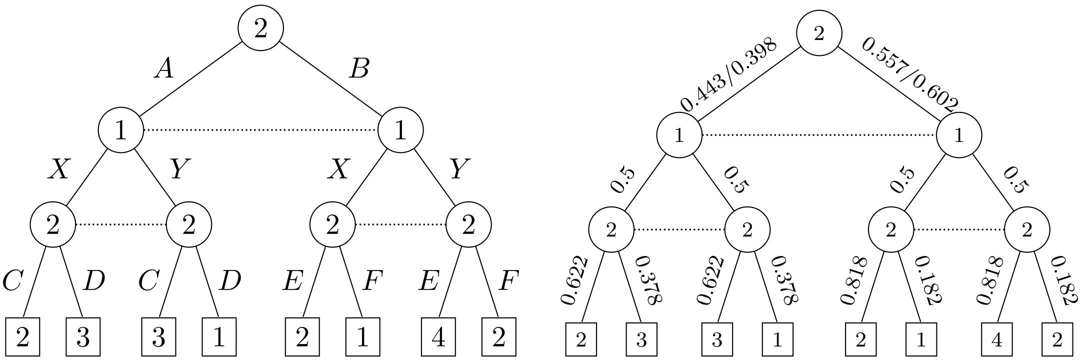


Figure 4.1: Example showing quantal responses in extensive form games. Left: Example game. Right: Same game as left but with strategies based on quantal response to strategy $X = 0.5, Y = 0.5$ using $\lambda = 1$. First value for actions A and B is quantal response along sequences, second value is quantal response in each information set.

for other players are fixed. And all other players play quantal response to all other strategies. Formally:

$$\forall \sigma_i \in \Sigma_i \quad u_i(\sigma_i^*, \sigma_{-i}) \geq u_i(\sigma_i, \sigma_{-i})$$

$$\forall j \in N\backslash\{i\}, \forall I \in \mathcal{I}_j, \forall a \in \chi(I) \quad \sigma_j(a) = \frac{e^{\lambda u_j(\sigma|_{I \to a}, I)}}{\sum_{a' \in \chi(I)} e^{\lambda u_j(\sigma|_{I \to a'}, I)}}$$

And quantal Stackelberg equilibrium:

$$\sigma_i^* = \arg\max_{\sigma_i \in \Sigma_i} u_i(\sigma_i, \sigma_{-i}) \quad \text{s.t.} \forall j \in N\backslash\{i\}, \forall I \in \mathcal{I}_j, \forall a \in \chi(I) \quad \sigma_j(a) = \frac{e^{\lambda u_j(\sigma|_{I \to a}, I)}}{\sum_{a' \in \chi(I)} e^{\lambda u_j(\sigma|_{I \to a'}, I)}}$$

In both cases, the only difference is the definition of quantal response that changes. I also define local quantal Stackelberg equilibrium as a local optimum of the shown optimization problem.

4.2.3 Problem

As in normal form games, I will look at properties of given equilibria. I will analyze which performs better and how exploitable the player becomes by playing these strategies. Moreover, I will test algorithms to solve the problems.

4.3 Decomposition

In the last part, I aim to decompose imperfect information extensive form game in order to use continual resolving later. When I split the game to trunk and subgames, I will analyze whether I can iteratively solve the trunk strategy and subgames as in CFR-D and point out difficulties that arise from using quantal response.

The required algorithm needs to solve the trunk strategy using some values at the root of the subgame, which can be computed one by one or at the end approximated by heuristic function, for example, neural network as in Deepstack.

Chapter 5

Problem solution

In this Chapter, I will describe the techniques used in the solution of defined problems.

5.1 Equilibria

I will show games to demonstrate properties of quantal Nash equilibria and quantal Stackelberg equilibria. In my games row player is the quantal response and maximizes payoffs in the matrix and column player is a rational agent and minimizes payoffs.

First, the essential thing is that I can no longer normalize the game without consequences regarding the strategy. I can shift payoffs of the game, and it will not change the strategy, but I can not scale the payoffs as it has the same effect as changing λ for quantal response player. These facts are demonstrated on results reported in Chapter 6.

A second important remark is that quantal Nash equilibrium and quantal Stackelberg equilibrium are not interchangeable even in a zero-sum scenario. Therefore, I will treat them as different points in the game. I will deal with algorithms aiming to solve both. This is also demonstrated on results shown in Chapter 6

5.2 Normal form games

In this section, I show the solution approach for normal form games.

5.2.1 Quantal Nash equilibrium

To compute quantal Nash equilibrium I use regret minimization. In this case I use regret for playing actions instead of counterfactual regret that is needed in extensive form games. Regret $R_i(a)$ for not playing action $a \in S_i$ for player i is defined as:

$$R_i(a) = u_i(\sigma_a) - u_i(\sigma)$$

where σ_a is strategy profile same as σ except that player i plays action a. Now I adopt it to the iterative scenario and then

$$R_i^T(a) = \frac{1}{T}\sum_{t=1}^{T} u_i(\sigma_a) - u_i(\sigma)$$

is cumulative regret at time T. Now $R_i^{T,+}(a) = \max(R_i^T(a), 0)$. In the algorithm, I will use regret matching as in section 2.4.3 and the opponent updates his strategy as quantal response to actual strategy of regret matching agent.

	A	**B**
X	1	3
Y	2	1

Table 5.1: Game used in regularization example in Figure 5.1.

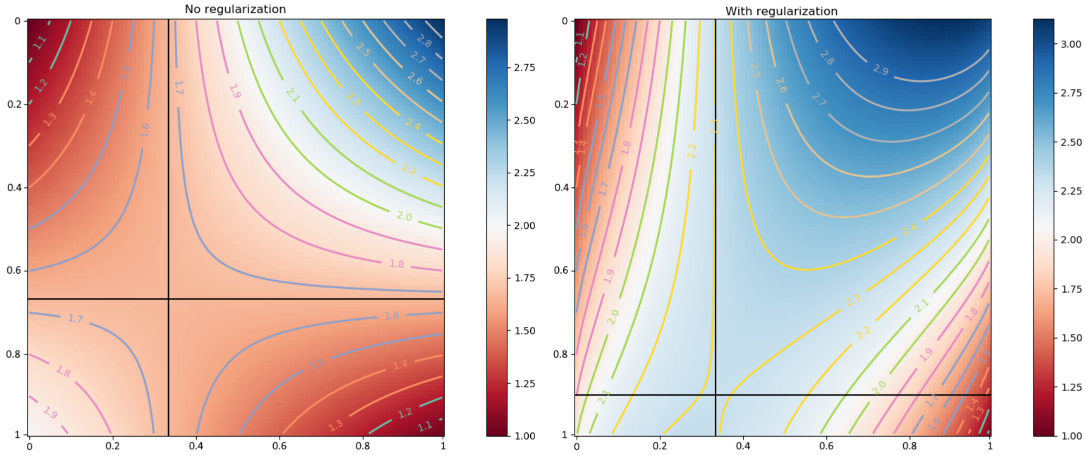


Figure 5.1: Regularization used to redefine finding min-max problem of finding Nash equilibria to min-max problem of finding quantal Nash equilibria on game from Figure 5.1. Black lines are showing saddle-point. Quantal response player is on X axis and rational player on Y axis. Value on the axis is probability that the player will play his first action, in this case action **A** for rational player and action **X** for quantal response player. Left: payoff graph without regularization showing value of function u. Right: payoff graph with regularization showing value of function u'.

Another possible approach is to use algorithms that can compute quantal Response equilibria [11, 4] and set rationality of one player to infinity. This will again lead to quantal Nash equilibrium. Example is in Figure 5.1. Let x be a strategy profile for player 1 and y strategy profile for player 2. Q is payoff matrix for the game. Value of the function $u(x, y) = x^T Qy$ is in the first graph and in the second graph there is added regularization for player 2 so the resulting function is $u'(x, y) = x^T Qy - \frac{1}{\lambda} \sum_{a \in y} a \log a$. In the Figure 5.1 saddle-point for function with no regularization is $(\frac{2}{3}, \frac{1}{3})$ which is Nash equilibrium and for function with regularization, saddle-point is $(0.89, \frac{1}{3})$ which for $\lambda = 1$ is indeed quantal Nash equilibrium.

5.2.2 Quantal Stackelberg equilibrium

The first approach to solve Security games against quantal response adversary used gradient descent [21] and is called BRQR. Authors used gradient descend restarted multiple times from different feasible starting points, so the solution was some local minimum of the problem.

In case of normal form game, there are multiple local minima on the objective function. Game with two local minima is in Table 5.2 and the corresponding objective function is in Figure 5.2.

Following these results, I tried the same approach as BRQR with restarted gradient descent. I compared this approach to gradient descent started from Nash equilibria.

λ	**A**	**B**
X	3	2
Y	7	-8

Table 5.2: Very small game which has two local minima in QSE objective function scenario.

My objective function is the quantal Stackelberg equilibrium formulation, but with a quantal response, constraints moved to the objective. Since I take into account two-player zero-sum game I will use Q as payoff matrix of the game, then strategy profile x^* for player 1 is:

$$x^* = \arg\max_x xQ \frac{e^{\lambda xQ}}{\sum e^{\lambda xQ}}$$

Where the part behind the Q matrix essentially produces opponent strategy by the quantal response, and then it is the same matrix multiplication as in the zero-sum min-max program. Also, there are constraints that x is a strategy, but I did not write it explicitly.

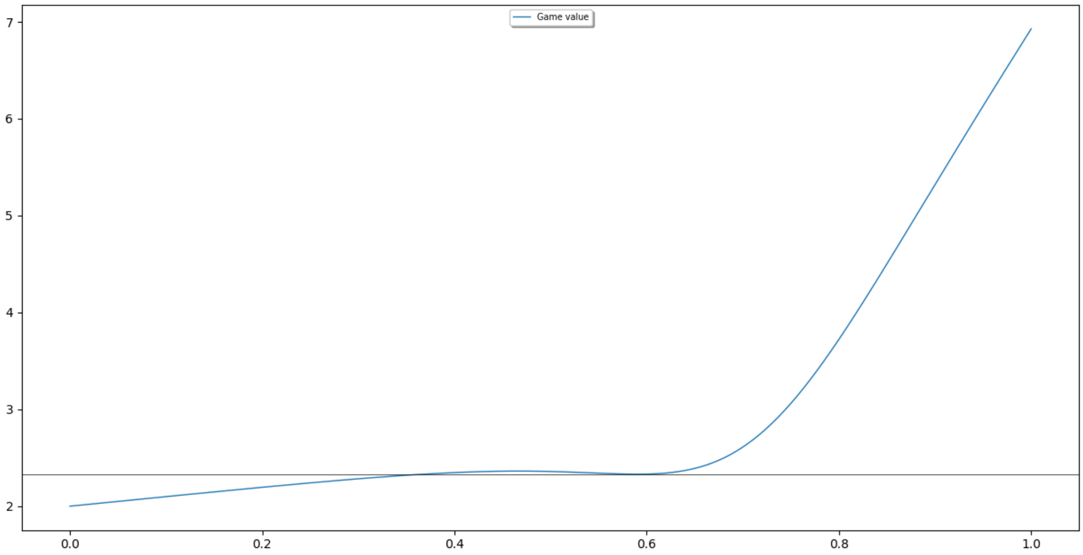


Figure 5.2: Objective function for rational player in game from Figure 5.2. Black line is horizontal line to show that there are really two local minima.

Starting from Nash equilibria can be very beneficial in practice. It does not give guarantee to find the quantal Stackelberg equilibrium, but it will find some local quantal Stackelberg equilibrium with better payoff than just playing the Nash equilibrium strategy against quantal response. On the contrary, starting from random points can reach different local quantal Stackelberg equilibria. Therefore, the probability of reaching quantal Stackelberg equilibrium is higher. However, there can be multiple local quantal Stackelberg equilibria with different values, as shown in Figure 5.2. Therefore, performing one gradient descent from a random point can find local quantal Stackelberg equilibrium with a lower value than the value I receive from playing the Nash equilibrium strategy against quantal response.

5.3 Extensive form games

In this section, I will deal with an approach to the solution for extensive form games.

5.3.1 Quantal Nash equilibrium

To compute the quantal Nash equilibrium, I use CFR-QR, which is based on CFR-BR [8] explained in Section 3.1.3. However, in CFR-QR, after each iteration of CFR opponent updates his strategy as a counterfactual quantal response instead of the counterfactual best response.

Another option, as in normal form games, would again be a modification of quantal response equilibria finding programs. However, CFR and similar algorithms are very well scalable.

5.3.2 Quantal Stackelberg equilibrium

In this case, I aimed to generalize the solution I used for extensive form games. The problem is that constraint I put into the objective is not one equation as in the case of extensive form games. It is a set of equations one for each information set of the opponent. Also, the value of the game is not an easy objective to define, as well.

