Czech Technical University in Prague

Faculty of Electrical Engineering

Algorithms for Playing Multi-player Simplified Poker

Master's thesis

Bc. Martin Münch

Supervisor: Mgr. Branislav Bošansky, Ph.D.

May, 2017

Czech Technical University in Prague
Faculty of Electrical Engineering

Department of Computer Science

DIPLOMA THESIS AGREEMENT

Student: Münch Martin

Study programme: Open Informatics
Specialisation: Artificial Intelligence

Title of Diploma Thesis: Algorithms for Playing Multi-player Simplified Poker

Guidelines:

Computer Poker is a well-known experimental domain for comparing algorithms for computing equilibria in sequential games. However, most of the research effort focused on the two-player variant of the game and three-player variants of Poker received significantly less attention. The goal of the student is to (1) select simplified variants of Poker that have been studied in the current literature, (2) analyze and implement at least two algorithms for computing game-theoretic solution concepts in three-player Poker games, (3) experimentally compare the quality of game-theoretic strategies with the strategies that are the result of the CFR algorithm.

Bibliography/Sources:

[1] R. Gibson. „Regret minimization in games and the development of champion multiplayer computer poker-playing agents". Ph.D. Thesis. University of Alberta, 2014
[2] R. Gibson. „Regret Minimization in Non-Zero-Sum Games with Applications to Building Champion Multiplayer Computer Poker Agents." arXiv, 2013
[3] D. Szafron, R. Gibson, N. Sturtevant. „A Parameterized Family of Equilibrium Profiles for Three-Player Kuhn Poker". AAMAS 2013

Diploma Thesis Supervisor: Mgr. Branislav Bošanský, Ph.D.

Valid until the end of the summer semester of academic year 2017/2018

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
Fakulta elektrotechnická
L.S.
4

Prague, January 16, 2017

Abstract

Poker is a popular scenario of the game theory, and it is used as the example of sequential finite games, where players do not share the same information. In this thesis, we use smaller poker variants with three players to compare and examine different approaches to computing strategies. We study a strategy computed by Counterfactual Regret minimization algorithm, which is a very popular algorithm for three-player computer poker players, in comparison with strategies computed by the concepts of MaxMin strategy and Stackelberg equilibrium. We use a tournament based methods to compare and examine computed strategies for Kuhn Poker and Leduc Hold'em. We have shown, it is a hard task to find global optima for Stackelberg equilibrium, even the three-player Kuhn Poker. We present a way to compute MaxMin strategy with the CFR algorithm. The tournaments suggest the pessimistic MaxMin strategy is the best performing and the most robust strategy. There is a tie for the second place between Stackelberg equilibrium strategy and CFR strategy, where different approaches lead to similar total average payoffs. The experiments also revealed the importance of different positions around the poker table and significance of ordering of players.

Abstrakt

Poker je populární doména v teorii her a je využíván jako příklad sekvenční konečné hry, kde hráči nesdílejí všechny informace. V této práci používáme zjednodušené varianty tříhráčového pokeru k porovnání a analýze různých přístupů pro výpočet strategií. Zabýváme se Counterfactual Regret minimization algoritmem, což je populární algoritmus pro počítačové hráče pro tříhráčový poker. Strategie z CFR algoritmu porovnáváme se strategiemi vypočítaných podle konceptu MaxMin strategie a strategií podle Stackelbergova equilibria. Strategie pro Kuhn Poker a Leduc Hold'em porovnáváme na základě turnajů. Ukázali jsme, že výpočet globálně optimální strategie dle Stackelbergova equilibria je složitá úloha i pro malou doménu jakou je Kuhn Poker. Dále jsme představili způsob jak vypočítat MaxMin strategii za použití CFR algoritmu. V turnajích se nejlépe umístila pesimistická MaxMin strategie, která se zároveň zdá nejvíce robustní strategií. O druhé místo se dělí strategie dle Stackelbergova equilibria a CFR strategie, kde různé přístupy dosáhly podobného průměrného zisku. Během experimentů se projevila důležitost pozice kolem pokerového stolu a také význam uspořádání hráčů.

Prohlášení autora práce

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze dne ....

Podpis autora práce

Acknowledgements

Firstly, I would like to thank my supervisor Mgr. Branislav Bošansky, Ph.D. for his guidance through the field of Game Theory and Poker for the second time. This thesis would be half of quality without his valuable insight and mentorship.

Secondly, I must express my gratitude to my parents and my nearest friends. Their support and encouragement made this thesis and years of study possible and survivable.

Finally, Access to computing and storage facilities owned by parties and projects contributing to the National Grid Infrastructure MetaCentrum provided under the program "Projects of Large Research, Development, and Innovations Infrastructures" (CESNET LM2015042), is greatly appreciated.

Contents

List of Algorithms

List of Figures

List of Tables

Chapter 1

Introduction

Field of game theory dates it beginning to the 40's and 50's of the previous century, with work of O. Morgenstern and Von Neumann and much more. Game of poker is a domain used since the beginning of the game theory field, for example, articles from 1950 for two-player poker [11] and three-player poker [16]. Poker is a card game, where imperfect information is represented by hole cards, cards known only to the players who is holding them, shared information, represented by table cards, known to all players and various betting actions, in which players take turns. It is also known as the game of bluffing and outplaying the opponents. This domain allows us to examine algorithms and solution concepts for the multiplayer imperfect-information perfect-recall sequential games.

Computational game theory combines mathematics and informatics, using algorithms and computers to find strategies according to theoretical concepts. This field is gaining popularity in last years, mainly thanks to successes in defeating human players in chess [13], Go [5] and also in one of the most popular poker variant Texas Hold'em, in the two-player limit game [8] and the two-player no-limit game [7].

These achievements have happened in the two-player games, where the position of both players is in the direct opposition. We focus only on three-player games, where this opposition is not so clear and, for example, two players can have the same interest and gain together from the loss of the third player. The indirect opposition also contributes to increased computational complexity of strategies, than in two-player games.

For computations of strategies we use Kuhn poker and Leduc Hold'em as our domains. Both variants have a small set of possible cards and limited bets. The Kuhn poker is a one-round poker, where the winner is determined by the highest card. Leduc Hold'em is a two-round game with the winner determined by a pair or the highest card.

In general, game theory focuses on describing scenarios containing players, payoffs, actions. Players take actions according to their knowledge or belief of the state of the game and usually try to maximize their payoffs at the end. Besides the description of the scenario, it aims at finding a solution for games. A solution of a game from game theory perspective is a strategy. There are two most used strategy types; pure strategy picks exactly one action for each decision node. On the other hand, mixed strategy is a probability distribution over actions in each decision node. Simple games we can imagine and use as examples are tic-tac-toe, chess, poker, prisoner's dilemma or go. Besides these there are multiple practical applications such as patrolling subway for passengers without a valid ticket, planning patrols and checkpoints on airports, preventing pirates to take over of cargo ships in Somalia.

Games differ in numerous aspects, cooperative vs. competitive, complete vs. incomplete information, simultaneous vs. turn base. For finding a strategy we usually use the concept of equilibria, which is a strategy advantageous for every concerned player. In this thesis, we are using two known equilibria: Nash equilibrium and Stackelberg equilibrium. Equilibrium is a set of stable strategies, where no player wants to change her own strategy. In Nash equilibrium, all players play best possible strategy to strategies of other players, therefore if one player changes her strategy, she will get the same payoff or worse. Stackelberg equilibrium is the solution concept, where one player (leader) publicly commits to a strategy and the rest of the players (followers) playing the best response against the leader's strategy.

We have these equilibria and maxmin as theoretical solution concepts, but they are very hard to compute, and we can not be even sure, how they will work in the three-player game because there are no theoretical guarantees in the current state of the art.

In this thesis, we work with three main strategies. One with computed with the Counterfactual Regret Minimization algorithm (abbreviated as CFR in this thesis). The first reason for using the CFR is that it is proven the algorithm converges to Nash equilibria for two-player perfect-recall zero-sum games. Another reason is, the CFR is commonly used for computing bots for three-player Kuhn poker (see [14] and [9] as an example). Next strategies are computed as heuristical maxmin, which can be viewed as computing a strategy against both opponents playing together. We sum the payoff for the other players and run the CFR algorithm for this situation. The maxmin strategy seems pessimistic, but it will hopefully lead to safe and careful strategy. The last strategy is computing a strategy according to a Stackelberg equilibrium, which we do by solving a non-linear mathematical program. As far as our knowledge goes, there is no comparison between CFR strategies, MaxMin strategies, and strategies according to Stackelberg equilibrium.

1.1 Thesis outline

In first two chapter, we start with a brief overview of used terms from game theory, following the description of used algorithms and methods for computing the strategies. Afterward, in Chapter 4 we describe our modifications of algorithms, to compute the strategies for three-player poker. Nextly, we describe poker, implementation details, compare results of CFR algorithm with ϵ-Nash equilibrium and Nash equilibrium and show limitations of computation Stackelberg equilibrium by nonlinear programming. Then we present the tournament-based comparison of strategies for small poker variants with insight on strategies behavior. At last, we sum up the algorithms, our modifications, results of tournaments in the Conclusion.

Chapter 2

Game Theory

Games are situations where players take actions that lead to outcomes. They can represent conflict, negotiation, company mergers or board games. Every game or domain has its parameters and properties, but all games have some things in common.

Games are played by players using their actions to influence the game and to reach an outcome. Players usually act based on some strategy, rule-based system on how to choose an action. Playing actions can be either sequential, when players take turns, or simultaneous when players play their action at once. Action can be the purchase of an estate in the game of Monopoly, placing a piece in Domino or moving a showing a sign in the rock-paper-scissors game. Games also vary in their length, compare prisoner's dilemma with chess. Prisoner's dilemma is a situation, where two players choose from two actions (confess or remain silent) and then the game is concluded and years in prison dealt. Chess is a game, where two players take turns, during which they move one piece on the chess board. Game of chess can take from 3 turns to something around 40 turns (there is no upper limit, but it is not probable to play a longer game then 80 turns).

Games in common view have a winner or winners at the end. Game theory describes the conclusion of games in more detail by the utility function, which tells us what each player receives for playing actions they played. Utilities distributed at the end do not have units; they can be view as points, -1 for the loss and 1 for the win or they can represent an amount of chip won/lost after the game of poker. When utilities for all players sum to zero in each possible outcome of the game, the game is called the zero-sum game. Otherwise, it is a general-sum game.

Some games have hidden information from a player, player's hand cards as in poker, player's positions as in battleships. These are the games with imperfect information.

In this chapter we formalize games as *extensive-form game*, define strategies and solution concepts as *best response*, *Nash equilibrium* and *Stackelberg equilibrium*. As the source of formal definitions we have used [15], [12], [19], [17], [9]

2.1 Games

Poker is a turn-based game, with a chance player dealing hole cards known only to the players and table cards, known to all. We use the extensive form to describe the game of poker because it is the most suitable form of representation. The extensive form can be viewed as directed tree (see following Example 1), where vertices represent the states of the game and the edges represent actions taken from the game state. Vertices grouped into sets, which represent the knowledge as Information sets. Nodes in the information set are indistinguishable for the acting player. The payoff function for every possible leaf (sometimes called terminal node).

Example 1. *This is an example of the game tree for the two-player imperfect-information game. It can model a situation, where both players secretly choose from the same set of actions* $\{a, b\}$ *and the utilities are then distributed according to this choice. The first player (the circle node) has one information set* $I1$ *containing one node, the second player (the square node) also has one information set* $I2$ *containing two nodes. The* $I2$ *signals us that the second players do not know the action player one took. Numbers in brackets are utilities in the form* (u_1, u_2).

