FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Using a High-Level Language to Build a Poker Playing Agent

Nuno Pedro Silva da Cruz

FINAL VERSION

Report of Dissertation
Master in Informatics and Computing Engineering

Supervisor: Luís Paulo Reis

June 2009

Using a High-Level Language to Build a Poker Playing Agent

Nuno Pedro Silva da Cruz

Report of Dissertation
Master in Informatics and Computing Engineering

Approved in oral examination by the committee:

Chair: João P. Moreira

External Examiner: José Torres

Internal Examiner: Luís Paulo Reis

June of 2009

Abstract

The past decade was witness to major breakthroughs in stochastic and imperfect information games, especially through the research in poker. This particular research has become the target of some hype in recent years following the game's popularity growth due to online gambling and TV shows. One branch of this research deals with creating and augmenting an artificial player to be able to beat professional poker players. Although somewhat successful, this branch of poker's research is still heavily based on computer programming skills and Artificial Intelligence knowledge instead of poker domain expertise.

What this work proposes is a high-level language capable of creating complex strategies for autonomous artificial players. This would approach poker players and Artificial Intelligence by enabling users without the mentioned skills to create their own custom players and strategies.

A user-friendly application was created in order to assist non-IT savvy poker players to create autonomous poker players in the LIACC's Texas Hold'em Simulator. Although unable to build truly competitive agents, the application serves as proof-of-concept.

Resumo

A última década foi testemunha do surgimento de descobertas importantes no domínio dos jogos de natureza estocástica e de informação imperfeita, especialmente através da investigação realizada no jogo de poker. Este domínio em particular tornou-se alvo de algum mediatismo nos últimos anos devido ao crescimento da popularidade do jogo online e em programas de TV. Um dos ramos desta investigação trabalha com a criação e o melhoramento de jogadores artificiais de forma a serem capazes de vencer jogadores profissionais de poker. Embora algum sucesso tenha sido atingido, este ramo ainda é extremamente baseado nas capacidades de programação e nos conhecimentos em Inteligência Artificial, invés de ser baseado no domínio pericial.

O que é proposto neste trabalho é uma linguagem de alto nível capaz de criar estratégias complexas para jogadores artificias autónomos. Desta forma aproximar-se-ão a Inteligência Artificial e os conhecedores de poker ao permitir que estes últimos sejam capazes de criar os seus próprios jogadores e estratégias, sem necessitar de conhecimentos em programação ou em Inteligência Artificial.

Foi criada uma aplicação de interface amigável para auxiliar a criação de jogadores de poker autónomos no simulador de Texas Hold'em do LIACC. Embora não seja capaz de produzir agentes verdadeiramente competitivos, a aplicação serve como prova de conceito.

“I'm a great believer in *luck*, and I find the harder I *work*, the *more* I have of it.”

“Acredito fortemente na *sorte*, e descubro que quanto mais *trabalho*, *mais sorte* obtenho”

-- Thomas Jefferson

Contents

List of Figures

List of Tables

Abbreviations

AI	Artificial Intelligence
AAAI	Association for the Advancement of Artificial Intelligence
CRPG	Computer Research Poker Group
IRC	Internet Relay Chat
LIACC	Artificial Intelligence and Computer Science Laboratory (Laboratório de Inteligência Artificial e Ciência de Computadores)

Chapter 1
Introduction

"We can only see a short distance ahead, but we can see plenty there that needs to be done"

-- Alan Turing

Games are one of the oldest areas of endeavor in Artificial Intelligence (AI). Although AI is a big field that has its research finding nest in a wide variety of domains like Economics, Computer Science, Medicine, Robotics and so much more, the study of games like board games, card games and other mathematical games of strategy have long and distinguishably been playing an important role in many of AI advancements.

Games possess a number of properties that remain attractive to AI research because of the numerous scientific complex problems provided while being based on well-define rules, using simple logistics, having an unambiguous definition of success and allowing for measurable results.

Challenges presented by Poker did not pass unnoticed to the research community in the last decade, following the popularity growth of the game. The game's recent boom in popularity worldwide has granted more exposure to the work being conducted on this topic, extensively broadening the research on the particularities of Poker.