I modified the sequence form a linear program to solve extensive form games. Where instead of one inequality for each action in each information set to ensure the best response of the opponent, I created a quantal response for the whole information set. The resulting program is

$$\begin{aligned}
&\min_{r_1,v} v(root) \\
s.t.\quad & r_1(\emptyset) = 1 \\
& 0 \leq r1(s_1) \leq 1 && \forall s_1 \in S_1 \\
& \sum_{a \in \chi I_1} r_1(s_1 a) = r_1(s_1) && \forall s_1 \in S_1, \forall I_1 \in inf_1(s_1) \\
& \frac{\sum_{a \in \chi(I)} f(I,a) e^{f(I,a)}}{\sum_{a \in \chi(I)} e^{f(I,a)}} = v(I) && \forall I \in \mathcal{I}_2, s_2 = seq_2(I) \\
\sum_{I' \in \mathcal{I}_2 : s_2 a = seq_2(I')} & v(I') + \sum_{s_1 \in S_1} u(s_1, s_2 a) r_1(s_1) = f(I,a) && \forall I \in \mathcal{I}_2, s_2 = seq_2(I), \forall a \in \chi(I)
\end{aligned}$$

The optimal solution to this program is quantal Stackelberg equilibrium, but since the program has nonlinear constraints, solvers used might end stuck in some local minima.

I used scipy [9] for the minimization with method SLSQP [10]. This method is for solving the constrained problem of the form.

$$\begin{aligned}
& \min_x f(x) \\
\text{subject to:} \quad & b(x) = 0 \\
& c(x) \geq 0 \\
& \mathrm{lb}_i \leq x_i \leq \mathrm{ub}_i && i = 1, ..., N
\end{aligned}$$

Where both f and c should be continuously differentiable. The solver uses an iterative approach with local search, but the search direction d is not a simple gradient, and for the problem, at iteration k it is computed by solving quadratic subproblem in a form

$$\begin{aligned} &\min_d f(x_k) + \nabla f(x_k)^T d + \frac{1}{2} d^T \nabla^2_{xx} \mathcal{L}(x_k, \lambda_k, \sigma_k) d \\ \text{s.t.} \quad &b(x_k) + \nabla b(x_k)^T d = 0 \\ &c(x_k) + \nabla c(x_k)^T d = 0 \end{aligned}$$

5.4 Decomposition

In this part, I used the subgame definition from [2]. I tried to create a working decomposition for both approaches.

5.4.1 Quantal Nash equilibrium

To find the trunk strategy in quantal Nash equilibrium without having to store all subgame strategies at once, I proposed an algorithm that is inspired by CFR-D. First I split the game to the trunk and the subgames. I initialize strategy for both players in the trunk to uniform and proceed to solve subgames. I solve one subgame at a time using current trunk strategy and CFR-QR then I update counterfactual values at the root of the subgame. After each subgame solved I perform an update of the trunk strategy for one player, in next iteration, I perform an update for the second player. In the trunk, one player uses CFR updates, and the other one updates its strategy as a counterfactual quantal response to the previous strategy with actual values.

Unsafe resolving is then used to resolve the subgame after playing. Trunk strategy reach probabilities for both players are taken, and the subgame is again solved using CFR-QR or even by decomposition again if it is still too large to handle the whole strategy at once.

Unfortunately, the algorithm does not always converge. I will show more results concerning convergence in Chapter 6.

5.4.2 Quantal Stackelberg equilibrium

I tried to apply decomposition to the sequence form program that I had. The first problem that I encountered was that the program has the flow of information from the root to terminal nodes for one player through the sequence probabilities and then from terminal nodes back to root through the constraints governing the quantal response. So when split to the subgame and trunk there are no values in the algorithm that can be directly put forward to the trunk from the subgame.

I tried to solve this by computing values at each node in the root of the subgame and then treating this as new terminal nodes. However, a much bigger problem that I encountered is that the algorithm is strictly minimizing using all the constraints it has. When I take the trunk with only constraints for the trunk and variables for the trunk the minimization problem is suddenly very different. Moreover, even when I know which strategies should be in the subgames, I set all the strategies in the subgame to the correct ones and generate the values the computed strategy in the trunk is incorrect.

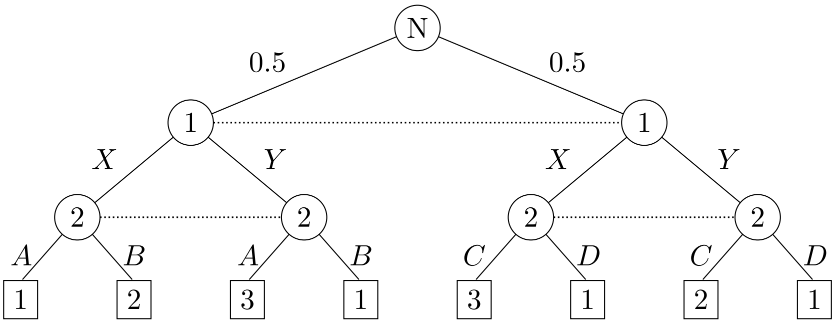


Figure 5.3: Simple game I used to test decomposition for quantal Stackelberg

That is because in the original problem changing the strategy changes the whole system and therefore I have to balance it to arrive at the local minima. With few values fixed the reaction to the strategy change is much smaller, and the resulting best strategy is mostly pure and incorrect.

Figure 5.3 shows a very simple game with one subgame where only the opponent acts. My player acts only in the trunk. Even in this game, any strategy in the subgame results in the pure optimal strategy in the trunk. In this case, the problem is obvious, when I fix the values from the subgame, the trunk strategy picks the better value, and the resulting strategy is pure. If I solve subgame again, the strategies change dramatically, and new values generate opposite trunk strategy. Thus, it goes like this infinitely.

One solution for future work that has arisen during the elaboration of my thesis is similar to CFR-D. Because I am using the solver that uses steps, it might help to solve the subgames and then perform one step of the trunk solving algorithm and iteratively continue like this.

Chapter 6

Evaluation

In this chapter I will present results from my experiments.

6.1 Equilibria analysis

First, I analyzed both equilibria that I defined in simple normal form games. I used normal form games with different parameters and showed the equilibria with different opponent rationality λ. I did this to get useful information for further work.

6.1.1 Shifting and scaling of payoffs in the game

The first result that I show is a problem with game normalization when quantal response opponent plays in the game as scaling utilities will change its strategy. Table 6.1 shows one game with shifted and scaled utilities and Table 6.2 shows corresponding strategies. Strategies were found by CFR-QR for quantal Nash equilibrium and gradient descent from Nash equilibrium for quantal Stackelberg equilibrium. Here I know that it is quantal Stackelberg equilibrium because there is only one local minimum in this game as shown in Figure 6.2. Rationality used for this experiment is $\lambda = 1$.

	A	**B**
X	1	3
Y	2	1

	A	**B**
X	2	4
Y	3	2

	A	**B**
X	2	6
Y	4	2

Table 6.1: Shift and scale example. From left to right: Original game, the game with shifted payoffs one up, the game with scaled payoffs by 2.

As shown in the example, shifting utilities does not change the strategy, and therefore, shifting is safe regarding quantal response. However, the same does not hold for scaling. The example shows that the game with scaled payoffs differs. Furthermore,

Game	Original				Shifted				Scaled			
Action	**X**	**Y**	**A**	**B**	**X**	**Y**	**A**	**B**	**X**	**Y**	**A**	**B**
QNE	0.33	0.66	0.90	0.10	0.33	0.66	0.90	0.10	0.33	0.66	0.78	0.22
QSE	0.42	0.58	0.78	0.22	0.42	0.58	0.78	0.22	0.42	0.58	0.72	0.28

Table 6.2: Strategies of both players in games from Table 6.1 in both quantal Nash equilibrium and quantal Stackelberg equilibrium.

the resulting strategy is the same as in the original game, where I set opponent rationality to $\lambda = 2$. It means that scaling payoffs by a constant have the same effect as multiplying λ by the same constant.

6.1.2 Example games

Games used in this part are handcrafted to fall into different categories as I needed to analyze different behavior present in games with fully mixed Nash equilibria and also games with dominated actions. These games are shown in Table 6.3 and properties of the games are that Game 1 has dominated action and pure Nash equilibrium. Game 2 has fully mixed Nash equilibria. Game 3 is bigger, has no dominated action but Nash equilibria is not fully mixed and Game 4 has fully mixed Nash equilibria.

	A	**B**
X	3	2
Y	-2	1

	A	**B**
X	1	3
Y	2	1

	A	**B**	**C**
X	4	1	2
Y	3	2	4
Z	2	3	2

	A	**B**	**C**
X	1	3	5
Y	4	1	2
Z	2	5	1

Table 6.3: Example games. From left to right: Game 1 - dominated action, Game 2 - fully mixed Nash equilibria, Game 3 - no action dominated but not fully mixed Nash equilibria, Game 4 - fully mixed Nash equilibria.

6.1.3 Nash equilibria

Table 6.4 shows Nash equilibria strategies for future comparison with quantal Nash equilibria and quantal Stackelberg equilibria.

	A	**B**	**C**	**X**	**Y**	**Z**	value
Game 1	0	1	/	1	0	/	2
Game 2	$\frac{2}{3}$	$\frac{1}{3}$	/	$\frac{1}{3}$	$\frac{2}{3}$	/	$\frac{5}{3}$
Game 3	0.5	0.5	0	0	0.5	0.5	2.5
Game 3	0.5	0.5	0	0.125	0.25	0.625	2.5
Game 4	0.452	0.290	0.258	0.290	0.452	0.258	2.613

Table 6.4: Nash equilibria of the games.

6.1.4 Quantal Nash equilibrium and quantal Stackelberg equilibrium examples

Since the strategy depends on the λ parameter in the quantal response, I show strategies in quantal Nash equilibrium and quantal Stackelberg equilibrium for multiple lambda parameters for comparison. Strategies are computed by CFR-QR for quantal Nash equilibrium and by gradient descent from Nash equilibrium for quantal Stackelberg equilibrium. In the tables, I do not use the same lambdas for both equilibria because I want to show changing strategies and since for both concepts, strategies change at a different rate and in different values of λ the values are different.

Game 1 examples

Tables 6.5 and 6.6 show that for small λ strategy is exactly opposite from the Nash equilibrium strategy, and it is the same for quantal Stackelberg equilibrium and quantal Nash equilibrium. As λ increases, strategies shift towards the Nash strategy, but for quantal Stackelberg equilibrium, it shifts faster than for quantal Nash equilibrium. From λ value 2 it is again the same, and the strategy slowly converges to Nash equilibrium for λ going to infinity. For the whole time, both solution concepts can exploit a quantal response opponent.

Results also show that for different λ value there is the same strategy, for instance in quantal Nash equilibrium for $\lambda = 0.3$ and $\lambda = 1$. This is not coincidence because for the whole time the strategy is shifting from $(1, 0)$ to $(0, 1)$ the quantal response strategy is $(0.628, 0.372)$. Same works for quantal Stackelberg equilibrium but with different λ values and strategies.

λ	**A**	**B**	**X**	**Y**	value
0.1	1	0	0.622	0.378	1.112
0.3	0.666	0.334	0.750	0.250	1.75
1	0.025	0.975	0.750	0.250	1.75
2	0	1	0.881	0.119	1.881
5	0	1	0.993	0.007	1.993

Table 6.5: Game 1 QNE

λ	**A**	**B**	**X**	**Y**	value
0.1	1	0	0.622	0.378	1.112
0.3	0.186	0.814	0.628	0.372	1.537
0.5	0.011	0.989	0.628	0.372	1.622
1	0	1	0.731	0.269	1.731
2	0	1	0.881	0.119	1.881

Table 6.6: Game 1 QSE

Game 2 examples

Results in Tables 6.7 and 6.8 show that quantal Stackelberg equilibrium and quantal Nash equilibrium strategies are same for small λ and differentiates as λ increases. There are some new observations in this case. First, quantal Stackelberg equilibrium stops playing the $(1, 0)$ strategy faster resulting in more exploiting of the quantal response. Second, as soon as the strategy in quantal Nash equilibrium starts changing, value of the game is already at the Nash equilibrium value of the game. Third, the strategy for quantal response is not changing since strategy for my agent started to change. Furthermore, in quantal Nash equilibrium it is Nash equilibrium strategy. So while trying to exploit quantal response the best response agent actually forces him to play Nash equilibrium strategy. This is caused by the fact that in order to play mixed strategy, best response agent needs to have same expected utility for both actions. Therefore, it needs the opponent to play Nash equilibrium strategy, because it is only strategy that accomplishes the requirement in this case. Quantal Stackelberg equilibrium also forces the quantal response adversary to play the same strategy, but in this case this strategy is exploitable.

λ	**A**	**B**	**X**	**Y**	value
0.1	1	0	0.475	0.525	1.525
0.5	1	0	0.378	0.622	1.622
1	0.898	0.102	0.333	0.667	1.667
10	0.690	0.310	0.333	0.667	1.667

Table 6.7: Game 2 QNE

λ	**A**	**B**	**X**	**Y**	value
0.1	1	0	0.475	0.525	1.525
0.5	0.893	0.107	0.416	0.584	1.611
1	0.780	0.220	0.416	0.584	1.639
2	0.678	0.322	0.416	0.584	1.664

Table 6.8: Game 2 QSE

Game 3 examples

I show results from bigger game with multiple Nash equilibria in Tables 6.9 and 6.10. Remarks from previous game can also be applied here, and the strategy in quantal Nash equilibrium that my agent forces upon the quantal response adversary is $(0.116, 0.268, 0.616)$. This strategy is also Nash equilibrium strategy because I list the corner cases in the table of Nash equilibria and this is convex combination of these two equilibrium strategies.