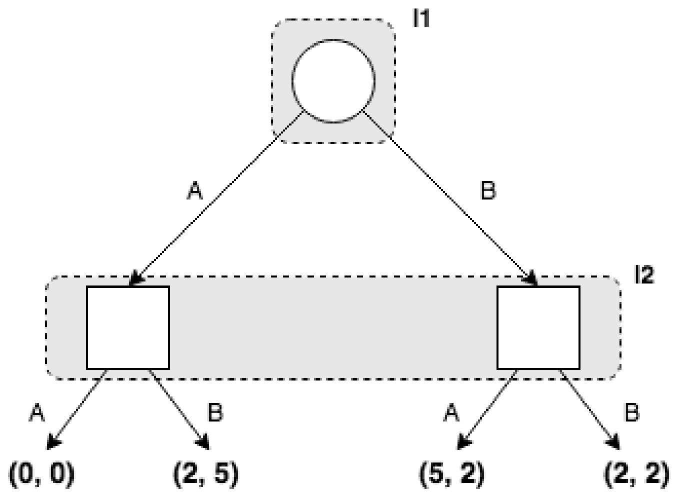

Figure 2.1: Game tree example

We use term history for identifying the state of the game. The history is a specific sequence of actions taken from the root node of the tree. The empty history, or history leading to the root node, is denoted as $\emptyset$. We use H as a set of all histories, and Z is a set of all terminal histories $Z \subseteq H$. When the player is in node defined by $h \in H$ and takes action a, we write it as $h' = ha$.

We formally define the extensive form as follows:

Definition 1. *A imperfect-information game in extensive form is a tuple* $G = (N, A, H, Z, \chi, \rho, \sigma, u, I)$*, where:*

- N *is a set of players*
- A *is a set of actions*
- H *is a set of all histories*
- Z *is a set of terminal histories,* $Z \subseteq H$
- $\chi : H \backslash Z \to 2^A$ *is the action function, which assigns to each choice node a set of possible actions*
- $\rho : H \backslash Z \to N$ *is the player function, which assigns to each nonterminal node a player* $i \in N$ *who chooses an action at that node*
- $\sigma : H \times A \to H \cup Z$ *is the successor function, which maps a choice node and an action to a new choice node or terminal node such that* $\forall h_1, h_2 \in H, \forall a_1, a_2 \in A : \sigma(h_1, a_1) = \sigma(h_2, a_2) \to h_1 = h_2 \wedge a_1 = a_2$
- $u = (u_1, \ldots, u_n)$*, where* $u_i : Z \to R$ *is a real-valued utility function for player* i *on the terminal nodes* Z.
- $\mathbb{I} = (I_1, \ldots, I_n)$*, where* $I_i = (I_{i,1}, \ldots, I_{i,k_i})$ *is a set of equivalence classes on (i.e., a partition of)* $h \in H : \rho(h) = i$ *with the property that* $\chi(h) = \chi(h')$ *and* $\rho(h) = \rho(h')$ *whenever there exists a* j *for which* $h \in I_{i,j}$ *and* $h' \in I_{i,j}$

Information sets are used for identifying histories, which are indistinguishable for the acting player. Since these states look the same for the player, she has the same set of possible actions in each state from the same information set. See Example 1 with two information sets, one for each player. Information set for the circle player **I1** contains only one node. The information set for square player **I2** contains two nodes and show, the square player does not know, which action the circle player took.

For information set, we define the function $A(I)$, which returns all possible actions for the acting player.

Dealing cards in the poker is a random event in the game. We use a nature player, to model these random events. This player receives no payoff and plays according to the mixed strategy, known to all players.

Games can also be distinct by what players know about previous states. If player remembers all previous actions, the game is called perfect recall. Otherwise, if she forgets some part of history, the game is called imperfect recall. More formally:

Definition 2. *An extensive-form game has perfect recall if for every player $i \in N$, for every information set $I \in \mathbb{I}_i$, and for any $h, h' \in I, X_i(h) = X_i(h')$. Otherwise, the game has the imperfect recall.*

2.1.1 Strategies

The strategy is a very important concept of the game theory. Strategy for given player defines a plan for all information sets of that player, where it specifies what actions to play. The strategy is a very common result of algorithms/mathematical programs used in computational game theory. The simplest strategies are Pure Strategies, Which prescribes one action for each information set. More used are mixed strategies, which presents a probabilistic distribution over actions for every information set.

Formally, pure strategies are defined as follows:

Definition 3. *Let $G = (N, A, H, Z, \chi, \rho, \sigma, u, I)$ be an imperfect-information extensive-form game. Then the pure strategies of player i consist of the Cartesian product $\Pi_{I_{i,j} \in I_i} \chi(I_{i,j})$.*

The strategy profile is a Cartesian product of mixed strategies for all players.

For evaluation of the strategy in the tree-like games, we use a concept of expected utility. Simply said, it is the sum of payoffs over all leafs, multiplied by the probability of reaching each leaf. The reaching probability comes from the strategies of players.

We use a sequence as the main definition, for the player's state, which are nodes in the game tree. The sequence is a path of actions taken by the player. It can lead to many states, and we need to know sequences of all players to determine the exact game state we are in.

The sequence is formally defined as follows:

Definition 4. *A sequence of actions of player $i \in N$, defined by a node $h \in H \cup Z$ of the game tree, is the ordered set of player i's actions that lie on the path from the root to h. Let $\emptyset$ denote the sequence corresponding to the root node. The set of sequences of player i is denoted Σ_i and $\Sigma = \Sigma_1 \times \ldots \Sigma_n$ is the set of all sequences.*

We use the sequence-form representation of games as a more refined representation than the extensive form. We formally define sequence form:

Definition 5. *Let G be an imperfect-information game. The sequence-form representation is a tuple $G = (N, \Sigma, g, C)$, where:*

- *N is a set of n players*
- *$\Sigma = (\Sigma_1, \ldots, \Sigma_n)$, where Σ_i is the set of sequences available to player i*
- *$g = (g_1, \ldots, g_n)$, where $g_i : \Sigma \to \mathbb{R}$ is the payoff function for player i*
- *$C = (C_1, \ldots, C_n$, where C_i is a set of linear constrants on the realization probabilities of player i*

Payoff function is the same as in the extensive form, but instead of being defined on leafs, it is defined for sequences leading into leafs as real-valued number and 0 for non-terminal sequences. Formally:

Definition 6. *The payoff function $g_i : \Sigma \to \mathbb{R}$ for player i is given by $g(\sigma) = u(z)$ if a leaf node $z \in Z$ would be reached when each player played his sequence $\sigma_i \in \sigma$, and by $g(\sigma) = 0$ otherwise.*

Now we have the sequence form and payoffs for sequences, and we can continue to the most important part: a realization plan. Realization plan is a form of strategy, for every sequence it assigns a probability for following sequences. Defined as:

Definition 7. *A realization plan for player $i \in N$ is a function $r_i : \Sigma_i \to [0, 1]$ satisfying the following constraints:*

$$r_i(\emptyset) = 1 \tag{2.1}$$

$$\sum_{\sigma_i' \in Ext_i(I)} r_i(\sigma_i') = r_i(seq_i(I)), \forall I \in I_i \tag{2.2}$$

$$r_i(\emptyset) = 1, \forall \sigma_i \in \Sigma_i \tag{2.3}$$

Where $seq_i : I_i \to \Sigma_i$ is a function returning sequence leading to the information set I_i.

And where $Ext_i : \Sigma_i \to 2^{\Sigma}$ is a function mapping from sequences to set of sequences. $Ext_i(\sigma_i)$ denotes the sequences, that extends the σ_i by playing one action. For leaf nodes Ext_i returns empty set. In the equation above, we used a shorthand $Ext_i(I) = Ext_i(seq_i(I))$.

2.2 Solution concepts

The solution concept is a subset of possible outcomes, which has some predefined property. In a single-player environment, we can restrict our computations simply to maximizing the player's utility. In a multi-player environment, we have to find a solution based not only on the environment but also on the other player's strategies.

In this section, we firstly describe the concept of best response, secondly possibly the most known solution concept of game theory: Nash equilibrium. Following with the definition of the concept of MaxMin. Lastly, we will define the Stackelberg equilibrium.

2.2.1 Best response

The best response is a very simple concept of utility maximization. It is a solution for one player, which knows other players strategies. Generally speaking, it is a simple utility maximization problem for one player. Firstly, we define strategy profile s_{-i} of all other players but player i: [19]

Definition 8. $s_{-i} = (s_1, \dots, s_{i-1}, s_{i+1}, \dots, s_n)$ *a strategy profile* s *without player* i*'s strategy. Thus we can write* $s = (s_i, s_{-i})$*.*

With a fixed strategy profile s_{-i} player can determine her best response strategy as:

Definition 9. *Player* i*'s best response to the strategy profile* s_{-i} *is a mixed strategy* $s_i^* \in S_i$ *such that* $\forall s_i \in S_i : u_i(s_i^*, s_{-i}) \geq u_i(s_i, s_{-i})$*.*

Recall our Example 1, if the first player plays action a with probability 1.0, then best response for the second player is playing action b with probability 1.0. She selects better option between a and b, which leads to utilities 0 and 5 respectively. Then, b is the best possible response for the second player in this simple game.

2.2.2 Nash equilibrium

Since the player will not know the strategies of the opponents, it is difficult to use the best response for finding a strategy. By abstracting the idea of best response for strategies of all players, we will get a Nash equilibrium, which is a solution concept, where all players play the best response.

Nash equilibrium is formally defined as follows:

Definition 10. *A strategy profile* $s = (s_1, \dots, s_n)$ *is a Nash Equilibrium if, for all players* i*,* s_i *is a best response to* s_{-i}*.*

When the strategy profile s is a Nash Equilibrium, deviation of a player from her strategy is either another best response or strategy, which is worse than the one in s.

If the players do not care about small changes in the utility, it can lead to the concept of an ϵ-Nash equilibrium. More formally:

Definition 11. *Fix $\epsilon \geq 0$. A strategy profile $s = (s_1, \dots, s_n)$ is an ϵ-Nash equilibrium if, for all agents i and for all strategies $s_i' \neq s_i, u_i(s_i, s_{-i}) \geq u_i(s_i', s_{-i}) - \epsilon$.*

There are Pure Nash equilibria, where strategy profile contains only pure strategies, and Mixed Nash Equilibriums, where the strategy profile contains mixed strategies.

Our previously used Example 1 has two Pure Nash Equilibria. Those are (a, b) and (b, a). If some player deviated from the strategy, she would get worse payoff than which she would get from following the equilibrium strategy. For example in strategy (a, b), if first player will play b instead of a, the payoff will drop from 5 to 2. Assuming, the first player plays a, if the second player rather plays a instead of the b, her payoff will drop from 2 to 0.

There is also a Mixed Nash Equilibrium in the Example 1. It has the same strategy for both players: playing (a, b) with probability $(0.6, 0.4)$. It will lead to an expected payoff 2 for both of them.

2.2.3 MaxMin

MaxMin is a concept suitable for games with more than two players. It models situation of player i, where other players $-i$ are trying to make the greatest harm to i. Therefore the player i is playing strategy, maximizing her worst-case payoff. Even when the strategy of $-i$ is not to make the greatest harm to player i, playing according to maxmin strategy ensures a minimal payoff, the i gets. Formally:

Definition 12. *The maxmin strategy for player i is $\arg\max_{s_i} \min_{s_{-i}} u_i(s_i, s_{-i})$, and the maxmin value for the player i is $\max_{s_i} \min_{s_{-i}} u_i(s_i, s_{-i})$.*

This solution concept leads to a conservative strategy, which prepares for the worst-case opponents but does not have to make any assumptions about their rationality or their interests.