Computer programs, mathematical analysis, psychological studies have all been focusing on Poker in order to retrieve knowledge but also wealth, as the industry around this particular game generates values of billions of dollars every year. With the recent online poker boom, a large number of tools have been created to assist players in their game. All of those tools, with a few exceptions, are either essentially mathematical or more of an empirical nature. The former mainly calculates bets and pot size while measuring against position, hand strength and drawing probabilities. The latter records and stores information of the games played and assists with the statistically correct move. One of the exceptions are computer programs that are used to create artificial players, so poker players can train and perfect their game against computer or human made agents. Unfortunately the creation of such poker agents necessarily requires knowledge in computer programming, hindering poker player adoption and research alike.

1.1 Categorization of Games

With the current exception of Robot Soccer, AI rarely applies to "physical" games. The main reasons for this are the large range of possible actions available in any given moment and the use of imprecise rules in these games. On the opposite, games that engage the intellectual faculties of Humans are especially attractive to AI. These games generally have a set of clearly defined rules, becoming easier to study, and possess a specific goal which makes it possible to test the success of different approaches for a specific game.

Games can be characterized by the various properties they embody but since important characteristics are induced by the types of uncertainty present in a game [51, pp. 333-345] it's those uncertainty factors that are used to categorize a game. Two parameters usually used are: whether the players can fully observe the current state of the game and whether or not the game contains chance elements. If one or more players can't fully observe the current state of the game, the game exhibits *state uncertainty*. Because the player has partial or imperfect information regarding the state, these are called incomplete or imperfect information games. On the opposite, if a player has complete knowledge of the entire game state, then it is called a complete or perfect information game. If a game contains elements of chance, for example throwing a dice, it embodies *outcome uncertainty* and is referred to as stochastic. Otherwise it is deterministic. Table 1 shows examples of popular games categorized according to state and outcome uncertainty.

Table 1: Examples of various games categorized by forms of uncertainty

Type	Perfect Information	Imperfect Information
Stochastic	Backgammon	Poker
Deterministic	Chess	Battleships

1.2 Artificial Intelligence and Poker

Artificial Intelligence (AI) is the study and design of so called intelligent agents. An intelligent agent is a system that perceives its environment and takes actions which maximize its chances of success [3, 34-36]. Until 90s decade, AI research was focused mainly on deterministic games of two players with perfect information which made chess, checkers and even backgammon the main games for AI research. Backgammon is the perfect example of AI achievement by making programs able to be on pair with the best Human players even in games with stochastic elements. In the last few years research on chess and similar games lost significance as the strategy for improvement as become known and resulting programs have achieved dominance over human players [14, pp. 1518-1522] [27] [63, pp. 189-193] [20, pp. 21-29].

Following the success of many endeavors through different games, researchers are now focusing on games of imperfect information and of stochastic nature as opposed to what has been the trend, with Poker being identified as the next benchmark problem for these kinds of games [1] [5] [4]. Despite its popularity, Poker offers many interesting challenges, that aren't available in traditional perfect information games and thus holds a particular interest for research. For instance, in Poker the best strategy aims to make the play which provides the best expected value but, contrary to backgammon for example, in Poker the best play is deeply correlated with the opponent strategy [24, pp.3]. Not only that but managing multiple competing agents, assessing risk management, performing opponent modeling, creating deception like bluffing and trapping, calculate perfect odds and implied odds, use mixed strategies and dealing with unreliable information are all but a few of the necessary properties for a complete poker strategy. All these properties make Poker a highly potential research topic for AI.

Regarding AI and Poker, more and more research has been focused on the game in the last decade. Previous to that, researchers like John Nash [56] and John von Neumann [59] made use of simplified poker to illustrate the fundamental principles of Game Theory [7] but the strategic aspects of Poker were left largely unstudied. Recently, with notable contributions by many members of the scientific community, particularly those with the University of Alberta CPRG (Computer Poker Research Group), AI research is finally stepping up to the challenges that this popular game presents.

1.3 Motivation

"The guy who invented gambling was bright, but the guy who invented chips was a genius"

-- Julius Weintraub

'Poker craze' has never been as widespread or growing as quickly as it is right now. With the wealth of Poker information available online, in books and articles, in TV shows and movies, this game's popularity is not likely to diminish or disappear anytime soon.