λ	**A**	**B**	**C**	**X**	**Y**	**Z**	value
0.1	0	1	0	0.301	0.332	0.367	2.067
1	0.083	0.917	0	0.116	0.268	0.616	2.5
10	0.458	0.542	0	0.116	0.268	0.616	2.5

Table 6.9: Game 3 QNE

λ	**A**	**B**	**C**	**X**	**Y**	**Z**	value
0.1	0	1	0	0.301	0.332	0.367	2.067
1	0.303	0.697	0	0.213	0.317	0.470	2.404
10	0.480	0.520	0	0.213	0.317	0.470	2.49

Table 6.10: Game 3 QSE

Game 4 examples

I have results from game with fully mixed Nash equilibria in Tables 6.11 and 6.12. All the observations can be verified on this game and also some new interesting facts arise. As can be seen the strategy for the opponent becomes fixed only after using all actions that are also in Nash equilibrium strategy and for quantal Nash equilibrium the same fact holds for value of the game.

λ	**A**	**B**	**C**	**X**	**Y**	**Z**	value
0.05	1	0	0	0.311	0.362	0.327	2.412
0.2	0.830	0	0.170	0.284	0.422	0.293	2.560
1	0.541	0.196	0.263	0.290	0.452	0.258	2.613
10	0.461	0.281	0.259	0.290	0.452	0.258	2.613

Table 6.11: Game 4 QNE

6.1.5 Conclusion

It is evident from multiple examples that quantal Stackelberg equilibrium and quantal Nash equilibrium are different. Quantal Stackelberg equilibrium agent can exploit quantal response adversary more than quantal Nash equilibrium depending on the parameter λ of the quantal response. When raising the parameter lambda, the game can get to the point where the quantal response strategy no longer changes. In quantal Nash equilibrium, it is the Nash equilibrium strategy for the adversary. This occurs in

λ	**A**	**B**	**C**	**X**	**Y**	**Z**	value
0.05	1	0	0	0.311	0.362	0.327	2.412
0.15	0.744	0	0.256	0.312	0.389	0.299	2.509
0.2	0.678	0.055	0.267	0.313	0.392	0.296	2.535
1	0.497	0.243	0.260	0.313	0.392	0.296	2.597
10	0.456	0.286	0.258	0.313	0.392	0.296	2.611

Table 6.12: Game 4 QSE

games where opponents strategy in Nash equilibrium can be fully mixed. This point is reached when my agent starts playing all actions that are in his Nash equilibrium strategy. For better understanding, I will show graphs of the actions and game value.

6.1.6 Game graphs

I show graphs of the example games for easier understanding of the concepts that I am solving in these games. Since the quantal response is always well defined, and only one, the expected utility in the game is directly dependent only on the actual strategy of my agent. This means that it can be shown as a function, depending on the strategy. For the graphs, I have chosen Game 2 because it is fully mixed and has only 2 actions for my player, so it can be easily shown on the 2D graph.

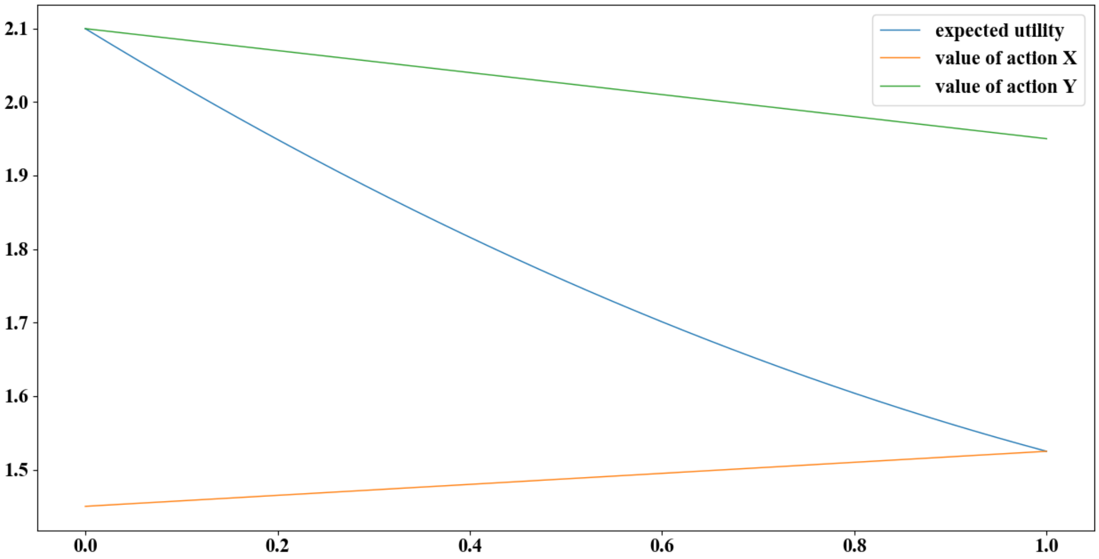


Figure 6.1: Game 2 lambda 0.1

6.1.7 Graph explanation

On X axis is my agent's strategy. 0 is strategy $(0, 1)$ and 1 is strategy $(1, 0)$. On Y axis is expected utility and the blue line is the expected utility for the whole game given strategy of my agent. Value of action X is the expected value of action X given the current strategy. In other words, it is the value of the game for pure X strategy when the opponent plays against the actual strategy. The same for value of the action Y.

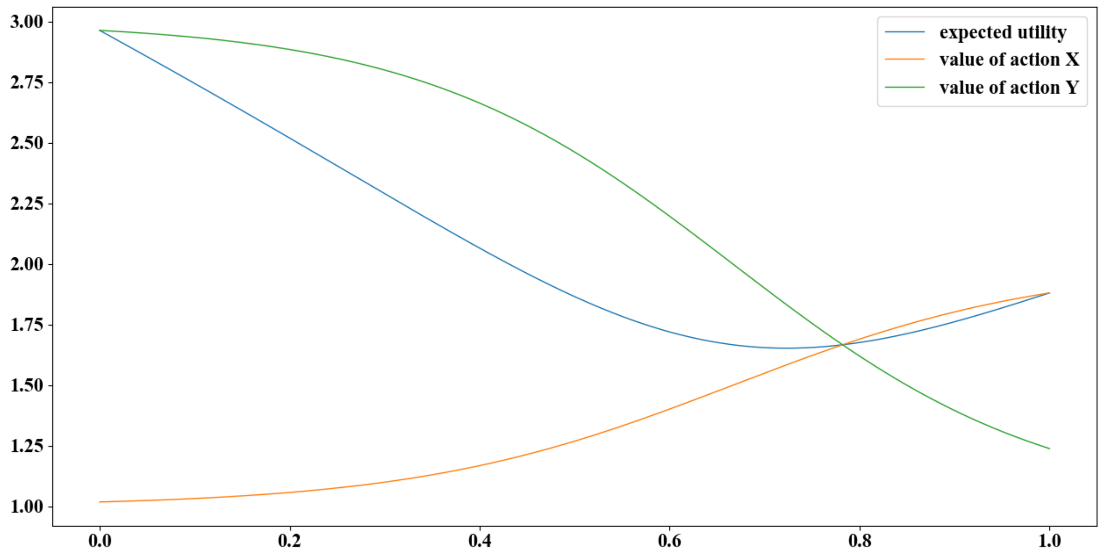


Figure 6.2: Game 2 lambda 2

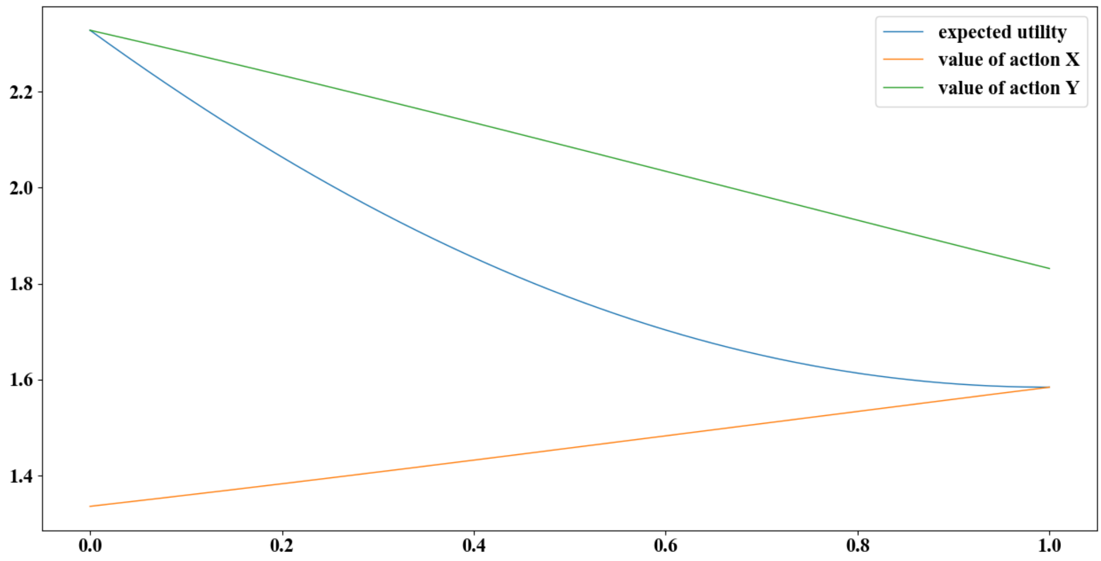


Figure 6.3: Game 2 lambda 0.3402

These graphs clearly show quantal Nash equilibrium as a point where action or both actions intersect value of the game. This is caused by the best response mechanic. Quantal Nash equilibrium simply can not use two actions if they do not have the same value. This fact leads to one action strategy or mixed strategy in the point of intersection. Quantal Stackelberg equilibrium is much simpler, it is just minimum on the game value function. So for $\lambda = 0.1$ quantal Stackelberg equilibrium and quantal Nash equilibrium is the same because graph minimum is in 1 and there is also an intersection of Y value and game value. For $\lambda = 2$, there is already intersection of all three, and the minimum is clearly somewhere else so quantal Stackelberg equilibrium and quantal Nash equilibrium differ. Moreover, with increasing λ both values move closer to the Nash equilibrium strategy, which is, in this case, $(0.66, 0.33)$ for our agent.

Finally, I isolated points where the strategy started to be mixed for both quantal

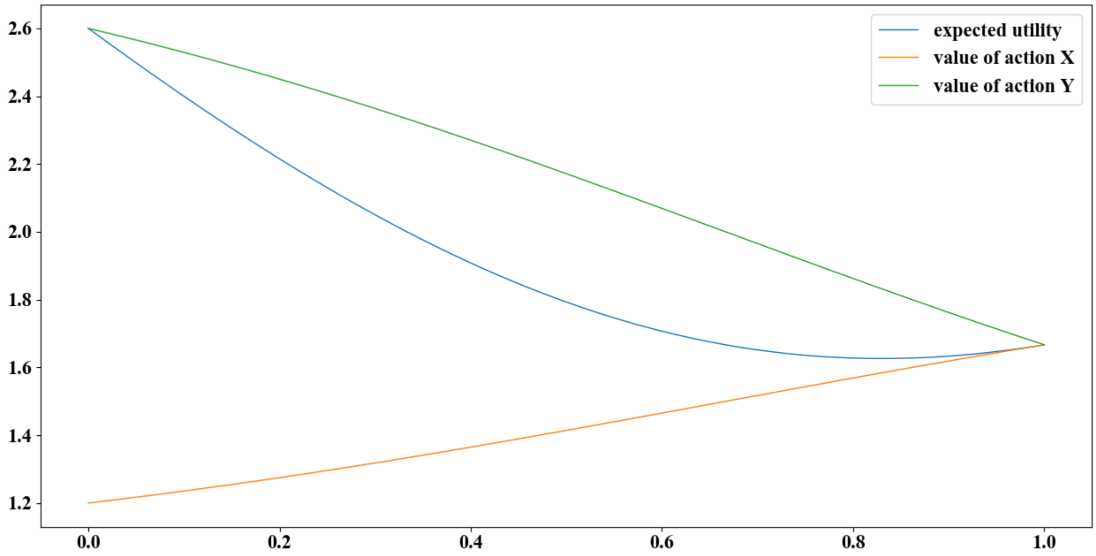


Figure 6.4: Game 2 lambda 0.693

	A	**B**	**C**
X	1	2	3
Y	2	2	2
Z	3	2	1

Table 6.13: Normal form game with two quantal Nash equilibria.

Stackelberg equilibrium and quantal Nash equilibrium and showed it in other graphs. For $\lambda = 0.3402$ quantal Nash equilibrium is still pure, but it is the point where the value of the game function is starting to curve upwards on the right, and therefore its minimum is moving to left. For $\lambda = 0.693$ both action values meet at X value of 1. As λ increases, the quantal Nash equilibrium moves towards the Nash equilibrium point. However, for the same lambda value, quantal Stackelberg equilibrium is already much closer to the Nash equilibrium strategy.

6.2 Normal form games

I will discuss the existence of multiple quantal Nash equilibria and Stackelberg equilibria In this part. I will then proceed to the evaluation of both algorithms on normal form games.

6.2.1 Multiple quantal Nash equilibria

There may be multiple Nash equilibria and quantal Nash equilibria in zero-sum games. Example game where there are two quantal Nash equilibria is in Table 6.13. This game is made in a way that playing $(0.5, 0, 0.5)$ by the rational column player produces the same quantal response as $(0, 1, 0)$ and both are best responses to the quantal response strategy $(\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$.

In the example, the two quantal Nash equilibria have the same value. I hypothesize that all quantal Nash equilibria in a game have to have the same value. Unfortunately, I was unable to prove the hypothesis. I tried to produce contradiction given two quantal Nash equilibria with different values using properties of quantal Nash equilibria. I also tried to produce counterexample that would fulfill all the requirements. However, I was unable to find such a game.