2.2.4 Stackelberg equilibrium

Third solution concept we are using in this thesis is Stackelberg equilibrium proposed by Heinrich Freiherr von Stackelberg in 1934. Originally, the von Stackelberg researched the duopoly model, where two companies plan their production on the base of the other company's production and market. He showed, the one company already in the market can gain an advantage in committing to the strategy, and other company entering the market would make their production strategy according to the leaders' commitment. It can lead to maximizing leader's profit while playing best response as the follower. In this section, we use [2] as the main source about Stackelberg equilibrium.

This solution concept is not approaching all players the same. One of the players is the leader, which commits to her strategy and other players are followers, who play the best response. The leader chooses the strategy, so the best response of followers maximizes her expected utility.

Definition 13. *A Strong Stackelberg Equilibrium is a strategy $S = (\delta*, \rho*)$, such that:*

- *The leader commits to his best response $V_d(\delta*, \rho) \geq V_d(\delta, \rho), \forall \delta$*
- *The follower plays his best response $V_a(\delta, \rho^*) \geq V_a(\delta, \rho), \forall \rho$*
- *The follower breaks ties in leader's favor $V_d(\delta^*, \rho^*) \geq V_d(\delta^*, \rho)$*

Where d denotes defender or leader player and a follower player (or attacker). V_i function returns expected utility for player i and strategies δ and ρ represents leader's and follower's mixed strategies respectively.

The Stackelberg equilibrium provides us an interesting point of view on the strategies. As our research of a state of the art goes, no paper dealing with three-player poker has used a strategy based on Stackelberg equilibrium.

Chapter 3

Algorithms

In this chapter, we provide a description of the two main methods of computing strategies. Counterfactual Regret Minimization algorithm(abbreviated CFR in the rest of the thesis) and nonlinear programming (abbreviated as NLP) formulations for computing Stackelberg equilibrium.

3.1 Counterfactual Regret Minimization

As mentioned earlier in this thesis, CFR produces strategies convergent to an ϵ-Nash equilibrium in two-player zero-sum games. There is no theoretical proof of any convergence of CFR in games with more than two players. The experiments in [14] show the CFR produced a strategy that is an ϵ-Nash equilibrium, but the strategy for the Leduc poker is not even in a ϵ-Nash equilibrium. The current conjecture is, the CFR produces a coarse correlated equilibrium.

The CFR traverses through the game tree and updates the regret for playing actions according to current strategy profile. The strategy profile in the next step is made from actions with the lowest regret in each information set. Intuitively, regret is a difference of payoffs between action we play and action we do not play. The probability of playing an action is calculated according to values of regrets which are then normalized by the sum of all regrets for given information set.

Firstly, we make an overview of the theory behind CFR algorithm. Secondly, we present pseudocodes with a description.

3.1.1 Regret

The following formulations of the theory behind CFR along with the algorithm are based on [9] and [12].

As stated above, the regret is a penalization for between action we take and the action we do not take. Formally, we define regret as follows:

Definition 14. *Given a sequence of strategy profiles $\sigma^1, \dots, \sigma^T$, the external regret R_i^T for player i is*

$$R_i^T = \max_{\sigma_i' \in \Sigma_i} \Sigma_{t=1}^T (u_i(\sigma_i', \sigma_{-i}^t) - u_i(\sigma^t)) \tag{3.1}$$

The Definition 14 gives R_i^T meaning of amount of the utility player i could gain if she would follow one single strategy over all iterations T.

Counterfactual value is the expected utility of the information set for player i when players follow strategy profile σ. We multiply payoff u by reaching probabilities of all players in each leaf reachable from the information set I. The sum of the products we call the counterfactual value. We follow by the definition of counterfactual value for player i

Definition 15. *Let the Z_I be the set of terminal histories passing through I, $z[I]$ is the history leading to leaf $z \in Z$ contained in I. The σ is a strategy profile for all players. Then we define counterfactual value $v_i(I, \sigma)$ for player i and information set I:*

$$v_i(I, \sigma) = \Sigma_{z \in Z_I} u_i(z) \pi_{-i}^{\sigma}(z[I]) \pi^{\sigma}(z[I], z) \tag{3.2}$$

The external regret for player i is a difference of utilities and is a cumulative sum over all iterations. The value of external regret is calculated as the maximum from all sequences of player i from strategy profile Σ. On the other hand, the counterfactual regret is a difference of expected utilities for given time point t. The counterfactual regret is not a single value as the external regret is, it is calculated for each information set separately.

From these v values, we can define the counterfactual regret in each iteration for every action and information set. The counterfactual regret is the difference between playing according to σ^t or playing according to σ^t with the exception in an information set I, where we choose single action a, noted as $\sigma_{I \to a}$.

Definition 16. *Let the v_i be counterfactual values, $\sigma_{I \to a}$ be a strategy profile, where we fix the action a to be played in information set I. Then we define the counterfactual regret r for every information set I and action a:*

$$r(I,a) = v_i(\sigma^t_{I\to a}, I) - v_i(\sigma^t, I) \tag{3.3}$$

Last variables and information CFR uses are cumulative counterfactual regret R_i^T (3.4), current strategy profile σ_i^{T+1} (3.6) and cumulative profile s_i^T (3.7):

$$R_i^T(I,a) = \Sigma_{t=1}^T r_i^t(I,a) \tag{3.4}$$

$$R_i^{T,+} = \max\{R_i^T(I,a), 0\} \tag{3.5}$$

$$\sigma^{T+1}(I,a) = \begin{cases} \frac{R_i^{T,+}(I,a)}{\Sigma_{a'\in A(I)} R_i^{T,+}(I,a')}, & \text{if the denominator is positive} \\ \frac{1}{|A(I)|}, & \text{otherwise} \end{cases} \tag{3.6}$$

$$s_i^T(I,a) = \Sigma_{t=1}^T \pi^{\sigma^t}(I)\sigma_i^t(I,a) \tag{3.7}$$

The assignment into current strategy profile is called regret-matching [12]. It is a mechanism of computing new strategy profile in each iteration of the algorithm. Regret-matching normalizes the positive portions of accumulated regret for each action as shown in 3.6.

If one or more actions have positive cumulative regret value R, then the probability of playing the action a in given information I is normalized regret for playing that action (first row of 3.6). If there is no action with positive cumulative regret value, regret-matching assigns the probability of playing an action a in the information set I according to the uniform distribution (second row of 3.6).

3.1.2 CFR Algorithm

In the previous section, we have described key concepts and notions used in CFR algorithm. Now, we show pseudocodes with a description. We describe initialization of CFR in the Algorihm 1 and recursive function of CFR in Algorithm 2. The variables r, s and σ have the same meaning in the CFR algorithm pseudocode as they have in the previous section. The counterfactual regret, cumulative profile and strategy profile respectively.

The CFR algorithm is an iterative algorithm, which recursively goes through the game tree (in each iteration one pass through for every player) and updates the values of regrets and strategies. The update of values in σ is taking place in updateRegretTable(), which updates the values for all information sets in the game tree, according to (3.6).

Its initialization and calling the iterations is described in Algorithm 1 and the recursive function is described in Algorithm 2.

The following pseudocodes are what we used in experiments, and it is modified for usage in three-player poker:

Algorithm 1 CFR initialization

```
1: $\forall I, \forall a \in A(I) : r_I[a] = 0, s_I[a] = 0$
2: $\forall I, \forall a \in A(i) : \sigma(I, a) = \frac{1}{|A(I)|}$
3: **for** $t = \{1, 2, \ldots, T\}$ **do**
4:     **for** $i \in 1, 2, 3$ **do**
5:         updateRegretTable()
6:         CFR($\emptyset, i, 1, 1$)
7:     **end for**
8: **end for**
```

The CFR function takes 4 parameters h, i, π_i, π_{-i}, where h is a history, which denotes state currently examined by the function, i is the searching player, whom point of view it is at this moment, i changes only after the complete walkthrough of the game tree. Moreover π_i, π_{-i} are probabilities of reaching the state for the searching player and all other players combined, respectively.

CFR deals with leaves and chance nodes very straightforward. It returns the searching player utility (Algorithm 2 lines 3-5) in the leaf node. Moreover, it expands the recursion in chance nodes according to chance probability, which is stored in σ_c (Algorithm 2 line 7 - 13).

On the lines 22 - 25, the CFR goes through all possible actions and recursively resolves the next states. Results are weighted by current strategy profile σ. If the state is the decision node of the searching player, the update of r, s takes place on the lines 27 - 32. CFR returns the weighted sum of the CFR value of the descendants (last line).

Algorithm 2 CFR recursive function

```
function CFR($h, i, \pi_i, \pi_{-i}$)

if $h$ is terminal then
    return $u_i(h)$
end if

if $h$ is chance node then
    $sum \leftarrow 0$
    for $a \in A(h)$ do
        $sum \leftarrow sum + \sigma_c(h,a) \cdot CFR(ha, i, \pi_i, \sigma_c(h,a) \cdot \pi_{-i})$
    end for
    return $sum$
end if

Let $I$ be the information set containing $h$
$v_\sigma \leftarrow 0$

for $a \in A(I)$ do
    $v_{\sigma_{I \to a}}[a] \leftarrow 0$
end for

for $a \in A(I)$ do
    $v_{\sigma_{I \to a}}[a] \leftarrow CFR(ha, i, \sigma(I,a) \cdot \pi_i, \pi_{-i})$
    $v_\sigma \leftarrow v_\sigma + \sigma(I,a) \cdot v_{\sigma_{I \to a}}[a]$
end for

if $P(h) = i$ then
    for $a \in A(I)$ do
        $r_I[a] \leftarrow r_I[a] + \pi_{-i} \cdot (v_{\sigma_{I \to a}}[a] - v_\sigma)$
        $s_I[a] \leftarrow s_I[a] + \pi_i \cdot \sigma(I,a)$
    end for
end if

return $v_\sigma$
```

3.2 Stackelberg equilibrium

As mentioned earlier, we compute Stackelberg equilibrium with the non-linear programming (abbreviated as NLP in the rest of the thesis). Mathematical programming is a description of an optimization problem with a function that is maximized and set of constraints, which the solution must be consistent with.

Stackelberg equilibrium defined in the previous chapter is formulated for the two-player game. For our three-player domain, we use the following with one leader and multiple followers [2]:

Definition 17. *A multi-follower game is a structure $G = (N, A, U)$ such that*

- $N = (d, \Psi)$ *is the set of players where d represents leader and* Ψ *is the set of followers*
- $A = A_d \times A_\Psi$ *is the set of players' actions where* A_d *is the set of leader's actions and* $A_\Psi = A_1 \times \ldots \times A_{|\Psi|}$ *is the set of followers' actions*
- $U = (U_d, U_\Psi)$, *where* $U_d : A \to \mathbb{R}$ *and* $U_\Psi : A \to \mathbb{R}^{|\Psi|}$ *represent the utilities of all players in each outcome of the game*

There are few possible combinations of pure and mixed strategies for leader and follower. Our focus is on the finding a mixed strategy for leader and followers. The combination where all players play the mixed strategy leads to the most realistic and complex model of the strategies for the game. For this purpose, we are using following nonlinear program and mixed-integer nonlinear program: [2]

Formulation 1.

$$\max_{\delta,\rho_1,\rho_2} \sum_{i\in D}\sum_{j\in A_a}\sum_{k\in A_b} \delta_i \rho^a_j \rho^b_k U_d(i,j,k) \tag{3.8}$$

$$s.t. \sum_{i\in D}\sum_{k\in A_b} v_a - \delta_i \rho^b_k U_a(i,j,k) \geq 0 \qquad \forall j \in A_a, \forall a \in \Psi \tag{3.9}$$

$$\sum_{i\in D}\sum_{k\in A_b} \rho^a_j v_a - \delta_i \rho^a_j \rho^b_k U_a(i,j,k) = 0 \qquad \forall j \in A_a, \forall a \in \Psi \tag{3.10}$$

$$\sum_{j\in A_a} \rho^a_j = 1 \qquad \forall a \in \Psi \tag{3.11}$$

$$v_a \in \mathbb{R} \qquad \forall a \in \Psi \tag{3.12}$$

$$\sum_{i\in D} \delta_i = 1 \tag{3.13}$$

Maximization function is a simple expected utility function for the leader. The variable v_a describes the value of the best response. Therefore, the constraint (3.9) forces strategies of one follower and leader to be not lower than the best response of the other follower. It sets the value of the best response for the other follower. The (3.10) provide a guarantee, that follower either plays action leading to best response or not play the action at all. The Problem 1 has the non-convex constraint (3.10), which makes the computation of global optimum a hard problem.