The impact of Poker is undeniable significant at various levels worldwide. At an economical level, especially but not restricted to, the global gambling industry has been achieving record profits. Online poker, since being established in the 1990s, has grown rapidly and has been partly responsible for the dramatic increase in the number of poker players worldwide.

At a social level, ever since the mainstream adoption by TV, Poker as evolved from a casino game to a real sport in the United States of America, with tournaments and worldwide competitions.

At a scientific level, AI research has been focusing more on how to deal with problems found in imperfect information games like Poker. Poker's game mechanics embodies a considerable number of complex but extremely interesting problems which studying and solving will most likely have potential application in other fields.

The combination of technology, popularity, curiosity, and, the allure of big money, all of this makes the investment in poker's research likely to become very profitable and useful in the near future.

1.4 Document Structure

Chapter 1 introduces the theme of this research as well as its motivation. Chapter 2 describes the game of poker itself, covering its origins and evolution along the years, in particular the Texas Hold'em variant. Texas Hold'em game rules and gameplay mechanics are described in detailed. An overview of other popular poker variants is also present in the chapter. Chapter 3 covers over a decade of research in poker, presenting the current state-of-the-art in this field. Chapter 4 describes the work done under this thesis research, covering both theoretical and practical work. Chapter 5 presents experimental results, indicates this thesis contributions as well as conclusions and future work.

Chapter 2
POKER: The Game

"Poker's a day to learn and a lifetime to master."

-- Robert Williamson III

"With so many claims to the name, the chance of narrowing down on the exact birthplace of the History of Poker is parallel to the chance of hitting a Royal Flush!"

-- ThePokerFather.com

2.1 Ultimate Origins

The origin of Poker has been a well-debated topic throughout history. There are as many variations regarding the possible birthplace of Poker as there are of the game itself. One popular theory places 17th century Persia has the birthplace, in the form of a game called Âs Nas. This 20 or 25-card game, divided between 4 or 5 players respectively, included betting rounds and made use of hierarchical hand rankings [8]. Such mechanisms resemble those found in Poker so it was not surprising that by the time Poker became an interesting research topic, claims were made in that sense [36, pp. 163] [90]. But this theory is often rebuffed by several historians mainly [78] because of the absence of any description of this game earlier than 1890 and due to the fact that "*Âs"* is not a card related word in Persian and most likely derives from the French word for ace. These arguments lead several researchers to believe that a European vying game [40] [78] was the inspiration for Âs Nas.

The spotlight was then directed to popular European card games of the 17th century, games that could share parts of the unique combination of mechanisms found in Poker. As expected those games did exist although in different forms. One of those was a German game named Pochen. This was a game that involved hand ranking, betting and also bluffing which can all be found in Poker [74]. Another popular game of that time was Poque, a French variation of the previously mentioned German game. Both these games were tripartite games played from 3 to 6 players that used a 32 or 36-card [75] deck with Pochen requiring a staking board of special design. As is easy to see both games were still quite different from the earliest form of Poker, a one-part game played with a 20-card deck equally divided among four players, but the game Poque carried a significant etymological root considering the French colonies role in north American territories. Given that Poker originated in culturally French territory, its likeliest immediate ancestor is indeed Poque but for that to be possible, the game had to be developed within a community that was already acquainted with a 20-card vying game and decided to use the same stripped pack for a new version game based only on Poque's vying section…

2.2 How It Came To Be

The earliest contemporary reference to Poker occurs in 1836 [44, pp. 128-130] although two slightly later publications show it to have been well in use by 1829 [78]. The birth of Poker has been convincingly dated to the 19^{th} century. It appeared in former French territory centered on New Orleans [57] [18, pp. 94-95] which became part of the United States of America by the year 1803. Gambling saloons in general and, the notorious floating saloons of the Mississippi river in particular, allowed Poker to spread north along the river and west with the gold rush that swept the American territory that century.