6.2.2 Quantal Nash equilibria results

In this section, I will show results on very small game and show that CFR-QR indeed finds quantal Nash equilibrium in this game. I perform and show this experiment to provide an example of a game where CFR-QR finds quantal Nash equilibrium.

	A	**B**	**C**
X	1	3	5
Y	4	1	2
Z	2	5	1

Actions	**A**	**B**	**C**
str.	0.541	0.196	0.263
E. val.	2.613	2.613	2.613

Actions	**X**	**Y**	**Z**
str.	0.290	0.452	0.258
E. val.	2.444	2.886	2.326
Exp v.	11.52	17.91	10.24

Table 6.14: Example of game solved by CFR-QR. Left: The game. Center: Strategy and expected values for actions for the rational player. Right: Strategy, expected values for action and their exponentials.

In Table 6.14 is an example game and also its strategy and expected values. This strategy is created by CFR-QR algorithm. As can be seen, expected values for the rational player are indeed all the same, so it is the best response. For the quantal response player, I also reported exponentials of the expected value. It can be easily computed that the strategy is quantal response with $\lambda = 1$.

This fact is true for all the games that I tested. I used CFR-QR to solve the game in this experiment. Then I checked whether strategy played by the rational agent is the best response, and if the strategy played by the opponent is a quantal response. I will also report the exploitation and exploitability of the resulting strategy for random games.

Convergence and speed

I proceeded tests with numbers of iteration, and I show different convergence curves of the algorithm in Figures 6.5 and 6.6. On X axis are the iterations of the algorithm. Figure 6.5 shows the expected value of the game based on strategies from the last iteration. Also, it shows the best response value to the strategy in each iteration. Finally, it shows expected value and best response value for quantal Stackelberg equilibrium strategy.

Figure 6.6 shows expected values for multiple games. These curves show that values stabilize around 40 iterations. I will use 100 iterations in speed tests for lower bound on speed to be sure the algorithm will converge.

Speed is reported in Table 6.15 and it shows that the algorithm scales very well with game size. I measured the speed for all generated games, and I report average. The speed is measured from the start of the algorithm with initialization to the point when the algorithm returns the strategy.

6.2.3 Quantal Stackelberg equilibria results

Because quantal Stackelberg equilibrium is very general minimization problem, checking whether I have reached the quantal Stackelberg equilibrium is as complex as finding

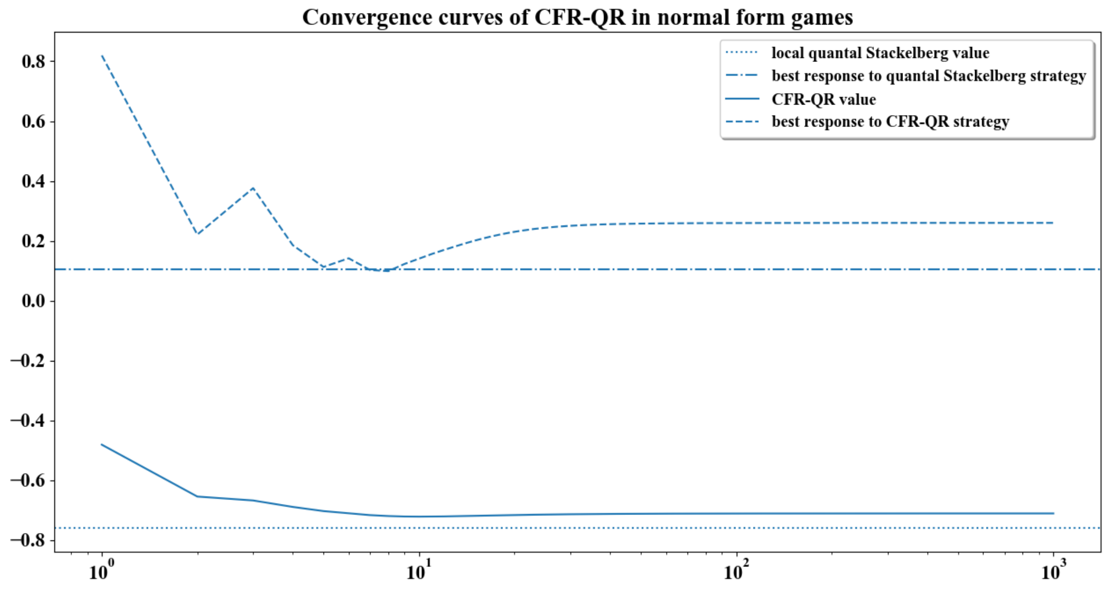


Figure 6.5: Convergence curves of CFR-QR on normal form game. Figure shows value after each iteration, best response to actual strategy and quantal Stackelberg equilibrium values for comparison.

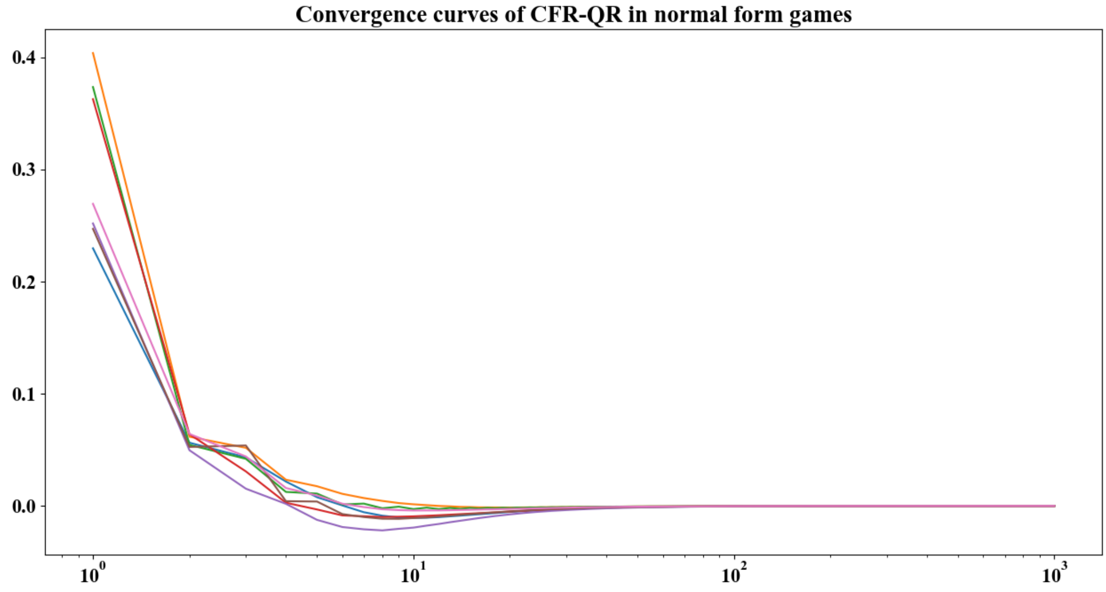


Figure 6.6: Values after each iteration of CFR-QR for multiple normal form games.

the equilibrium. Because of this fact I can only check the correctness on very small game, which I will deal with in the next paragraph. I will report exploitation of the opponent and also exploitability by rational player for bigger games. I will report these values for both quantal Stackelberg and quantal Nash equilibria.

The example game is shown in Table 6.16. For this game, the strategy found by my Gradient descent from Nash is $(0.78, 0.22)$ for the rational player and $(0.42, 0.58)$ for quantal response player. Graph of the value based on rational player strategy for $\lambda = 1$ is shown in Figure 6.7. The minimum is showed to be 0.78, which is exactly the strategy that my algorithm found.

Game size	2	3	5	8	13	21	34	55	89	144	2x8	2x16	8x2
Time [ms]	10	11	12	12	14	16	20	26	38	58	12	13	12

Table 6.15: Speed of CFR-QR algorithm on Normal Form games with 100 iterations. One number in Game size means that the game is square.

λ	**A**	**B**
X	1	3
Y	2	1

Table 6.16: Game used for quantal Stackelberg evaluation.

Problem with this solution is scalability, Nash equilibrium can be found very fast in zero-sum games using Linear programming, but function on which I am performing gradient descent is not convex. Therefore, for higher dimensions, the algorithm is very slow. Speed of the algorithm is reported in Table 6.17. Time is measured for the whole procedure, so it is both finding Nash equilibrium and the gradient descent. The time required is very small for small games as Gradient descent does not need to perform many iterations as CFR-QR does. However, at 144x144, the time required is already at almost three times as much as CFR-QR.

6.2.4 Testing on random games

I created games of different sizes, and for square games, I created $\frac{10000}{\text{game size}}$ games. I created 2500 of each category for rectangle games. I generated payoff as an integer from -10 to 9. I generated these games to test and compare both approaches. I also compare both approaches to the Nash equilibrium strategy playing against quantal response. This is lower bound, and my algorithms should not perform worse in terms of exploitation of the opponent.

Square games

Square games values are reported in Figures 6.9 and 6.8. For all graphs similar to these I will use the same notation. I will use RGD-QR for Gradient descent starting from 100 random starting points and quantal response to this strategy. NGD-QR is gradient descent starting from Nash equilibrium strategy and quantal response to the found strategy. NE-QR is Nash equilibrium strategy to which the opponent plays a quantal response. NGD-BR is strategy found by gradient descent from the Nash equilibrium strategy, but the opponent plays the best response. NE-BR is a Nash equilibrium. QNE-QR is quantal Nash equilibrium, and QNE-BR is quantal Nash equilibrium strategy with the best response as an opponent.

Game size	2	3	5	8	13	21	34	55	89	144	2x8	2x16	8x2
Time [s]	0.002	0.003	0.004	0.006	0.01	0.02	0.05	0.2	0.49	2.8	0.002	0.003	0.003

Table 6.17: Speed of gradient descent from Nash equilibrium on Normal Form games. One number in Game size means that the game is square.

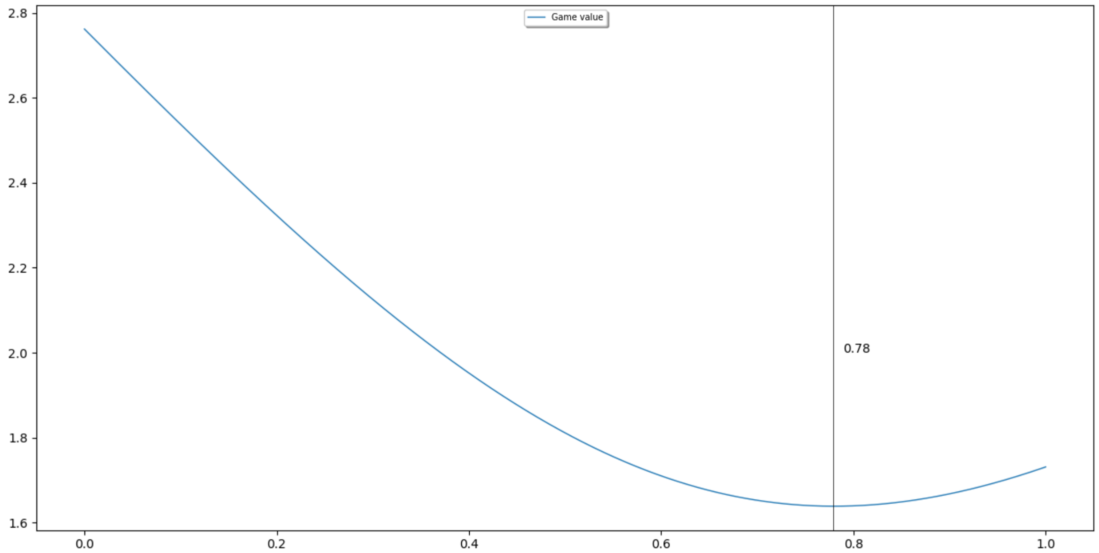


Figure 6.7: Value of the game from Table 6.16 showing strategy in quantal Stackelberg equilibrium.

For a group of games, I use average values in all solution approaches and graphs. For RGD-QR, I generate a random strategy for my agent, compute the quantal response, and then optimize using scipy [9] minimize with SLSQP [10] method. For NGD-QR I compute Nash equilibrium using linear program with Gurobi [7] solver. Then I use computed strategy as a starting point and optimize using the same tools as in RGD-QR. In NE-QR, I compute Nash equilibrium using a linear program with Gurobi and then compute the quantal response to the strategy of my agent. NGD-BR is computed by taking my agent strategy from NGD-QR and computing best response to it. NE-BR is computed using linear program and Gurobi. QNE-QR is computed using CFR-QR, and by taking the strategy of rational agent and computing best response against it, I get QNE-BR.