Formulation 2.

$$\max_{\delta,\rho_1,\rho_2} \sum_{i\in D}\sum_{j\in A_a}\sum_{k\in A_b} \delta_i \rho_j^a \rho_k^b U_d(i,j,k) \qquad \forall a,b \in \Psi, b \neq a \tag{3.14}$$

$$s.t. \sum_{i\in D} \delta_i = 1 \tag{3.15}$$

$$\sum_{j\in A_a} \rho_j^a = 1 \qquad \forall a \in \Psi \tag{3.16}$$

$$u_j^a = \sum_{i\in D}\sum_{k\in A_b} \delta_i \rho_k^b U_a(i,j,k) \geq 0 \qquad \forall j \in A_a, \forall a,b \in \Psi, a \neq b \tag{3.17}$$

$$v_a \geq u_j^a \qquad \forall j \in A_a, \forall a \in \Psi \tag{3.18}$$

$$r_j^a = v_a - u_j^a \qquad \forall j \in A_a, \forall a \in \Psi \tag{3.19}$$

$$\rho_j^a \leq 1 - s_j^a \qquad \forall j \in A_a, \forall a \in \Psi \tag{3.20}$$

$$r_j^a \leq M s_j^a \qquad \forall j \in A_a, \forall a \in \Psi \tag{3.21}$$

$$s_j^a \in \{0,1\} \qquad \forall j \in A_a, \forall a \in \Psi \tag{3.22}$$

$$\rho_j^a \geq 0 \qquad \forall j \in A_a, \forall a \in \Psi \tag{3.23}$$

$$\delta_i \geq 0 \qquad \forall i \in D \tag{3.24}$$

The Problem 2 describes the mixed-integer NLP without the nonconvex constraint (3.10) and leads to a program, where a solver can find an optimal solution in finite time.

This formulation presents three support variables s, r, u. u_j^a describes utility for follower a taking action j on (3.18). These utilities than set v_a as high as they can (3.18), which leads to finding a best-response. r_j^a represents the difference between utility for taking action and best response value (3.19).

Because the v_a is directly set by the highest r_a, the difference can be either 0 or some positive real number. If the r_j^a is higher than 0, it sets the binary variable s_j^a to 1 on (3.20). This setting signals, action j has worse expected utility than is the best response; therefore we do not play action j. Which is managed by the constraint on (3.21).

Chapter 4

Algorithms modifications

We have the solution concepts, which produce reasonable strategies, defined in Chapter 2, but the computation of the Nash Equilibrium for the n-player game is an FIXP-hard problem. Besides the Nash equilibrium does not provide any guarantee against two players deviating from their strategies in the n-player zero-sum game.

Therefore we are not explicitly computing the Nash equilibrium, but rather use a novel approach for the domain of poker. We use solution concepts such as Stackelberg equilibrium or modification of MaxMin strategy, which we compare against CFR strategy. We use the CFR strategy because it is a common approach in the multiplayer poker community, for example [14].

In this chapter, we describe our modifications of CFR and NLP for Stackelberg equilibrium, to compute reasonable strategies, for three-player poker game. As presented earlier, CFR converges to Nash Equilibrium only in two-player zero-sum games, without any guarantee for games with more two players. We use the CFR algorithm to compute regret minimizing strategy and modified CFR algorithm to compute the heuristical MaxMin strategy. Lastly, we modify the NLP to reflect the structure of poker game.

4.1 Counterfactual Regret Minimization

The articles about the usage of CFR in multiplayer poker games were missing a pseudocode for CFR with more than two players. In the previous chapter, we have described multiplayer CFR algorithm, where we abstracted the probabilities π_i, π_{-i}. They originally represented player 1 and player 2, in our formulation, they represent the probability of searching player and the probability of all opponents of searching player combined. The formulation of CFR in Algorithm 2 is applicable to n-player games as is.

4.2 MaxMin computation

We want to compute a robust strategy according to the MaxMin concept. We assume the MaxMin strategy will lead to a defensive strategy, that plays with same quality regardless of its opponents. The straightforward computation of MaxMin leads to the situation, where both opponents can be seen as one player. It would mean, the opponents share all their knowledge including the hand cards. Which is an unrealistic situation and very pessimistic assumption for the searching player.

The less pessimistic approach is to see opponents as one player but merges only their payoff. This situation is an imperfect-recall game, where the opponent player forgets the card of the other teammate during her turn.

This leads to the reasonable model of the conservative player defending herself against all other opponents, who align their interests against one player. This aligning of interests is represented by merging their payoffs, but the opponents both act on their own. Which is a more realistic situation, which aims to compute careful strategy focused on minimizing exploitability by other players. The MM strategy does not try to model opponents in any way; it assumes they try to do the most damage and therefore generates defensive approach indifferent on the opponents.

We use CFR algorithm to heuristically approximate the MaxMin strategy. The two players $-i_1$ and $-i_2$ are playing against the third one i. When the game is grouped into two antagonistic sets of players, it is clear, the gain of one set is the direct loss of the other. To compute this strategy (abbreviated in the rest of the thesis as MM), we have changed a way of calculating payoffs. For the player i, the payoff is calculated the same way. The new payoffs u'_{-i_1}, u'_{-i_2} for team-up players $-i_1$ and $-i_2$ are the sum of the original ones:

$$u_{-i} = u'_{-i_1} = u'_{-i_2} = u_{-i_1} + u_{-i_2} \tag{4.1}$$

4.3 Stackelberg equilibrium

The Formulation 1 and Formulation 2 are for the games, where players choose from a single set of actions. We have modified the formulations in Formulation 3 and Formulation 4 specifically for the poker, which is the extensive-form game.

In the game of poker, the set of possible actions depends on information set, where the player is. The possible action sets depend on player's position[1] and actions of previous players.

[1]For explanation of positions see Section 5.1.3

Table 4.1: Actions of players in betting round divided by positions and actions of other players

Denoted	Position 1	Position 2	Position 3
PA_1	(check), (bet)	(check), (bet)	(check), (bet)
PA_2	$\emptyset$	(call), (fold)	(call), (fold)
PA_3	(check, call), (check, fold)	(check, call), (check, fold)	$\emptyset$

The possible sets for each position are shown in the following Table 4.1. The PA_1 represents the state, where other players played check. The PA_2 is a set of actions, reacting to a bet made by another player. The PA_3 is a set, where the player played check, and some of the following players played bet, so the player has to react to the bet.

The previous actions of betting round and hand card determine the information set. We denote $\mathbb{I}_a^{PA_1}$ the set of all information sets, where player a can play actions from the PA_1 set.

To compute Stackelberg equilibrium, we have to work properly with information sets, which occurs during the game. In the objective function and all following lines, we use only combinations of actions, which lead to leaf.

The A_a in original formulation denotes set of all actions for player a. We have extended this notation to A_a^I, which represents set of all possible actions given information I. For example, $A_a^{check,check}$ is a set of actions after other players both checked.

Secondly, the resulting strategies from the NLP are valid realization plans. The probabilities of taking actions from $\mathbb{I}_a^{PA_1}$ sum to 1 and the probabilities of taking actions from $\mathbb{I}_a^{PA_2}$ sum to 1 as well ((4.5), (4.7), (4.11) and (4.13). The probabilities of taking actions from $\mathbb{I}_a^{PA_3}$ sum to a probability of playing (check) by the given player a in the previous information set ((4.6), (4.8), (4.12) and (4.14)).

Thirdly, the v_a representing the best response value for player a is also extended, so it represents best response value with respect to the information set. The v_a is a value of best response for player a and we are using v_a^I with similar meaning as we used in A_a^I. The v_a^I refers to the best-response value of player a given information I (4.16). There are v for each action set, we distinct between them by information about player's a action.

In the Formulation 3 and Formulation 4 we denote the $\mathbb{I}$ as the set of all information sets, $\mathbb{I}_a$ as the set of all information sets for given player a.

Formulation 3.

$$\max_{\delta,\rho_1,\rho_2} \sum_{i\in D} \sum_{j\in A_a^{\delta_i}} \sum_{k\in A_b^{\delta_i \rho_j^a}} \delta_i \rho_j^a \rho_k^b U_d(i,j,k) \tag{4.2}$$

$$s.t. \sum_{i\in D^{\rho_j^a}} \sum_{k\in A_b^{\delta_i \rho_j^a}} v_a^j - \delta_i \rho_k^b U_a(i,j,k) \geq 0 \qquad \forall j \in A_a, \forall a \in \Psi \tag{4.3}$$

$$\sum_{i\in D^{\rho_j^a}} \sum_{k\in A_b^{\delta_i \rho_j^a}} \rho_j^a v_a^j - \delta_i \rho_j^a \rho_k^b U_a(i,j,k) = 0 \qquad \forall j \in A_a, \forall a \in \Psi \tag{4.4}$$

$$\sum_{j\in A_a^I} \rho_j^a = 1 \qquad \forall a \in \Psi, \forall I \in (\mathbb{I}_a \setminus \mathbb{I}_a^{PA_3}) \tag{4.5}$$

$$\sum_{j\in A_a^I} \rho_j^a = \rho_{check}^a \qquad \forall a \in \Psi, \forall I \in \mathbb{I}_a^{PA_3} \tag{4.6}$$

$$\sum_{i\in D^I} \delta_i = 1 \qquad \forall I \in (\mathbb{I}_d \setminus \mathbb{I}_d^{PA_3}) \tag{4.7}$$

$$\sum_{j\in D^I} \delta_i = \delta_{check} \qquad \forall I \in \mathbb{I}_d^{PA_3} \tag{4.8}$$

$$v_a^I \in \mathbb{R} \qquad \forall a \in \Psi, \forall I \in \mathbb{I}_a \tag{4.9}$$

Formulation 4.

$$\max_{\delta,\rho_1,\rho_2} \sum_{i\in D} \sum_{j\in A_a^{\delta_i}} \sum_{k\in A_b^{\delta_i \rho_j^a}} \delta_i \rho_j^a \rho_k^b U_d(i,j,k) \qquad \forall a,b \in \Psi, b \neq a \tag{4.10}$$

$$s.t. \sum_{j\in A_a^I} \rho_j^a = 1 \qquad \forall a \in \Psi, \forall I \in (\mathbb{I}_a \setminus \mathbb{I}_a^{PA_3}) \tag{4.11}$$

$$\sum_{j\in A_a^I} \rho_j^a = \rho_{check}^a \qquad \forall a \in \Psi, \forall I \in \mathbb{I}_a^{PA_3} \tag{4.12}$$

$$\sum_{i\in D^I} \delta_i = 1 \qquad \forall I \in (\mathbb{I}_a \setminus \mathbb{I}_d^{PA_3}) \tag{4.13}$$

$$\sum_{j\in D^I} \delta_i = \delta_{check} \qquad \forall I \in \mathbb{I}_d^{PA_3} \tag{4.14}$$

$$u_j^a = \sum_{i\in D^{\rho_j^a}} \sum_{k\in A_b^{\delta_i \rho_j^a}} \delta_i \rho_k^b U_a(i,j,k) \geq 0 \qquad \forall j \in A_a, \forall a,b \in \Psi, a \neq b \tag{4.15}$$

$$v_a^I \geq u_j^a \qquad \forall j \in A_a, \forall a \in \Psi, \forall I \in \mathbb{I}_a \tag{4.16}$$

$$r_j^a = v_a^I - u_j^a \qquad \forall j \in A_a, \forall a \in \Psi, \forall I \in \mathbb{I}_a \tag{4.17}$$

$$\rho_j^a \leq 1 - s_j^a \qquad \forall j \in A_a, \forall a \in \Psi \tag{4.18}$$

$$r_j^a \leq M s_j^a \qquad \forall j \in A_a, \forall a \in \Psi \tag{4.19}$$

$$s_j^a \in \{0,1\} \qquad \forall j \in A_a, \forall a \in \Psi \tag{4.20}$$

$$\rho_j^a \geq 0 \qquad \forall j \in A_a, \forall a \in \Psi \tag{4.21}$$

$$\delta_i \geq 0 \qquad \forall i \in D \tag{4.22}$$

Chapter 5

Computing poker strategies

In this chapter, we will describe poker and poker variants we are using and difficulties we have to overcome during computing strategies with CFR algorithm and nonlinear programming for Stackelberg equilibrium.