The earliest known form of Poker was played with a 20-card pack featuring Ace-King-Queen-Jack-10 evenly dealt amongst four players. There was no draw and bets were made on a narrow range of combinations: one pair, two pair, triplets, full (so called because it allows for all five cards to be active) and four of a kind. Unlike it is seen nowadays in some variations of Poker, in which the top hand (royal flush) can be tied in another suit, the original top hand of four Aces or four Kings and an Ace was absolutely unbeatable [57] [82, pp. 112-113] [78]. The game then borrowed great inspiration from the English card game Brag, gradually adopting a 52-card deck which enabled more than four players to participate, allowed the flush to be introduced into the hand ranking and most importantly provided enough cards for the Draw [60]. This last one increased the excitement of the game by adding a second betting interval and enabling poor hands to be significantly improved, elevating Poker from a gambling game to a skill game.

The 19^{th} century civil war that swept the United States of America saw Poker experience many more changes and innovations resulting in structural divisions of the game. One of most notorious was the birth of the Stud Poker, in its 5-card version, credited to be a cowboy invention by the year 1864 [83]. Finally following Draw and Stud, a third major structural division of the Poker game appeared around the year 1920, where one or more cards were 'shared' among all players. Making use of communal cards, these games became known as Community Poker. Today the most popular version of this type of Poker is represented by the Texas Hold 'em game.

Since its humble beginning on the banks of the Mississippi River, in over two centuries, the popularity of this widely played game has grown tremendously and evolved into numerous variations and sub-variations.

2.3 The Rise of (Texas Hold'em) Poker

In the early 1960s Poker was an illegal activity throughout the United States of America with the exception of the 'gambling states' of Nevada and California. But the allure for big rewards and high stakes excitement made sure poker remained popular within the gambling community. In 1970 a particular event would open Poker to media coverage. Following the first ever major poker tournament, the World Series of Poker was created, pitting players against each other in a winner-take-all world championship. Nationwide coverage was achieved and gone were the days when Poker was played in back rooms by a handful of regular players [17, pp. 37-45] [53].

Over the years the WSOP grew and so did its popularity. The 2003 WSOP was the first edition to saw extensive television coverage, broadcasting each stage of the event, allowing audiences to gasp or cheer as the event unfolded. This mainstream acceptance allowed Poker to make an evolutionary jump from being viewed as a game to being viewed as a sport in the United States and partially in the rest of the world. The premise for this is that Poker allows players to do what is impossible in almost any other sport: Start from nowhere and be able to challenge the best in the game. The payout in high stakes poker is still very rewarding and WSOP is no different having paid over 29 million dollars in prize money for the top nine players of the 2008 WSOP edition [70].

And the main tournament format in these major Poker events? Texas (No Limit) Hold'em [92]. This particular version of poker is more confrontational and explosive than traditionally accepted formats such as 5-card draw or 7-card stud. In high stakes games, Texas Hold'em is simply the most exciting format of poker especially when an audience is watching. Reaping the benefits of mainstream coverage, Texas Hold'em became the most popular version of poker in casinos, internet casinos and gambling arenas in general.

2.4 The Very Basics

One should start by saying that there is no such thing as the official rules of Poker. Some rules are universal, some are considered standard even though there are some places that do not use them, and some are so varied that a player should be aware of the rules in the area whenever playing in a new cardroom.

Following the rules presented in Robert Ciaffone's book [80] and in Roy Cooke's book [23, pp. 12-18] Poker is played with a standard deck of 52 cards although some variant games use multiple packs or add a few cards called jokers or wild cards. The cards, ranked from high to low are Ace, King, Queen, Jack, 10, 9, 8, 7, 6, 5, 4, 3, 2 as depicted in figure 2.1. Ace is the one card that can either be high or low, assuming the nominal value of 1, but is usually high. There are four suits: spades, hearts, clubs and diamonds. However, no suit is higher than another. Figure 2.2 depicts standard symbols for each suit. All poker hands contain five cards where, usually, the highest ranked hand wins. In some games it's the lowest hand that wins and in other games both the highest and the lowest hands win.