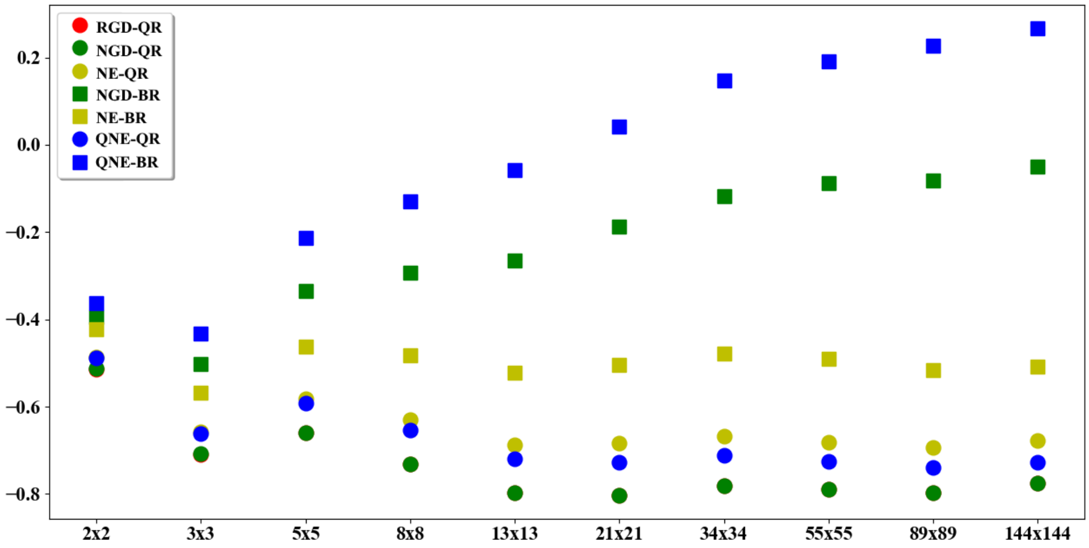


Figure 6.8: Values of different solution concepts in square games with rationality $\lambda = 1$.

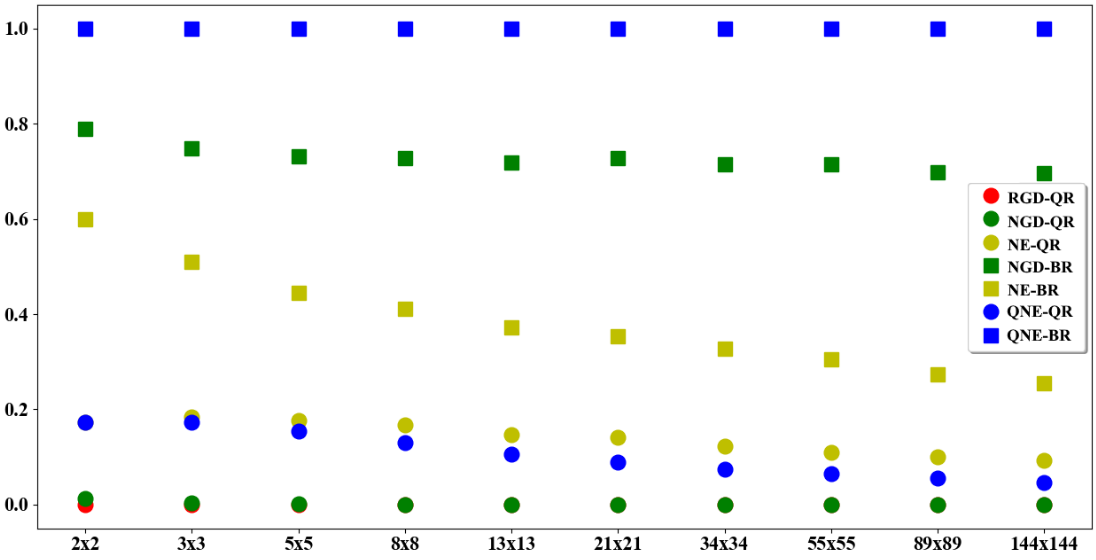


Figure 6.9: Normalized values of different solution concepts in square games with rationality $\lambda = 1$.

To summarize, squares are best responses, circles are quantal responses and colors indicate the same strategy for my agent. Difference between Nash equilibrium and Nash equilibrium strategy against quantal response shows how much I gain simply by the fact that the opponent has bounded rationality. Difference between Nash against quantal response and quantal Nash equilibrium and quantal Stackelberg equilibrium shows how much my agent can gain when expected rationality model is correct. Difference between Nash equilibrium and best response against both quantal Nash and quantal Stackelberg strategies show how much my agent can lose when the opponent is rational.

From the results it is evident that quantal Stackelberg equilibrium is overall better solution concept than quantal Nash equilibrium as not only my agent can gain more but also he loses less when the bounded rationality assumption is wrong.

I also show how the values change when I change the rationality of the quantal response opponent. In Figure 6.10 I show values against opponents with rationality $\lambda = 0.1$ and the trend is very similar to previous graphs except for that quantal Nash equilibrium and quantal Stackelberg equilibrium are much closer. Overall values are higher in terms of exploitation and exploitability.

In Figure 6.11 are values against opponent with rationality $\lambda = 10$ and here values are stacked very closely around corresponding Nash equilibria. Quantal Nash equilibrium is in this case very close to Nash against quantal response and ratio of exploitation against exploitability is very high. On the other hand, quantal Stackelberg is holding similar ratio even against a more rational opponent. Still, the biggest value difference is about 0.1 while in the $\lambda = 0.1$ scenario it is over 4.

The last graph in this section shows all values for 144x144 games sorted by Nash equilibrium value in Figure 6.12. It shows very well that the values are smooth and there are no wild extremes canceling each other in the average.

6.2.5 Rectangle games

I also tested 3 classes of rectangle games 2x8, 2x16 and 8x2. I only show normalized values in Figure 6.13 because when shown together unnormalized the difference between

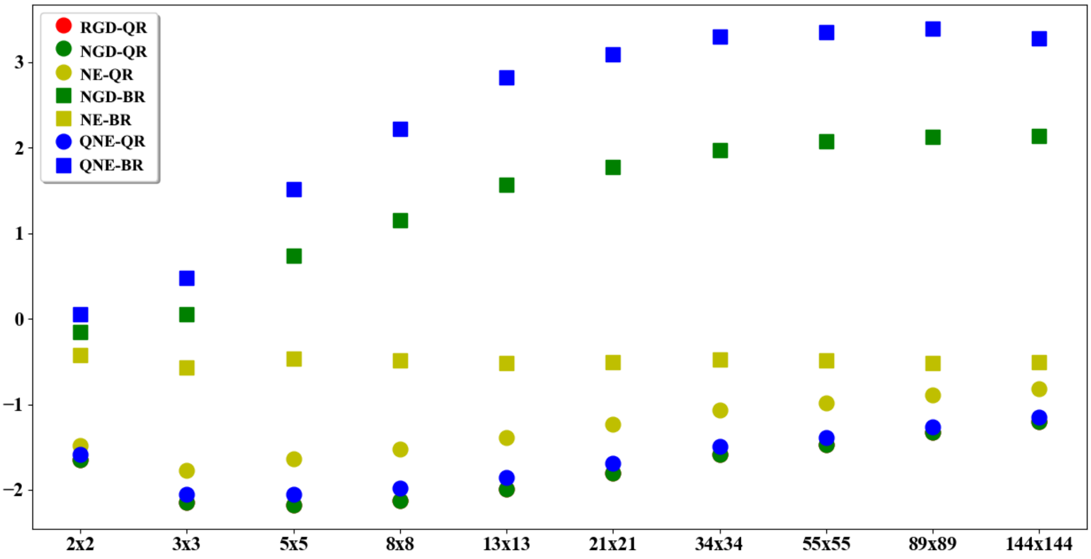


Figure 6.10: Values of different solution concepts in square games with rationality $\lambda = 0.1$.

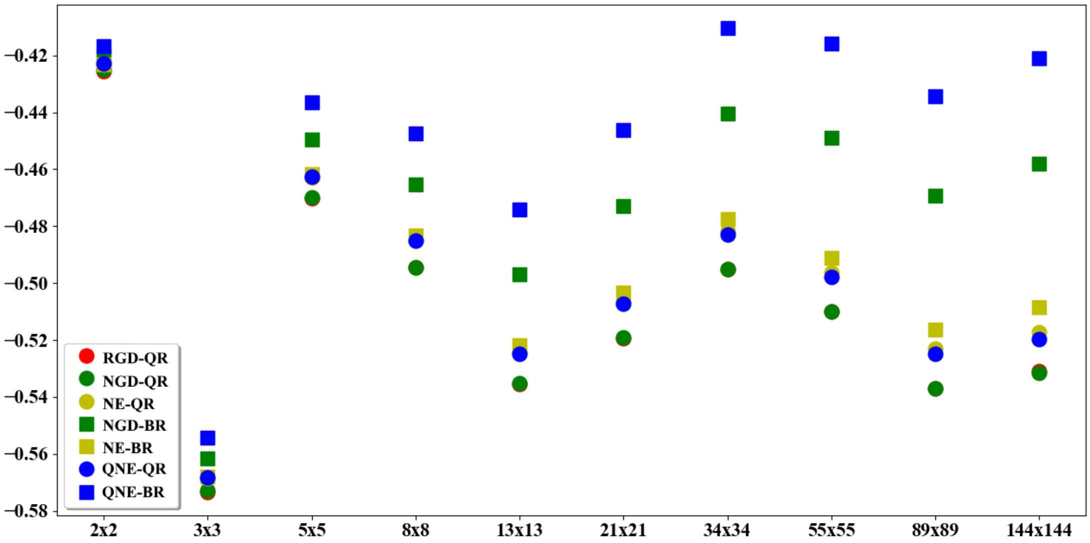


Figure 6.11: Values of different solution concepts in square games with rationality $\lambda = 10$.

the values in one game is very little compared to the difference between games. I used 2500 games in each class.

When my agent has fewer actions than the opponent, it is much harder for quantal Nash equilibrium to exploit the opponent while quantal Stackelberg equilibrium can still exploit relatively well. Also, when my agent has a low number of actions, it is less exploitable when exploiting the opponent.

6.2.6 Conclusion

As showed above it is obvious that quantal Stackelberg equilibrium is better solution concept than quantal Nash equilibrium. Even though I do not have an algorithm for

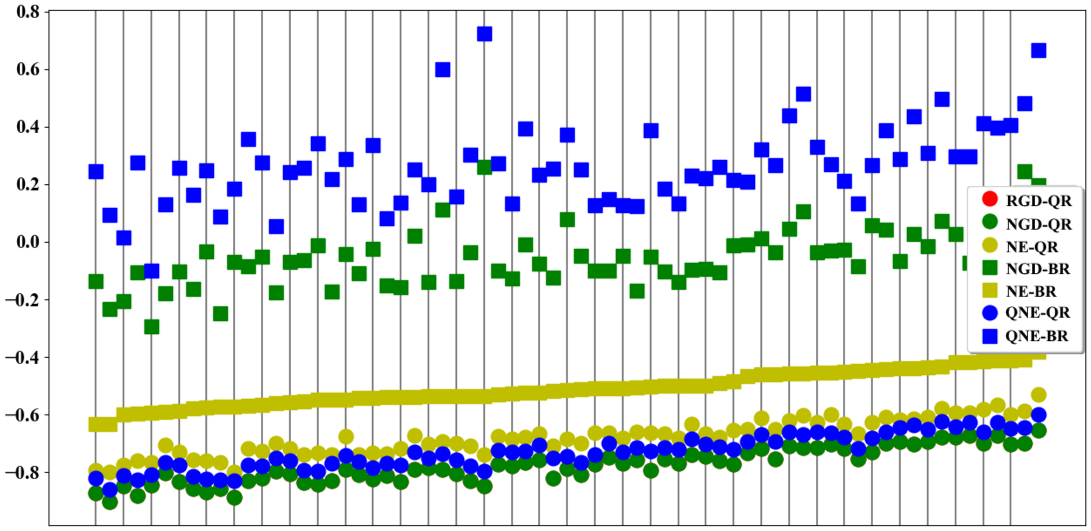


Figure 6.12: Values of different solution concepts in all generated games with size 144x144 with rationality $\lambda = 1$.

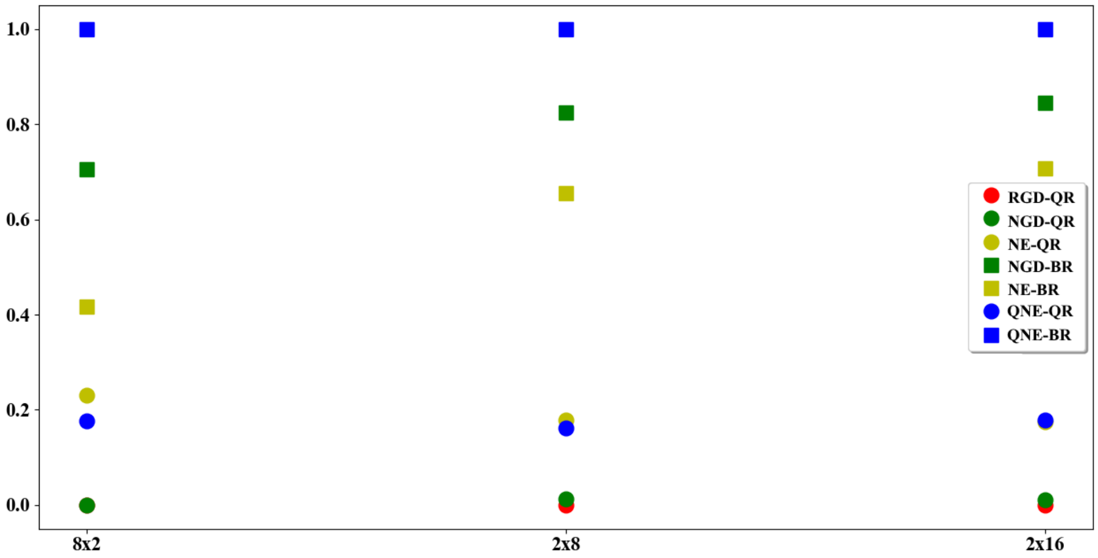


Figure 6.13: Normalized average values for different solution concepts in rectangle games.

exact solution and gradient descend may end in local quantal Stackelberg equilibrium, the solution is still better than quantal Nash equilibrium. Figure 6.12 shows that this also holds for all single games in 144 set and not only for the averages. However, the algorithm used to find local minima has very bad scalability compared to CFR-QR. CFR-QR can exploit quantal response opponents as well and can be efficiently computed. On the other hand, the cost for playing against an opponent that does not have bounded rationality can be great.