5.1 Poker variants

In the game of poker, players operate with cards and chips. Cards determine how good is player standing; the player can have highest card, pair or straight made of several cards. There are hand cards (or private cards), which are unique for each player, and table cards, which are same for all players. Chips can be viewed as "game money." Players use them to bet on their hands; they can be used, to "pretend" (or bluff) better hand cards, then the player has. Bets are made to the *pot*, which is the sum of all bets of all players. The player can bet chips, reaction to this is a call (adding the same amount of chips into the pot) or fold (giving up this hand with no possibility of winning). When the player wants to continue in a game, her chip sum in the pot must equal to other playing players.

Actions can be divided into leading (check or bet) and following (call or fold) actions. Until some player plays bet, we can choose only from leading actions. When some player before us played bet, we can play only call or fold.

We are computing our strategies for two known poker variants: Kuhn poker and Leduc Hold'em. Both these variants were created as two-player games (in [11] and [6] respectively), and for our purposes, we are using three-player variants as described in [14]. Both variants have a very small deck, and the main difference is that Kuhn poker has only one betting round and no table card and Leduc Hold'em has two betting rounds and one table card.

5.1.1 Kuhn poker

Kuhn poker is a one-round poker game, where the deck has four cards (A, K, Q, J). Players can win only with the highest card, the order of cards is A > K > Q > J, with no possible tie. Firstly, each player places one chip (called Ante) into the pot, then players get one hand card randomly from the deck. After card dealing, players take actions in betting round, where they take turns in the order determined before the hand. Lastly, when the end of betting round is reached, the winner is declared, and she gets whole pot.

Kuhn poker has a 1-bet cap, which means, there can be only one bet action in betting round. The bet value is one chip.

5.1.2 Leduc Hold'em

Leduc poker is a two-round poker game, where the deck has eight cards, four ranks and two suits (As, Ad, Ks, Kd, Qs, Qd, Js, Jd). Order of cards is the same as in the Kuhn poker, and tie is resolved by splitting the pot. Ante, dealing hand cards and first betting round is the same as in the Kuhn poker, with one exception, bet in the first round has value of 2 chips. When there are at least two players, who have not played fold in the first round, the table card is dealt out of the remaining deck, and another betting round is played with active players (not folded). In the second round, the bet has the value of four chips.

In Leduc Hold'em, the player can match their card with the table card, and the same rank makes a pair, which beats all other hands. When no pair is showed, highest card wins.

5.1.3 Positions

The players take turns in each betting round until the betting round is resolved. The order of taking turns is determined by the position of the player. We can imagine the position as the place around the table, where the players sit. Player in position 1 plays first after her action is the turn of the player in position 2 and so on. The order does not change during the game, but it changes after each game of the tournament.

5.2 CFR computations

The CFR algorithm is symmetrical and it computes the strategy for all positions in one run. CFR is trying to model the opponents for the searching player by the same way; it is computing strategy for the searching player. The MM and Stackelberg equilibrium strategies handle opponents for the searching player asymmetrically. The MM is expecting them to be the worst possible enemies and the SE is modeling them to be the player playing

Table 5.1: Expected utilities for Kuhn poker

	Position 1	Position 2	Position 3
CFR	-0.027914	-0.021318	0.049232
BR	-0.02775	-0.019357	0.049311
|CFR - BR|	1.63565E-4	0.001961	7.91307E-5

Table 5.2: Expected utilities for Leduc Hold'em

	Position 1	Position 2	Position 3
CFR	0.002381	-0.04881	0.046429
BR	1.063844	1.02619	0.332143
|CFR - BR|	1.061463	1.075	0.285714

the best response against the strategy of the searching player. The MM and SE compute strategy only for one position in one run, and therefore it had to be computed for each position separately and merged afterward.

For evaluation, whether the strategy is the ϵ-Nash or not, we compute best response for each position with fixed strategy for the other two, as done in [14] $\sigma^{BR} = (\sigma_1^{BR}, \sigma_2^{BR}, \sigma_3^{BR})$. The comparison of expected utilities and best responses for Kuhn poker and Leduc Hold'em are shown in Table 5.1 and Table 5.2 respectively. The values of the CFR for Kuhn poker resulted in ϵ-Nash equilibrium with $\epsilon = 0.001961$ (the maximum from the last line in Table 5.1) and not resulted in ϵ-Nash equilibrium for Leduc Hold'em. These results are in agreement with results from [14].

To compute MM strategies, we have altered game tree as described in the previous chapter. After running CFR on this altered game for each position, we have merged strategies on the positions, where the player plays against other players. This merged strategies we abbreviate MM. The expected utilities are in Table 5.3 in comparison to SE3 (described below) and CFR strategy, where the utilities suggest the MM strategy being the most pessimistical strategy in expected utility point of view.

5.3 Stackelberg equilibrium computations

To compute the NLP for Stackelberg equilibrium, we use the formulation from Formulation 4 as a baseline, because it has a better possibility to find a global optimum in real time. And formulation from Formulation 3 as more easily computable, but without guarantee for global optimum, with the non-convex constraint.

The NLP formulation is different for each position, and the consequence is, searching for the optimum is not equally difficult. We have used a NEOS Server for finding optima

Table 5.3: Expected utilities for SE3 strategy and MM strategy

	Position 1	Position 2	Position 3
CFR	-0.027914	-0.021318	0.049232
SE3	0.0359	0.04167	0.1562
MM	-0.03645	-0.020845	0.041679

for our NLPs[1].

We have tried to compute the optimal solution for every position with Formulation 4, where only Position 2 resulted in global optimum. The solver also found a local optimum for Position 3. To compute the global optimum for Position 1 and 3, we tried to pre-set the strategies to a strategy from CFR. The pre-configuration resulted in better expected utility for Position 3. Therefore we are using this strategy. We were unable to calculate strategy for Position 1 even with the manual setting of variables to CFR strategy.

Each strategy can be divided in the distinct strategy for each position $s = (s_1, s_2, s_3)$, where s is complete strategy and s_i is a strategy for position i. Because the Formulation 4 did not returned any results for Position 1, we have filled the s_1 in two ways: by s_1 from CFR strategy (denoted as SE2) and by result of 3 (denoted as SE3). This complementation of s_1 strategy implies, the SE2 and SE3 strategies differ only on Position 1.

Comparison of expected utilities for CFR, SE3, and MM for all three positions are described in 5.3.

The limitations of finding a solution for NLP for Kuhn poker has lead to inability of computing Stackelberg equilibria for Leduc Hold'em. Kuhn Poker has around 600 nodes in 48 information sets, while Leduc Hold'em has almost 33000 nodes in 1200 information sets, which is a much larger game and leads to uncomputable formulations by our NLPs in real-time. Kuhn Poker has almost around 240 variables, a similar number of equations 250. In the objective function, there are more than 300 multiplications of three variables. In the Leduc Hold'em, the objective function would consist of almost 17000 of multiplications. The counts are made estimated for Formulation 4.

5.4 Computations

Computation of strategies is not an easy nor quick task. To compute the CFR and MM strategies, we have compiled executables and used MetaCentrum for evaluation. We have run the CFR algorithm for computing CFR and MM strategies for 10^8 iterations each. For example, Leduc Hold'em CFR strategy was computed in 9 days, and we have used 336

[1]See the following section for more information about the usage of NEOS Server

days of CPU time to do all the computations.

We have used NEOS server ([3], [4], [10]) to compute the Stackelberg equilibrium strategies for all positions. We have created the formulation for all three positions in GAMS format and submitted them into NEOS. Tne NEOS server provides 8 hours of CPU time and 3 GB of memory. We have tried several solvers (KNitro, Baron, scip, SNOPT), out of which the KNitro solver was the most successful. The KNitro [18] found an optimal solution for Position 2; it also found a locally optimal solution for Position 3 in a given time. However, could not find a solution for Position 1.

Chapter 6

Comparison of strategies

There is no concept of the value of the game (as is for two-player games), and no other similar concept for multiplayer games exists. Thus, we analytically can not tell, which strategy is better than the other. We use a tournament-bases evaluation to compare the strategies for Kuhn poker and Leduc Hold'em. The tournament-based evaluation is a commonly used method for multiplayer poker domain [1] and [14].

We follow the common practice and aggregate the results in the total average payoff for each strategy. However, as we show later, there is a substantial difference in the performance of different algorithms playing at different positions. There we compare the results of the tournaments in a less aggregated way. We also look which strategies are defensive and which tends to bluff and take more risky approach. Lastly, we examine how different ordering around poker table changes the outcome for the same set of players.

For experiments besides our CFR, MM, SE2 and SE3 strategies (described before), we are also using two simple heuristic strategies: Random and Aggressive (abbreviated RND and Aggro respectively). The Random strategy sets uniform probability distribution for all actions in every information set. Aggressive strategy always chooses the bet and call actions.

6.1 Tournament comparison

6.1.1 Tournament setting

For running the tournaments, we used the ACPC[1] server [1], which is the general easily configurable server for different poker variations. ACPC server rotates the order of given players. If we run the tournament between players A, B and C, they will rotate on

[1]Annual Computer Poker Competition, which usually takes place at the AAAI conference

position. First dealing (A, B, C), second dealing (B, C, A), third (C, A, B) and fourth is the same as first. Therefore, we have to run both (A, B, C) (*table set-up I*) and (A, C, B) (*table set-up II*) to try all six possible orderings around the table.

For Kuhn poker, we have run 10^7 hands for each of the set-ups *I* and *II*. For Leduc Hold'em, we have run $5 \cdot 10^6$ hands for each of the set-ups *I* and *II* to finish the tournaments in the reasonable time.

6.1.2 Tournament results

We run the tournaments in both set-ups for the most significant and interesting triplets. We sum payoffs over all hand in the set-up, calculate the standard deviation, and the resulting score of strategy in the triplet is the 95% confidence interval. The values in the tables are averages of milliantes per hand (ma/h), within interval stated in table description. The last line is overall average for each strategy.

The values in cross-tables are for strategies noted in columns. The other two players of the triplet are described in the row. All examined triplets[2] for Kuhn poker are shown in Table 6.1, with strategies order from the best to the worst: MM, SE3, CFR, SE2, Aggro and RND. Few interesting points, the MM strategy not only is the best strategy in the total payroll point of view, but it is also the only strategy ending up with the positive score after each game. The aggressive strategy seems good on the first look, but when examining results in more detail, this strategy gets the positive score only when one of the other players is random players, the most exploitable strategy we have used. Against only rational strategies, the aggressive strategy has the average score of -136.5.

The Table 6.2 shows the results for the triplets made out of rational players only. The order of strategies changed only for CFR strategy, which dropped to the last place, ending up with the only negative overall payoff. The CFR strategy plays reasonably well against opponents, where at least one is the heuristic player, but against other rational strategies, it does not do well.

The results for Leduc Hold'em are in Table 6.3, which shows similar results as seen in Kuhn poker. Order of strategies is MM, CFR, Aggro and RND. Where aggressive strategy again loses against (MM, CFR) opponents.

[2]Note, we are not comparing SE2 and SE3 directly since they differ with strategies only for Position 1. We compare them on their score against other strategies

Table 6.1: Cross-table for 3-player Kuhn poker tournament. Values are in ma/h within ± 1.2 ma/h with 95% confidence

	mm	**cfr**	**se3**	**se2**	**aggro**	**rnd**
cfr, mm	†	†	**43.40**	27.10	-211.20	-323.90
cfr, aggro	**177.10**	†	125.50	46.30	†	-476.60
cfr, rnd	**166.90**	†	166.70	99.30	82.60	†
mm, aggro	†	**34.10**	-10.80	-47.30	†	-461.20
mm, rnd	†	157.00	89.50	66.00	**211.80**	†
se2, cfr	**79.50**	†	†	†	-125.60	-309.80
se2, mm	†	-106.60	†	†	**-54.00**	-187.00
se3, cfr	**91.50**	†	†	†	-173.70	-336.70
se3, mm	†	-134.90	†	†	**-117.80**	-220.10
se2, aggro	**101.30**	79.30	†	†	†	-412
se2, rnd	121.00	210.50	†	†	**335.80**	†
se3, aggro	**128.70**	48.20	†	†	†	-438.40
se3, rnd	130.60	170.00	†	†	**261.60**	†
rnd, aggro	249.40	**394.10**	76.20	176.80	†	†
Overall	**138.44**	**94.63**	**98.52**	**44.6**	**23.28**	**-351.74**

Table 6.2: Cross-table for 3-player Kuhn poker tournament for rational players. Values are in ma/h within ± 1.2 ma/h with 95% confidence

	mm	**cfr**	**se3**	**se2**
cfr, mm	†	†	43.40	27.10
se2, cfr	79.50	†	†	†
se2, mm	†	-106.60	†	†
se3, cfr	91.50	†	†	†
se3, mm	†	-134.90	†	†

Table 6.3: Cross-table for 3-player Leduc Hold'em tournament. Values are in ma/h within ± 4.5 ma/h with 95% confidence

	mm	**cfr**	**aggro**	**rnd**
mm, aggro	†	**293.2**	†	-1116.1
cfr, aggro	**390.3**	†	†	-1028.3
mm, rnd	†	358.8	**565**	†
cfr, rnd	365.1	†	**731.1**	†
mm, cfr	†	†	**-683.5**	-723.9
rnd, aggro	**551.2**	297.2	†	†
Overall	**436**	**316**	**204**	**-956**

6.2 Position comparison

In Texas Hold'em, the latter is the players turn, the better for her. Because she can gather information from actions of the previous players. The last position (called dealer position) is the best, from the general point of view.

Therefore we have analyzed the behavior of strategies based on position[3] through all hands played for Kuhn and Leduc experiments.

The Table 6.4 propose, the position is important, but each strategy handle positioning differently. Best scoring strategy MM has the highest average utility from the dealer position. However, it is the only one. It seems that CFR strategy takes advantage of the position in opposite from our belief. In fact, SE2, SE3, and MM take advantage of CFR and simple strategies on the dealer position.

The SE2 and SE3 have the average payoffs for Position 2 and 3 in the interval of confidence, which is expected. However, on Position 1 they score very differently, and it seems, the NLP program for computing Stackelberg equilibrium come up with a very strong strategy for starting position. Which we attribute to the nature of Stackelberg equilibrium, where it commits to a strategy and the opponents follow, which is the situation on Position 1.

When we look at the Table 6.5, we see almost no change for aggressive and random strategy. Interesting points, CFR is no longer losing on Position 3 but is still strongest on Position 1. MM strategy looks balanced in Kuhn Poker, but in Leduc Hold'em, it relies heavily on Position 2 with a good game on Position 1.

Our experiments showed, our strategies do not exploit dealer position at all. For simple strategies, the dealer position is the most exploited. All strategies except MM got the highest average values on Position 1. MM got highest on Dealer Position in Kuhn poker and on Position 2 for Leduc Hold'em. Maybe this is the difference, which made MM strategy successful in the tournament comparison.

Table 6.4: Average payoffs in ma/h based on positions for Kuhn poker

	Position 1	**Position 2**	**Position 3**	**Overall**
MM	126.5	111.4	**177.4**	138.4
CFR	199.6	**152.7**	-68.5	94.6
SE3	**297.8**	-109.7	107.5	98.5
SE2	135.5	-109.4	107.7	44.6
Aggressive	124.7	40.2	-95.1	23.3
Random	-282.5	-339.3	-433.5	-351.7

[3]Note, SE2 and SE3 have the same strategies for positions 1 and 2.

Table 6.5: Average payoffs in ma/h based on positions for Leduc Hold'em

	Position 1	**Position 2**	**Position 3**	**Overall**
MM	**452**	**639**	215	436
CFR	380	259	**310**	316
Aggressive	436	305	-129	204
Random	-815	-967	-1085	-956

Another view on positions we show in Figure 6.1, where the x-axis divide payoffs for different positions and the y-axis the payoffs. The bars in the graphs show the relative number of payoffs of the same value. The red points show average payoff for given position, and the green point is the overall average utility.

For example, first sub-graph shows the CFR strategy. The columns represent the three positions around the poker table. For CFR in Position 1 the graph shows, the most hands resulted in payoff -1A (the A stands for Ante).

Graphs suggest the CFR plays less safely than the rest of the strategies, by having more loss of 2A chips, where other strategies have fewer of these situations. The CFR compensate the loses by more wins.

This distinction can also be seen on Position 1 for SE2 and SE3, where SE3 has more -2A loses and also more 4A wins.

The MM, SE2, and SE3 are trying to minimize the -2A loses, resulting in lower number of wins.

The graphs in Figure 6.2 show the CFR and MM strategy for the Leduc Hold'em. The Leduc Hold'em seems to bring complexity and randomness, which makes the MM and CFR approach more similar to each other. The biggest difference is in position handling, where the CFR loses less on Position 1 and more on Position 3, while the MM loses less on Position 2 and the most loses takes on Position 1, which are compensated by higher wins.

6.3 Bluffing and value betting

In this section we look at the strategies from the point of their value betting, bluffing and how safely/risky they play the game. Bluffing in our poker variants can be defined as winning the hand with worse cards than one or both opponents. Safety of the strategy can be viewed as how many instant folds strategy does (losing only the Ante), how much the strategy bluffs, and so on. Value betting is a term from the game of poker, where the player gets as many chips as they can get, from the dealt hand. It is easy to win with Ace in Kuhn Poker, but it is not easy to win with Ace and make both opponents bet the maximum.

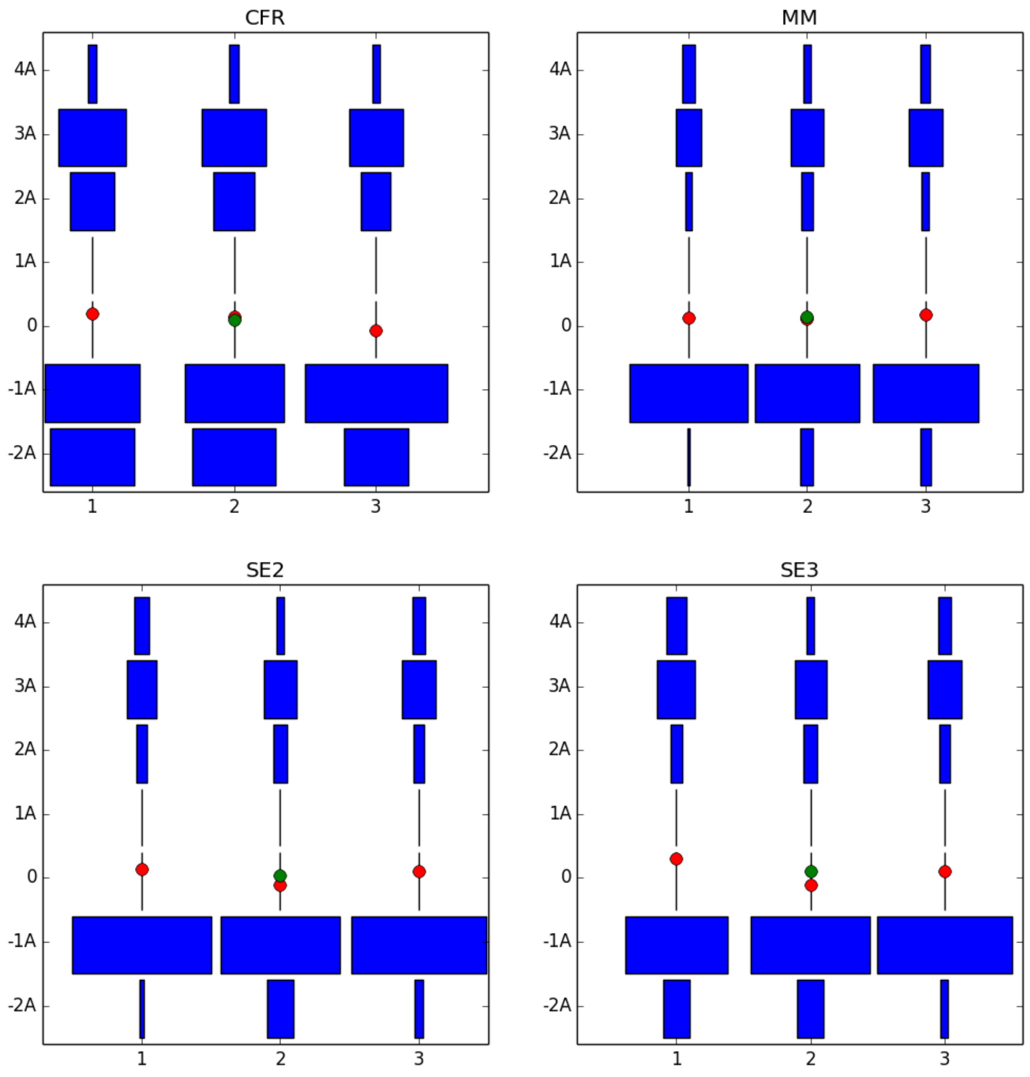

Figure 6.1: Payoffs based on positions in Kuhn poker. Bars show number of payoffs of a value noted on y-axis divided by positions

We analyze the overall percentage of winning hands (Wins), a number of wins with the best hand (BH Wins), which also points to the number of wins by bluffing. To examine the ability to win the most, we compare the number of wins with the best hand, where both opponents folded (BH Wins FF). Last value we check, measure number of folds, when

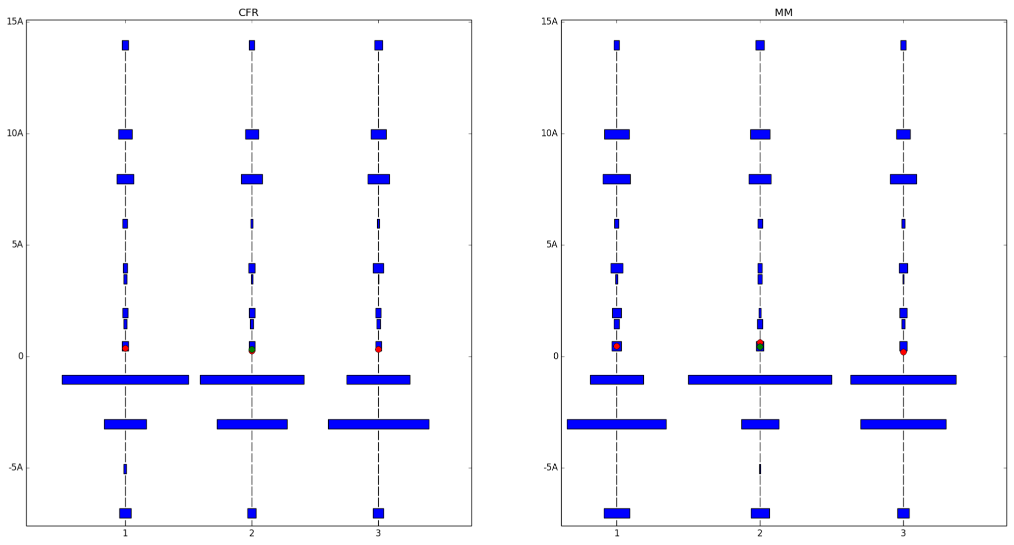

Figure 6.2: Payoffs based on positions in Leduc Hold'em. Bars show number of payoffs of a value noted on y-axis divided by positions

someone placed a bet, with respect to the total number of loses.