There is space for improvement in quantal Stackelberg equilibrium computation for future work. Firstly a big advantage would be to elaborate with an algorithm to solve the problem exactly. Secondly, at least create the approximation algorithm with better scalability than the current one.

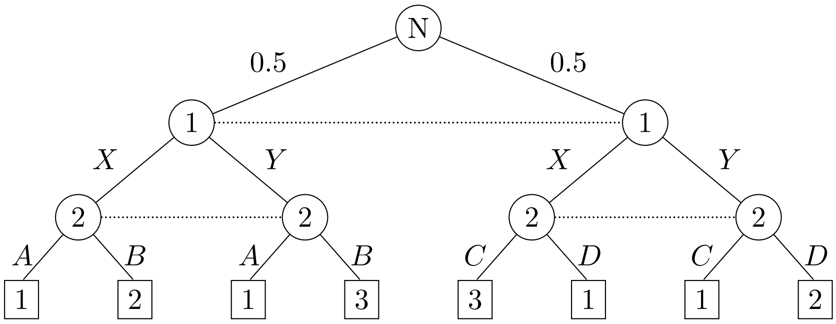


Figure 6.14: Extensive form game for testing of quantal Nash and quantal Stackelberg equilibria.

6.3 Extensive form games

In this part, I will evaluate the proposed sequence form program and CFR-QR on extensive form games. I will begin on smaller ones to test it directly, and then again, I will report values of different computed results compared to the Nash equilibrium strategy against quantal response. I will use the game shown in Figure 6.14 for the first tests for both solution concepts.

For time tests and exploitability tests, I use random games generated by framework from Game Theoretic Library [1]. I label my games as **mini**, **small**, **bigger** and **big**. All games generated by this framework are sequential, so one player acts, then the other player acts and so on. Game is generated randomly with a setting of depth, which governs the number of rounds being played, maximal branching factor and a maximal number of observations. **Mini** games have depth 1, which means that each player plays once maximal branching factor 3 and maximum observations also 3, so the game can be of perfect information. **Small** games have depth 2, maximal branching factor and observations also 3. **Bigger** games also have depth 2, but the branching factor is increased to 4 while still keeping maximal 3 observations, meaning that these games can not be perfect information anymore. Finally, **big** games have depth 3 and branching factor and observation also 4 and 3 respectively.

The utility is generated in such a way that utilities are correlated by the path from the root for all games. Generation of utilities proceeds as follows, at the root the value is set to 0 and with each node created the value from the parent is propagated to the node and can be changed by one up or down. This means that maximal and minimal utility is twice the depth of the game and that utilities on nodes that have the same parent can not differ by more than 2.

6.3.1 Quantal Nash equilibrium

In this section, I will show results on example game from Figure 6.14 to show that CFR-QR finds quantal Nash equilibrium in this extensive form game. This example is provided to explicitly show a game where CFR-QR finds quantal Nash equilibrium.

The strategies found by CFR-QR for the game are in Table 6.18. It is evident that the rational player plays the best response, and it can be easily computed that opponent strategy is quantal response with $\lambda = 1$.

This holds for all games from my testing set. I solved the game by CFR-QR and checked if the computed strategy fulfills the requirements of a Nash equilibrium.

Action	**X**	**Y**	**A**	**B**	**C**	**D**
str.	0.494	0.506	0.320	0.680	0.560	0.440
E. val.	1.899	1.899	0.5	1.253	0.993	0.753
Exp v.			1.649	3.501	2.702	2.123

Table 6.18: Strategy found by CFR-QR with expected values for actions and also exponential values for quantal response adversary.

Convergence

I also tested convergence and speed for CFR-QR in extensive form games. I show different convergence curves in Figures 6.15 and 6.16. On X axis are the iterations of the algorithm. Figure 6.15 shows the expected value of the game based on strategies from the last iteration. Also, it shows the best response value to the strategy in each iteration. Finally, it shows expected value and best response value for quantal Stackelberg equilibrium strategy. The first value in both cases is uniform strategy and quantal response against it.

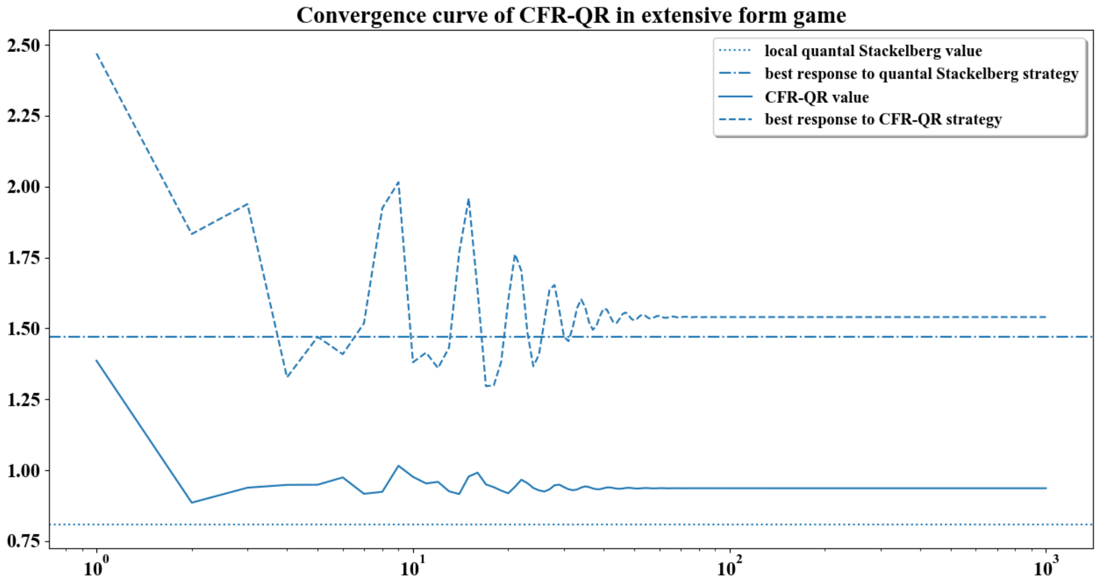


Figure 6.15: Convergence curves of CFR-QR on extensive form game. Figure shows value after each iteration, best response to actual strategy and quantal Stackelberg equilibrium values for comparison.

Figure 6.16 shows that values stabilize after 400 iterations. All curves are based on a set of games that I call **big** because it had the worst convergence speed of all my generated sets. I will use 1000 iterations in my speed tests to be sure that the algorithm will converge. Speed is reported in Table 6.19 along with quantal Stackelberg equilibrium solution speed.

6.3.2 Quantal Stackelberg equilibrium

Game value of game from Figure 6.14 based on actual strategy of the rational player is shown in Figure 6.17. I reported a minimum in the graph, which is a point that the quantal Stackelberg equilibrium finding algorithm should find. The strategy that

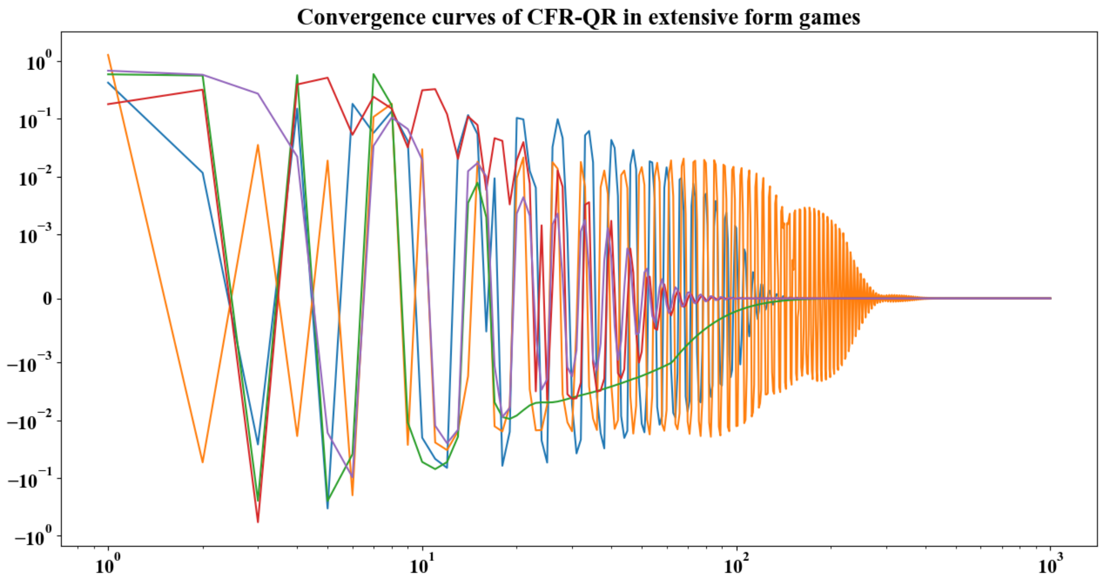


Figure 6.16: Values after each iteration of CFR-QR for multiple extensive form games.

sequence form program finds is $(0.487, 0.513)$. Thus, for this small game, the algorithm works correctly. For games with multiple local minima, it will find one. Because the starting point is from the Nash strategy, the value found will be less or equal to the Nash equilibrium strategy against quantal response.

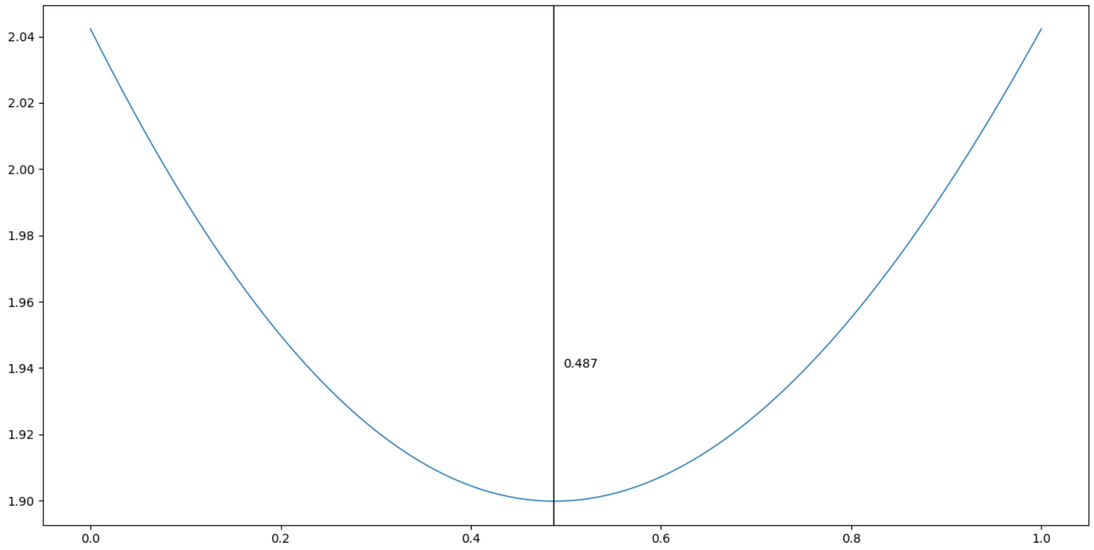


Figure 6.17: Game value of the game from Figure 6.14 based on rational player strategy.

Speed of the Algorithm is shown in Table 6.19. Scalability of sequence form program is bad as CFR-QR with 1000 iteration is 50 times faster for **big** games.

6.3.3 Testing on random games

I will evaluate how much I can gain practically by computing quantal Nash equilibrium and quantal Stackelberg equilibrium on random games. I will use the same notation of solution concepts as in normal form games. To remind NE-QR is Nash equilibrium

Size	**mini**	**small**	**bigger**	**big**
CFR-QR speed [s]	0.003	0.01	0.02	0.06
quantal Stackelberg equilibrium speed [s]	0.009	0.048	0.177	2.58

Table 6.19: Speed of CFR-QR with 1000 iterations and sequence form with quantal response on random extensive form games.

strategy to which the opponent plays quantal response. NE-BR is a Nash equilibrium. QNE-QR is quantal Nash equilibrium, and QNE-BR is quantal Nash equilibrium strategy with the best response as an opponent. However, the program for computing local quantal Stackelberg differs so NGD-QR is sequence program initialized from Nash strategy and quantal response to the found strategy. NGD-BR is strategy found by sequence program from the Nash equilibrium strategy, but the opponent plays the best response.

Nash equilibrium is computed in this case by linear sequence form program for zero-sum games, which is solved by scipy [9] minimize with the linear optimization method. My sequence form program with exponential constraints is solved by scipy minimize with SLSQP [10] method.