Our assumption for good strategy is, that this strategy plays safe and waits for good cards. Trying to minimize loss with bad cards and maximize winnings with good cards (by value betting). Therefore solid strategy plays fewer hands by playing many folds and win hands without other players folding.

Table 6.6: Behavior analysis of strategies in Kuhn Poker. Meaning of columns in order: percentage of wins from all games; percentage of wins with the best hand from all wins; percentage of wins with best hand, when both opponents folded; percentage of loss with the best hand out of all loses; percentage of losing only ante from all loses

	Wins	**BH Wins**	**BH Wins FF**	**BH loss**	**F/L**
CFR	36%	86%	25%	3.4%	50%
MM	29%	98%	5.5%	6.5%	83%
SE2	28%	95%	9.1%	9.6%	83%
SE3	30%	94%	8.4%	7.3%	77%
Aggressive	45%	74%	45%	0%	0.0%
Random	28%	87%	22%	12%	41%

The values for strategies behavior are shown in Table 6.6 and Table 6.7 for Kuhn

Table 6.7: Behavior analysis of strategies in Leduc Hold'em. Meaning of columns in order: percentage of wins from all games; percentage of wins with the best hand from all wins; percentage of wins with best hand, when both opponents folded; percentage of loss with the best hand out of all loses; percentage of losing only ante from all loses

	Wins	**1st Rnd Wins**	**BH Wins**	**BH Wins FF**	**BH loss**	**IF/L**
CFR	27%	5.3%	87%	11%	19%	53%
MM	30%	5.2%	86%	9.9%	18%	51%
Aggressive	62%	23%	65%	54%	0%	0%
Random	25%	9.2%	79%	21%	28%	46%

poker and Leduc Hold'em respectively. In the values for Kuhn poker, there are two distinct groups of strategies, group 1 (CFR, Random and Aggressive) and group 2 (MM, SE2 and SE3). Group 1 wins by bluffing more often, the wins with worse hands make 13% and more out of all wins, where for the group 2 it is 6% and less (for MM it is even around 2%). The bluffing itself is not an issue; the strategy can certainly win a lot by bluffing. The group 1 forces other players to fold their hands (which are worse) in 22% and more of wins with the best hand. However, the group 2 forces all other players fold in 9.1% and less.

The other part of this behavior analysis is about losing in a showdown (comparing cards at the end of the game) or by folding hand (where cards are not shown and the folding player automatically loses). The groups are divided still the same. CFR instantly folds only half of its lost hands. Others are lost by betting or passive checking. The group 2 folds more than 77% of hands. This also leads that the Group 2 loses some of their best hands, but when compared to overall payrolls, it pays off to lose a few winning hand than to lose much more with worse hands.

This analysis for Leduc Hold'em surprisingly differs from Kuhn poker. The CFR and MM strategies behave very similarly, with differences only in the order of units of percentages. They win by bluffing around 14% of the winning hands, which is closer to CFR values from Kuhn poker. Forcing other players to fold are 10% of winnings and playing with worse hands around 50% of hands. Except the forcing other players to fold, all these parameters moved to the values of CFR from Kuhn poker. Which leads us to the assumption, CFR has more stable strategy concept, which holds over poker variants, while MM, is more instable and not as conceptual.

6.3.1 Value betting

When we want to analyze value betting, we use graphs showing payoffs over cards (similar to Figure 6.1) in Figure 6.3 for Kuhn poker and in Figure 6.4 for Leduc Hold'em. The red points show average payoff for given position and the green point is the overall average utility. These graphs agree with the conclusion made above, CFR plays with more risk, thus losing more and winning more, while MM, SE2 and SE3 play more safely, trying to get as much as it is possible from Ace, while SE3 is also playing more action with King than MM and SE2.

The graphs for payoffs over different hand cards for Leduc Hold'em shown in Figure 6.4 again suggests the closer relationship between approaches of MM and CFR in Leduc Hold'em than in Kuhn poker. This leads to the conclusion, the table card in Leduc variant brings the insecure position of hand card in the first betting round. Therefore both strategies take as little action with Jack and Queen card as possible.

6.4 Importance of player ordering

The last variety we examine is the importance of player ordering, which we have discovered during experiments done by ACPC server, which preserves the order of players, but rotate them on positions, which can be seen as (A, B, C) -> (C, A, B) -> (B, C, A). Therefore we have run the two possible orderings (A, B, C) and (A, C, B), which covers all possible orderings by rotating positions. During these separate runs, we have discovered unexpected differences between the two basic orderings, shown in Table 6.8 and Table 6.9 for Kuhn poker and Leduc Hold'em respectively.

The tables are organized in the same manner as were in the previous sections. Each strategy in column plays against the opponents in row two times, therefore the two columns for each strategy. The first column is the ordering (A, B, C), where A and B are row strategies and C is column strategy. The second column is the ordering (A, C, B), where the column strategy sits between the row strategies for the whole tournament. For example, the first column (MM) and second row (CFR, Aggressive), where the first column with value 155.8 is for ordering CFR, Aggressive, MM and second column with value 198.4 is for ordering CFR, MM, Aggressive.

The ordering around the game table is important and can make a big difference on the performance of a strategy for Kuhn Poker. The data in Table 6.8 suggests the MM strategy is more robust strategy than all others, because of the least differences between payouts between two different orderings. The CFR strategy is the opposite and is experiencing big differences against same opponents but different sitting.

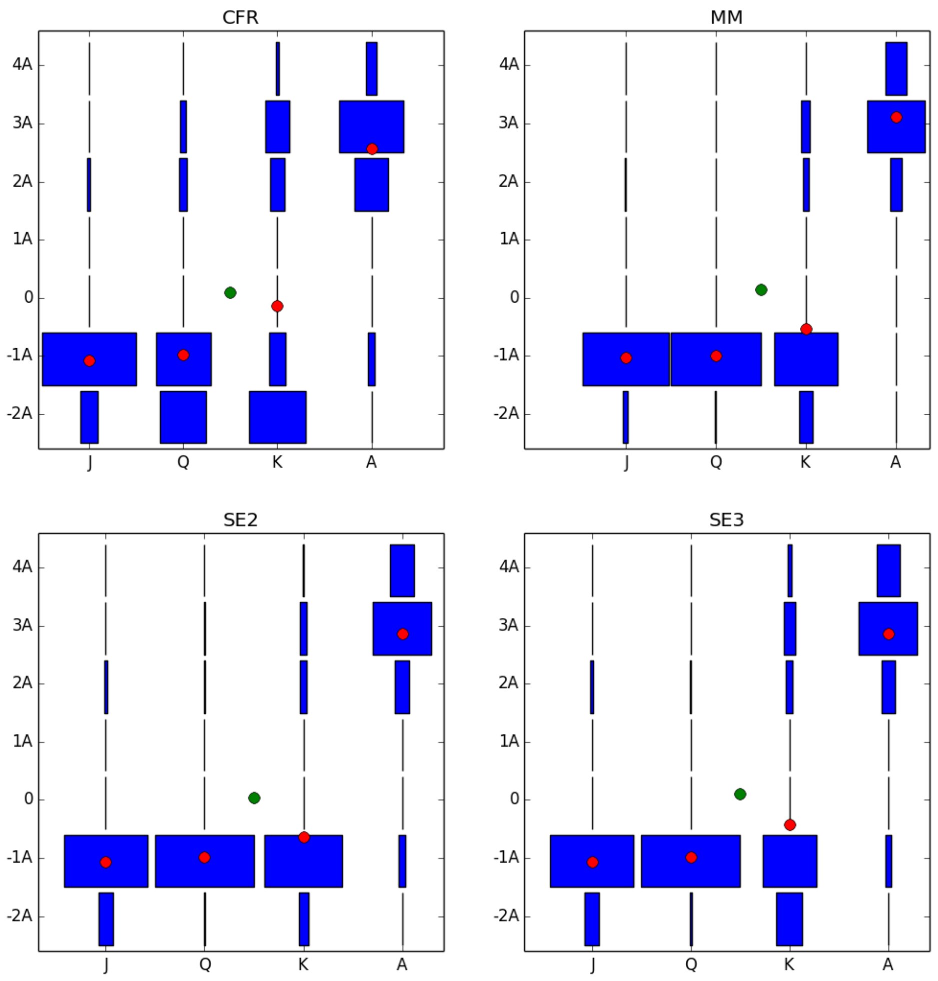

Figure 6.3: Payoffs based on cards for Kuhn poker. Bars show relative number of payoffs of a given value (y-axis) divided by possible hand cards

The aggressive strategy is in the most of the big differences, and it points to the conclusion, this strategy can exploit CFR, SE2 and SE3 when playing on position before them.

The results for Leduc Hold'em (Table 6.9) do not show the same variation of payoffs of CFR as in Kuhn Poker. We assume this is caused by the larger complexity of Leduc variant, which reduces the exploitability of CFR and MM in different orderings. However, the aggressive strategy is still able to gain from the random opponent and score a reasonably

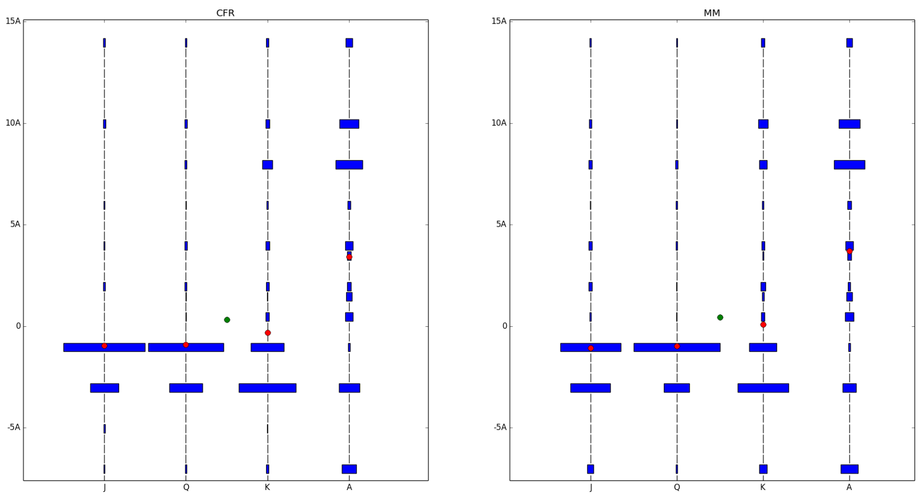

Figure 6.4: Payoffs based on cards for Leduc Hold'em. Bars show relative number of payoffs of a given value (y-axis) divided by possible hand cards

good in tournaments against the rational opponent and random one.