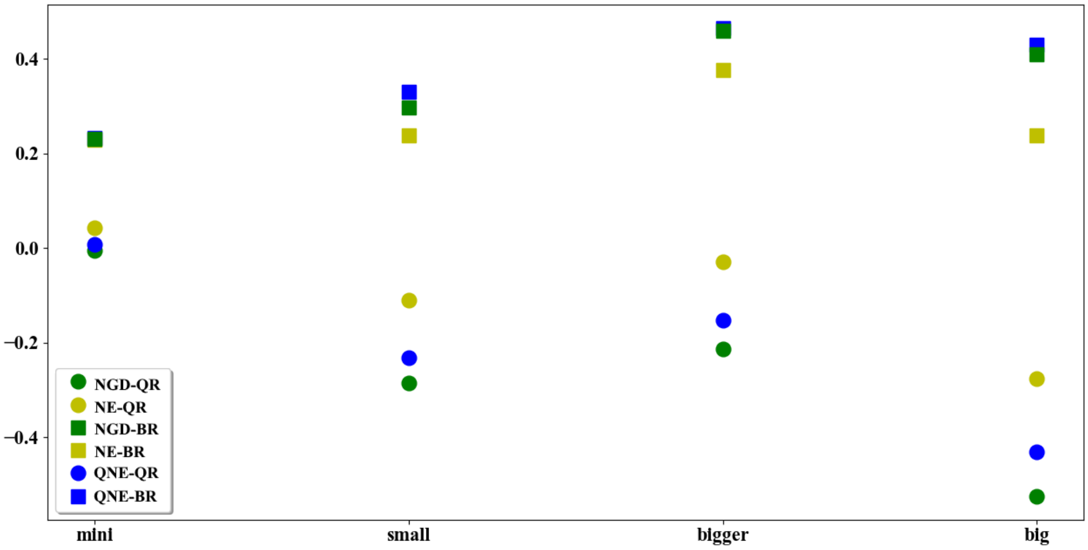


Figure 6.18: Average values of different solution concepts in randomly generated extensive form games with $\lambda = 1$.

Average values for the mentioned solution concepts are shown in Figure 6.18 and normalized in Figure 6.19. From these values, the exploitation and exploitability ratio is much better than for normal form games. Both quantal Nash equilibrium and local quantal Stackelberg equilibrium can exploit the opponent very well while still maintaining low exploitability in comparison to exploitation. However, as in normal form games, quantal Nash equilibrium is still worse in both parameters than solution created by sequence form program.

I also show results with increased rationality to $\lambda = 2$ in Figure 6.20. In this case, the local quantal Stackelberg equilibrium is still very similar in terms of a ratio of exploitation and exploitability while quantal Nash equilibrium is much worse, in this case, being so close in the **mini** set of games that because of numerical instability it is even shown above the Nash against quantal response. Thus, quantal Nash equilibrium ability to exploit is significantly reducing with increased opponent rationality.

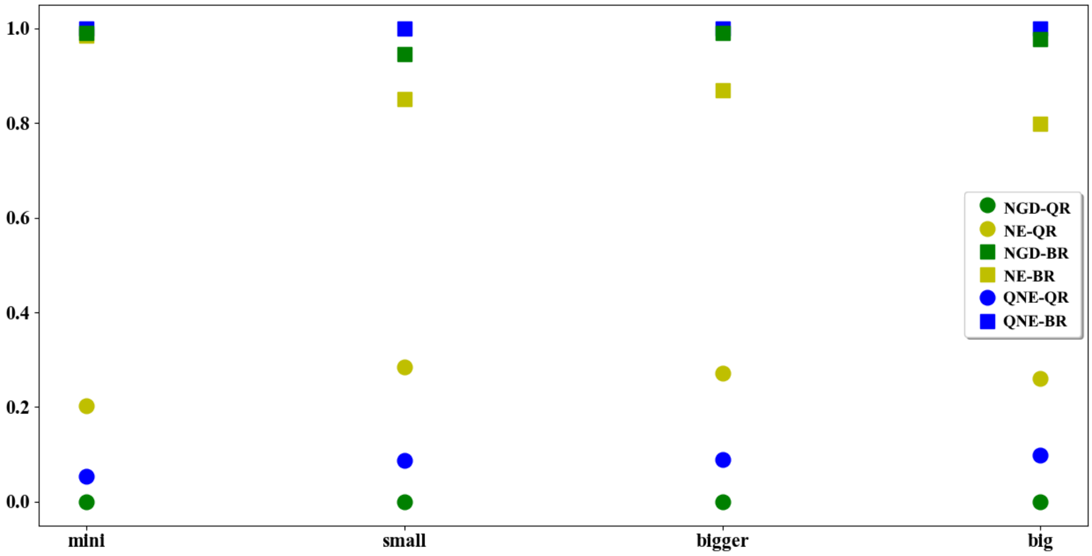


Figure 6.19: Normalized average values of different solution concepts in randomly generated extensive form games with $\lambda = 1$.

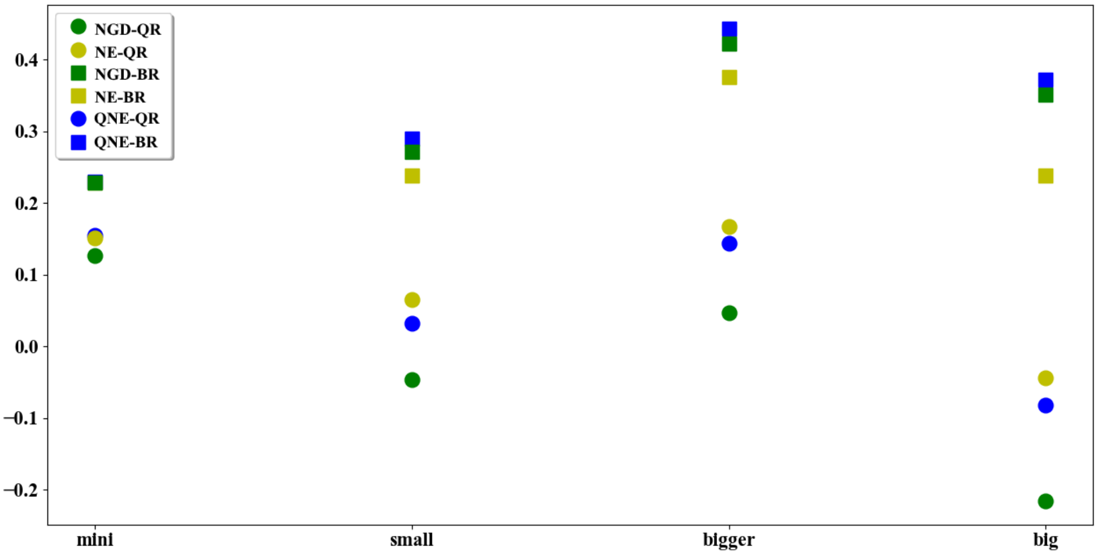


Figure 6.20: Average values of different solution concepts in randomly generated extensive form games with $\lambda = 2$.

Last overall values that I show are values with decreased rationality to $\lambda = 0.5$ showed in Figure 6.21. There the exploitation of the opponent is even bigger than the exploitability caused by playing the strategy. Also, values of quantal Nash equilibria and local quantal Stackelberg equilibria are very close, so if I knew with high probability that the opponent plays quantal response with low rationality, both solution concepts would be very good to use.

6.3.4 Conclusion

From the results, it is evident that quantal Stackelberg is better than quantal Nash equilibrium. Overall exploiting and exploitability ratio is better for extensive form

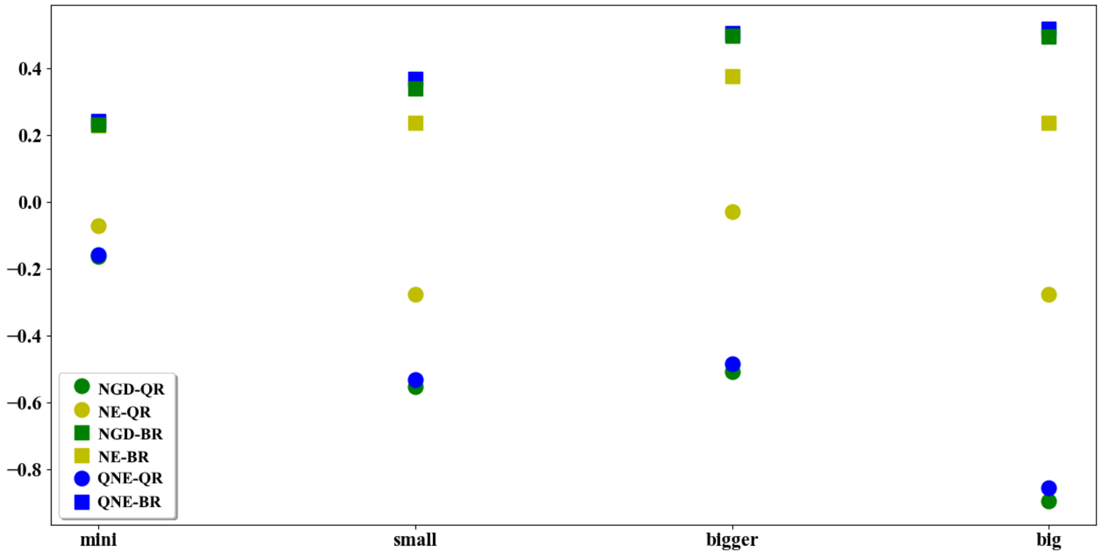


Figure 6.21: Average values of different solution concepts in randomly generated extensive form games with $\lambda = 0.5$.

games than it is for normal form games, probably because of higher complexity of the game and more steps where the opponent can make mistakes by playing quantal response.

The problem is that sequence program designed for finding local quantal Stackelberg equilibrium has very poor scalability and therefore can not be used to solve very large imperfect information games. On the other hand, CFR-QR scales very well, and even though the value gained is lower, it could be potentially used against low rationality opponents, as is shown in Figure 6.21.

6.4 Decomposition

Because I was not able to create reasonable decomposition algorithm for solving quantal Stackelberg equilibrium, I can only show results for the new algorithm for quantal Nash equilibrium decomposition. I call it CFR-QR-D and I tested this on **small**, **bigger** and **big** games. **Mini** games were not used because I wanted such subgame split where both players have actions in both trunk and the subgame.

6.4.1 Convergence

First, I will show the convergence curves to show a required number of trunk iterations. I will use 1000 subgame iterations because subgame is solved using CFR-QR. I already showed in Figure 6.16 that 1000 iterations are enough for games of the size that I am using. Convergence curves are in the Figure 6.22. I did not show the convergence curve of the game that does not converge, because it would not give information about the number of iterations I need. Curves show the values of the game after each iteration and are computed from the whole strategy. The first value is already after solving each subgame because I initialize the strategy when solving the subgame for the first time. Therefore, even the first values can be already close to the final solution.

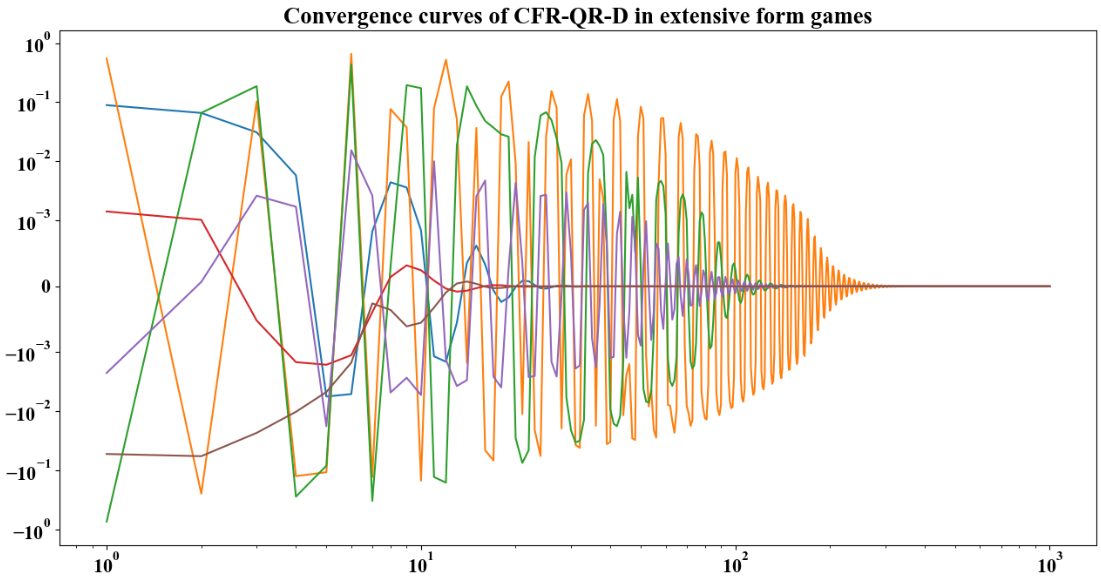


Figure 6.22: Values after each iteration of CFR-QR-D for multiple games.

Game size	**small**	**bigger**	**big**
Time [s]	427.42	660.82	1640.21

Table 6.20: Speed of CFR-QR-D algorithm using 100 trunk iterations and 1000 subgame iterations.

6.4.2 Algorithm speed

The speed of the algorithm is generally hard to measure because I can set two parameters that both influence the speed. First is the number of iterations in the trunk, and the second one is the number of subgame iterations. I used 1000 iterations in the trunk and 1000 subgame iterations. Values are shown in Table 6.20. Compared to CFR-QR speed follows the same trend as **bigger** is twice slower than **small** and **big** is approximately 3 times slower than **bigger**. This means that scalability is still good and it becomes even better when some heuristic function is used instead of solving the subgames, for example, neural networks.

6.4.3 Solution quality

CFR-QR-D converges for the majority of the games I tested. When the algorithm converges, then the solution after convergence is the same as CFR-QR. Unfortunately, in some cases, the algorithm was not able to converge. This happened in 11% games from small set, 16% games for **bigger** set and 14% games from **big** set. When analyzing, the algorithm gets to a point where it switches between few strategies and is unable to converge. I managed to increase the convergence rate to 99% when I stored average counterfactual values in the roots of the subgame. Instead of using newly computed counterfactual values after each subgame solved when computing trunk strategy, I use average counterfactual values in order to compute the trunk strategy.

I also tried to do iterations of the trunk at once or for one player and after solving another subgame for another player, which did not change the solution. I also tried to perform trunk update only after all subgames are solved one by one and that did not help either. Thus, for future work, I will try to get into the theory behind the algorithm, and I will try to make it finally converge in all cases.

6.4.4 Resolving

When I used unsafe resolving based only on the reaches to the subgame from the trunk strategy for both players, the strategy generated in the subgame was the same as the one generated by CFR-QR.

Chapter 7

Conclusion

I have worked on solving large imperfect information extensive form games with the assumption of bounded rationality of adversary. First, I showed related work from the domain of solving large imperfect recall games. Second, I showed related work from security games domain where opponent modeling and quantal response is already used to some degree of success.

In my work, I have defined two new solution concepts, quantal Nash equilibrium, and quantal Stackelberg equilibrium. I analyzed the basic properties of these concepts, and I showed that they are not interchangeable. I showed properties of strategies in these solution concepts for both rational agent and the quantal response adversary.

For normal form games, I showed that CFR-QR could be used to get the quantal Nash equilibrium strategy. For quantal Stackelberg equilibrium, I tested a new method which uses gradient descent from Nash equilibria. In average this method performs only slightly worse than restarted gradient descent but is much faster, and the resulting expected value is guaranteed to be less or equal to the value I gain by simply playing Nash equilibrium strategy against quantal response.

In extensive form games, I tested CFR-QR as an algorithm to compute quantal Nash equilibrium and created a sequence form program based on a linear program to solve Nash equilibria. Sequence form program starts from a Nash equilibrium strategy. Therefore, the resulting value cannot be higher than the Nash strategy played against quantal response adversary.

In tests, both methods aiming to compute quantal Stackelberg equilibrium perform better than CFR-QR in terms of both how much they can exploit the opponent and how much they can be exploited by a rational opponent. Unfortunately, both methods scale poorly compared to CFR-QR, thus for very large games, CFR-QR is still the only option.

Finally, I explored decomposition using mentioned algorithms for extensive form games, and I showed problems that are present when using the sequence form program. Therefore, I was unable to propose a working algorithm for quantal Stackelberg equilibrium. As for CFR-QR, I was more successful, and I developed CFR-QR-D based on CFR-D. CFR-QR-D converges, in most cases, and when it does the result is quantal Nash equilibrium. For some games, the algorithm, unfortunately, did not converge even though I tried many different versions. In the test, the algorithm converged for 99% of the games.

7.1 Future work

In normal form games, there is a place for improvement in the scalability of the gradient descent algorithm. Another option is using some non-local search methods which could find the global minima while having the upper bound from the Nash equilibria strategy starting point.

In extensive form games, the algorithm finding the quantal Stackelberg equilibrium is very slow, and it can be improved by using some linear approximation of constraints as is used in PASAQ [22] for computing the equilibrium in security games. Another possible approach is to explore the concept of online updates in each information set as CFR does. If possible, use it with gradient descent to compute the quantal Stackelberg equilibrium.

Concerning decomposition, there is an open problem with decomposition for quantal Stackelberg equilibrium, it may be worth to explore averaging of the values in the game, but it may lead back to the solution of CFR-QR.

For CFR-QR-D, I will try to improve the algorithm and test it on more games and in the long run, possibly in some real-world scenario.

Bibliography

[1] Branislav Bošanský, Jiří Čermák, Viliam Lisý, and Ondřej Vaněk. Game theoretic library, 2014.

[2] Neil Burch, Michael Johanson, and Michael Bowling. Solving imperfect information games using decomposition. In *Twenty-Eighth AAAI Conference on Artificial Intelligence*, 2014.

[3] Fei Fang, Thanh Hong Nguyen, Rob Pickles, Wai Y Lam, Gopalasamy R Clements, Bo An, Amandeep Singh, Milind Tambe, Andrew Lemieux, et al. Deploying paws: Field optimization of the protection assistant for wildlife security. In *AAAI*, pages 3966–3973, 2016.

[4] Gabriele Farina, Christian Kroer, and Tuomas Sandholm. Online convex optimization for sequential decision processes and extensive-form games. *arXiv preprint arXiv:1809.03075*, 2018.

[5] Drew Fudenberg and Jean Tirole. *Game Theory.* MIT Press, Cambridge, MA, 1991.

[6] Nicola Gatti and Marcello Restelli. Sequence-form and evolutionary dynamics: realization equivalence to agent form and logit dynamics. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[7] LLC Gurobi Optimization. Gurobi optimizer reference manual, 2018.

[8] Michael Johanson, Nolan Bard, Neil Burch, and Michael Bowling. Finding optimal abstract strategies in extensive-form games. In *AAAI*, 2012.

[9] Eric Jones, Travis Oliphant, Pearu Peterson, et al. SciPy: Open source scientific tools for Python, 2001–.

[10] Dieter Kraft. A software package for sequential quadratic programming. *Forschungsbericht- Deutsche Forschungs- und Versuchsanstalt fur Luft- und Raumfahrt*, 1988.

[11] Chun Kai Ling, Fei Fang, and J Zico Kolter. Large scale learning of agent rationality in two-player zero-sum games. *arXiv preprint arXiv:1903.04101*, 2019.

[12] Richard D McKelvey, Andrew M McLennan, and Theodore L Turocy. Gambit: Software tools for game theory. 2006.

[13] Richard D McKelvey and Thomas R Palfrey. Quantal response equilibria for normal form games. *Games and economic behavior*, 10(1):6–38, 1995.

[14] Richard D McKelvey and Thomas R Palfrey. Quantal response equilibria for extensive form games. *Experimental economics*, 1(1):9–41, 1998.

[15] Matej Moravčík, Martin Schmid, Neil Burch, Viliam Lisý, Dustin Morrill, Nolan Bard, Trevor Davis, Kevin Waugh, Michael Johanson, and Michael Bowling. Deepstack: Expert-level artificial intelligence in heads-up no-limit poker. *Science*, 356(6337):508–513, 2017.

[16] Thanh H Nguyen, Francesco M Delle Fave, Debarun Kar, Aravind S Lakshminarayanan, Amulya Yadav, Milind Tambe, Noa Agmon, Andrew J Plumptre, Margaret Driciru, Fred Wanyama, et al. Making the most of our regrets: Regret-based solutions to handle payoff uncertainty and elicitation in green security games. In *International Conference on Decision and Game Theory for Security*, pages 170–191. Springer, 2015.

[17] Thanh Hong Nguyen, Rong Yang, Amos Azaria, Sarit Kraus, and Milind Tambe. Analyzing the effectiveness of adversary modeling in security games. In *AAAI*, 2013.

[18] James Pita, Richard John, Rajiv Maheswaran, Milind Tambe, Rong Yang, and Sarit Kraus. A robust approach to addressing human adversaries in security games. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 3*, pages 1297–1298. International Foundation for Autonomous Agents and Multiagent Systems, 2012.

[19] Bernhard Von Stengel. Efficient computation of behavior strategies. *Games and Economic Behavior*, 14(2):220–246, 1996.

[20] Kevin Waugh. Abstraction in large extensive games. 2009.

[21] Rong Yang, Christopher Kiekintveld, Fernando Ordonez, Milind Tambe, and Richard John. Improving resource allocation strategy against human adversaries in security games. In *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*, volume 22, page 458. Barcelona, 2011.

[22] Rong Yang, Fernando Ordonez, and Milind Tambe. Computing optimal strategy against quantal response in security games. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 2*, pages 847–854. International Foundation for Autonomous Agents and Multiagent Systems, 2012.

[23] Martin Zinkevich, Michael Johanson, Michael Bowling, and Carmelo Piccione. Regret minimization in games with incomplete information. In *Advances in neural information processing systems*, pages 1729–1736, 2008.

Appendix A

User guide

Here I will explain how to use provided algorithms.

A.1 Requirements

My programs use python modules that are not present in common python distributions. Modules used are numpy, scipy, and for the linear program in normal form games even gurobipy.

A.2 Data

When using with own data, the program uses Gambit [12] representation for normal form and extensive form games. It accepts gbt format for normal form games and for extensive form games gbt and efg format.

A.3 Normal form games

Here I will explain how to use algorithms for normal form games. They are located in the folder `normal_form_games`.

A.3.1 CFR-QR

CFR-QR can be run on normal form games from file `main_cfrqr.py` and has four arguments that can be set. They are all optional, and the algorithm can run without setting any argument because there are default values. The first argument is `-f` that sets the path to the game file, default is example test normal form game. The second argument is `-i` which specifies the number of iterations performed by the algorithm, default is 100, and it must be an integer. Third is `-r` which sets the rationality for quantal response player, default is 1, and it must be a float. The last argument is `-v` with possible values 0,1,2 and 3. 0 shows only the game value after solving, 1 includes resulting strategies for both players, 2 also shows expected value for actions in the end, and 3 also shows average regret. Example:

```
python main_cfrqr.py -r=0.5 -i=50 -v=3
```

takes the example game and performs CFR-QR with 50 iterations against quantal response with rationality $\lambda = 0.5$ and reports the value of the solution, strategy for both players and expected values for the actions and also average regrets.

A.3.2 Gradient descent

Gradient descent to find quantal Stackelberg equilibrium can be started from file `main_gradient.py` and has four optional arguments. First `-f` specifies the path to game file, default is the example normal form game. The second argument is `-r` and sets the rationality of the quantal response opponent, this must be a float. The third argument is `-s` and sets whether to use Nash equilibrium strategy as starting point of the search or use randomly generated strategy, this is 0 for Nash strategy and 1 for random strategy, default is 0. And the last argument `-v` determines whether to show only resulting expected value of the game with argument value 0 or with value 1 also strategies for both players. Example:

```
python main_gradient.py -r=1.5 s=1 -v=1
```

takes the example game and preforms gradient descent from randomly generated strategy against quantal response with rationality $\lambda = 1.5$ and reports the value of the solution and strategy for both players.

A.4 Extensive form games

This part is focused on algorithms used with extensive form games that are located in separate folder `extensive_form_games`.

A.4.1 CFR-QR

CFR-QR on extensive form games can be run from `main_cfrqr.py` and has four arguments to set. All arguments are optional with default values set for example run. The first argument is `-f` which specifies the path to the game file, default is example test game. The second argument is `-i` which sets the number of iterations performed by the algorithm, default is 1000, and it must be an integer. Third is `-r` which sets the rationality for quantal response player, default is 1, and it must be a float. The last argument is `-v` with possible values 0,1,2 and 3. 0 shows only the game value after solving, 1 includes resulting strategies for both players, 2 also shows counterfactual values in the end, and 3 also shows counterfactual regret. Example:

```
python main_cfrqr.py -r=0.1 -i=100 -v=2
```

takes the example game and performs CFR-QR with 100 iterations against quantal response with rationality $\lambda = 0.1$ and reports the value of the solution, strategy for both players and counterfactual values.

A.4.2 Sequence program

Sequence program for computing quantal Stackelberg equilibrium can be run from `main_sequence.py` and has two arguments. Arguments are optional with default values so that the code can be simply run with no arguments. The first argument is `-f` which specifies the path to the game file, default is example test game and the second argument is `-r` and sets the rationality of the quantal response opponent, this must be a float. Example:

```
python main_sequence.py -r=2
```

runs the Sequence program on example game with rationality $\lambda = 2$.

A.4.3 CFR-QR-D

CFR-QR-D on extensive form game can be run from `main_cfrqrd.py` and has six arguments. All arguments are optional with default values set for easy running. The first argument is `-f` which specifies the path to the game file, default is example test game. The second argument is `-it` which sets the number of iterations int the trunk, default is 200, and it must be an integer. Third is `-is` which sets the number of subgame iterations, and the default number is 1000, it must be an integer. Fourth is `-r` which sets the rationality for quantal response player, default is 1, and it must be a float. The last argument is `-v` with possible values 0,1,2 and 3. 0 shows only the game value after solving, 1 includes resulting strategies for both players, 2 also shows counterfactual values in the end, and 3 also shows counterfactual regret. Example:

```
python main_cfrqrd.py -r=0.1 -it=10 -is=100 -v=1
```

takes the example game and performs CFR-QR-D with 10 iterations in the trunk and 100 iterations in subgames against quantal response with rationality $\lambda = 0.1$. Then it shows the value of the solution and strategy for both players.

Appendix B

CD structure

```
/
├── README.txt ...  description of folders and files and
│                   instructions how to run the program
├── data .........  data used by the program
│   └── ...
├── src
│   ├── normal_form_games ......  source files for algorithms on normal form
│   │   │                         games
│   │   └── ...
│   └── extensive_form_games ...  source files for algorithms on extensive
│       │                         form games
│       └── ...
└── text
    └── thesis.pdf ...........  text of the thesis in pdf
```