6.4.1 Two same strategies in one hand in Leduc Hold'em

As the last experiments to see the behavior of strategies based on their order around the table, we have run tournaments with (MM, MM, CFR) and (MM, CFR, CFR) for Leduc Hold'em. For both of these triplets, there is only one order. We show the results of the two tournaments in Table 6.10, the first column shows triplets, the rest of columns show average payoffs in the same ordering as the triplets. Therefore, the first row (MM, MM, CFR) shows the average payoffs in the same order as the triplet is written: -181.2 for the first MM strategy, 132.2 for the second MM strategy and 49.1 for the CFR strategy,

The second row suggests the CFR strategy can exploit playing on position after the player with the same strategy. Which is why the first CFR strategy is losing -63.5 and the second CFR strategy is winning 151.9.

This reasoning does not hold for the first triplet (MM, MM, CFR), where the MM strategies do not do any assumptions about the opponents. We are unsure about the reason behind the exploitation of the first MM strategy in the triplet.

Table 6.8: Cross-table of average payoffs with respect to ordering of players in Kuhn poker in ma/h. Column strategies (C) plays according to ordered pairs of row strategies (A, B). The first column for each triplet is value for (C) strategy in ordering (A, B, C), the second column is for ordering (A, C, B)

		C											
		mm		cfr		se3		se2		aggro		rnd	
A, B	cfr, mm	†	†	†	†	85.7	1.1	**86.8**	**-32.6**	-102.7	-319.7	-257.6	-390.2
	cfr, aggro	155.8	198.4	†	†	**43.1**	**207.9**	**-26.0**	**118.6**	†	†	-519.0	-434.3
	cfr, rnd	156.8	177.0	†	†	158.7	174.7	103.4	95.2	128.3	36.8	†	†
	mm, aggro	†	†	**-95.7**	**163.9**	**-57.4**	**35.8**	**-116.6**	**22.0**	†	†	-492.1	-430.3
	mm, rnd	†	†	**80.6**	**233.5**	44.7	134.3	10.7	121.3	170.8	252.8	†	†
	se2, cfr	71.5	87.4	†	†	†	†	†	†	-184.3	-66.9	-341.0	-277.7
	se2, mm	†	†	-54.8	-158.3	†	†	†	†	17.7	-125.6	-143.7	-230.3
	se3, cfr	85.7	97.3	†	†	†	†	†	†	-211.9	-135.5	-355.1	-318.3
	se3, mm	†	†	-98.4	-171.4	†	†	†	†	-69.0	-166.6	-187.3	-253.0
	se2, aggro	103.6	98.9	**-51.7**	**210.3**	†	†	†	†	†	†	-446.6	-377.4
	se2, rnd	109.0	133.1	182.4	238.5	†	†	†	†	351.6	320.1	†	†
	se3, aggro	130.9	126.4	**-72.3**	**168.8**	†	†	†	†	†	†	-468.4	-408.3
	se3, rnd	118.7	142.6	143.6	196.4	†	†	†	†	271.8	251.4	†	†
	rnd, aggro	259.5	239.3	305.9	482.2	136.5	217.1	25.8	126.5	†	†	†	†

Table 6.9: Cross-table with respect to ordering of players in Leduc Hold'em in ma/h. Column strategies (C) plays according to ordered pairs of row strategies (A, B). The first column for each triplet is value for (C) strategy in ordering (A, B, C), the second column is for ordering (A, C, B)

		C							
		mm		cfr		aggro		rnd	
A, B	cfr, mm	†	†	†	†	-670.7	-696.3	-667.4	-780.3
	cfr, aggro	346.2	434.4	†	†	†	†	-1085.8	-970.8
	cfr, rnd	453.7	276.5	†	†	751.1	711	†	†
	mm, aggro	†	†	236.2	350.1	†	†	-1178.1	-1054.1
	mm, rnd	†	†	391	326.6	550.1	579.8	†	†
	rnd, aggro	504	598.3	219.6	374.8	†	†	†	†

Table 6.10: Cross-table for 3-player Leduc Hold'em tournament, 2 instances of the same rational strategy. Values are in ma/h within ± 3 ma/h with 95% confidence

Strategies	Average payoffs		
MM, MM, CFR	-181.2	**132.2**	49.1
CFR, CFR, MM	-63.5	**151.9**	-88.4

Chapter 7

Conclusion

Poker is an interesting domain for testing solution concepts for turn-based zero-sum games with imperfect-information. Nowadays, most of the focus goes to the two-player variant, and the variants with more players have significantly less attention. We have selected three-player poker, because of its bigger complexity and interesting features. Three-player zero-sum games reduce the absolute opposition of players, which we observe in two-player variants.

As the specific variants of poker, we have chosen relatively small Kuhn poker and larger variant Leduc Hold'em. Even though the Kuhn poker is a small game with a number of nodes in the game tree around 600, it is complex enough for finding a Stackelberg equilibrium in reasonable time for all positions around the poker table.

For the chosen variants we have computed strategies according to CFR algorithm, which is a commonly used approach for multiplayer poker domain and represents state of the art. We approached the problem of strategy finding with two other solution concepts, not used in the three-player poker domain: heuristic variant of MaxMin and Stackelberg equilibrium strategy. We have suggested a reasonable way of computing the MaxMin for the realistic situation in the three-player game and provided the formulation of a nonlinear program for computing the Stackelberg equilibrium in the extensive-form game of poker. Because the Kuhn poker is large enough nonlinear problem and the Solvers did not find the solution for Position 1, we substituted the missing strategy for Position 1 by different NLP with the non-optimal solution (SE3) and also by strategy for this position from CFR (SE2).

Lastly, we have run the tournament-based comparison of computed strategies CFR, MM, SE2, and SE3 against each other and also against simple random and aggressive players. During evaluation of tournaments by the total average payoff of strategy, as is common in this domain, we have noticed incompleteness of information, this comparison provides us. Except for the total average payoff comparison, we have compared the strategies based on their strategy for different positions around the table, their ability to bluff and take the risk and also their robustness against different ordering of players in the tournament.

In the total average payoff comparison, the MM won and SE2 lost, while SE3 and CFR scored statistically similar values and ties for the second and third place. MaxMin strategy also shown a good robustness against changing the ordering of players and CFR showed its weakness against aggressive strategy playing before CFR.

The MaxMin strategy along with the SE2 and SE3 strategies plays a careful game. SE strategies are performing well on beginning positions, showing their ability to lead the game in they favor while losing more on the last position around the table. CFR, which is trying to model the opponents playing as best as they can, taking more risks, therefore losing more compensated by winning more.

7.1 Further work

This thesis provides a detailed analysis of novel approaches for computer players in the three-player poker. We recommend two main directions for following research: computing the strategies from this thesis for larger games (CFR, MM) or with more resources (SE) and analysis of other possible concepts for three-player poker games.

Therefore, we propose to extend our work to even larger variants of poker and evaluate behavior and performance of the MM and SE strategies in the more complex games.

Secondly, computing the Stackelberg equilibrium with more resources and finding a faster way of computing these strategies.

Thirdly, because the CFR algorithm produces different strategies based on implementation details. We suggest deeper analysis of the performances of these strategies along with a method for choosing the best strategy.

Lastly, research other possible concepts for three-player games, such as minmax strategies, correlated equilibrium, Stackelberg equilibrium, where followers play correlated equilibrium. Also examine other possibilities than offline strategies (such as MM, CFR, and SE) and include learning players into the multiplayer poker domain.

Bibliography

[1] Annual computer poker competetion. `http://www.computerpokercompetition.org/index.php/about`. Accessed: 2017-05-05.

[2] Stefano Conti. *Algorithms for Finding Leader-Follower Equilibrium with Multiple Followers*. PhD thesis, Politecnico di Milano, 2013.

[3] Joseph Czyzyk, Michael P. Mesnier, and Jorge J. Moré. The neos server. *IEEE Journal on Computational Science and Engineering*, 5(3):68 —— 75, 1998.

[4] Elizabeth D. Dolan. The neos server 4.0 administrative guide. Technical Memorandum ANL/MCS-TM-250, Mathematics and Computer Science Division, Argonne National Laboratory, 2001.

[5] David Silver et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 2016.

[6] F. Southey et al. Opponent modeling in poker. *UAI*, 2005.

[7] M. Moravčík et al. Deepstack: Expert-level artificial intelligence in heads-up no-limit poker. *Science*, 2017.

[8] Oskari Tammelin et al. Solving heads-up limit texas hold'em. *Proceedings of the 24th International Joint Conference on Artificial Intelligence*, 2015.

[9] R. Gibson. *Regret minimization in games and the development of champion multiplayer computer poker-playing agents*. PhD thesis, University of Alberta, 2014.

[10] William Gropp and Jorge J. Moré. Optimization environments and the neos server. In Martin D. Buhman and Arieh Iserles, editors, *Approximation Theory and Optimization*, pages 167 – 182. Cambridge University Press, 1997.

[11] H. W. Kuhn. A simplified two-person poker. *Annals of Mathematics Studies, 24, vol. 1*, 1950.

[12] Marc Lanctot. *Monte carlo sampling and regret minimization for equilibrium computation and decision-making in large extensive form games.* PhD thesis, University of Alberta, 2013.

[13] F. Hsu M. Campbell, A. J. Hoane Jr. Deep blue. *Artificial Intelligence 134*, 2002.

[14] D. Szafron N. A. Risk. Using counterfactual regret minimization to create competetive multiplayer poker agents. *AAMAS*, 2010.

[15] Y Narahari. *Game Theory and Mechanism Design.* World Scientific Publishing, 2014.

[16] J. F. Nash and L. S. Shapley. A simple three-person poker game. *Annals of Mathematics Studies, 24, vol. 1*, 1950.

[17] Noam Nisan. *Algorithmic Game Theory.* Cambridge: Cambridge University Press, 2007.

[18] R. A. Waltz R. H. Byrd, J. Nocedal. Knitro: An integrated package for nonlinear optimization. 2006.

[19] Y. Shoham and K. Leyton-Brown. *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations.* IT Pro. Cambridge University Press, 2008.

Appendices

Appendix A

Content of CD

- `doc/` PDF file of the written part of this thesis. Folder with LaTeX source codes for the written part.
- `sources/src/` Folder with JAVA project, where algorithms are implemented
  - `Game package` contains implementation of definition of Poker game tree
  - `Algorithms/CFR.java` contains implementation of CFR Algorithm
  - `Algortihms/NFGStackelberg.java` contains implementation of generation of GAMS files for NLP in Formulation 4
  - `Algortihms/NFGStackelberg18.java` contains implementation of generation of GAMS files for NLP in Formulation 3
- `strategies/` contains text files with definition of strategies for Kuhn poker and Leduc Hold'em
- `acpc_server/` folder with ACPC server
- `acpc_server/play_match.pl` run script of ACPC server for one tournament
- `acpc_server/kuhn.limit.3p.game` definition of Kuhn poker for ACPC Server
- `acpc_server/leduc.limit.3p.game` definition of Leduc Hold'em for ACPC Server
- `acpc_server/KuhnYouHandleIt/` folder with implementation of poker bot player[1] for Kuhn Poker

[1]For implementation of both clients connecting to ACPC server, we have used KuhnYouHandleIt from Charles Evans from UCL as a baseline

- `acpc_server/LeducYouHandleIt/` folder with implementation of poker bot player for Leduc Hold'em
- `acpc_server/run_test_experiments.sh` example bash script of running experiments for Kuhn poker and Leduc Hold'em