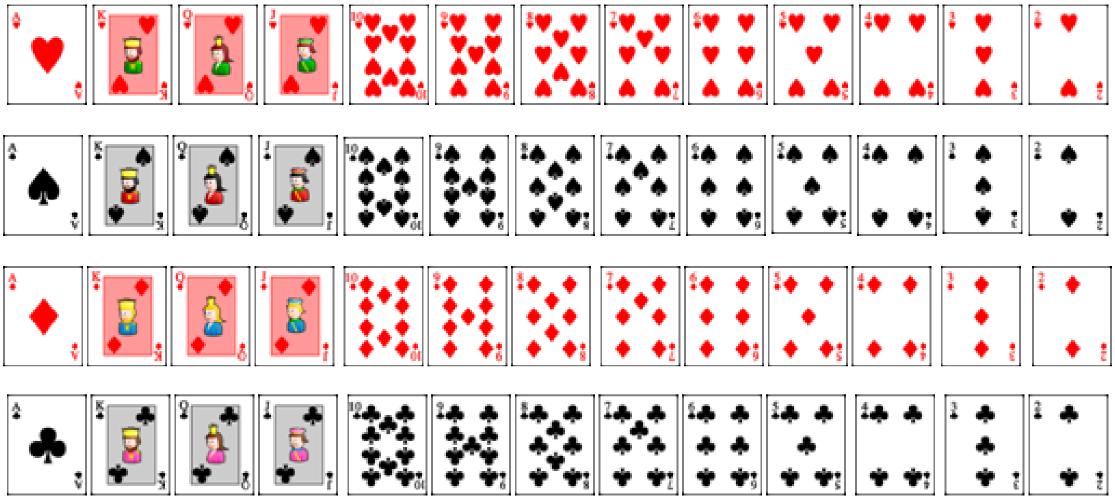


Figure 2.1: Ranking highest cards from left to right

Some games use blank cards called wild cards which can take on whatever suit and rank the player requires. Sometimes jokers will be used as wild cards. Other times the game will specify which cards are wild; deuces, one-eyed jacks, or something else.

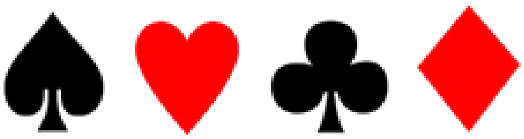


Figure 2.2: Spades, hearts, clubs and diamonds (from left to right)

Poker is won by winning pots, which is money or chips wagered during the play of each hand or round. Pots are won by revealing the best hand at the end of the round, also known as showdown, or by having all opponents relinquishing claim over the wagered pot, also known as folding. Winning the most pots doesn't necessarily mean victory although winning most of the best pots will undoubtedly make it easier to achieve it.

2.5 Hands Ranking

A hand refers to the cards in possession of a player and always consists of five cards. In games like Texas Hold'em, where more than five cards are available to each player, the best five-card combination of those cards is the one that plays.

The ranks of the various possible hands is based on the probability of being randomly dealt such a hand from a well-shuffled deck, therefore, the rarer the hand, the highest it's ranked. For example a straight flush, which is much less likely to occur than a full house, is ranked higher. That's why three-of-a-kind beats two pair, which in turn beats one pair.

Hands are ranked first by category then by individual card ranks: even the lowest qualifying hand in a certain category defeats all hands in all lower categories. The smallest two pair hand defeats all hands with just one pair or high card. Only between two hands in the same category are card ranks used to break ties.

Hands are ranked (from high to low) as follows [25, pp. 16]:

2.5.1 Five-of-a-Kind

A five-of-a-kind, which is only possible in games using wild cards, is the highest possible hand. If more than one hand has five of a kind, the higher cards wins. For example five aces beat five kings, which beat five queens, and so on.

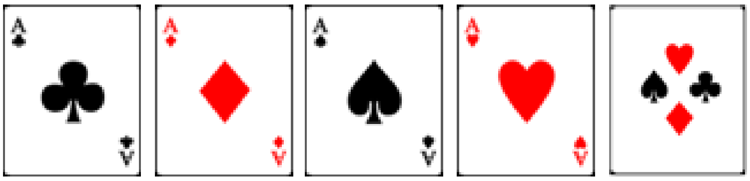


Figure 2.3: A possible hand of five aces composed of four natural aces and a wild card

2.5.2 Straight and Royal Flush

A straight flush is a sequence of five cards in order, such as K-Q-J-10-9, that share the same one suit. The particular sequence with ace high straight flush, A-K-Q-J-10, is called a royal flush and is the highest ranked hand in games without wild cards.

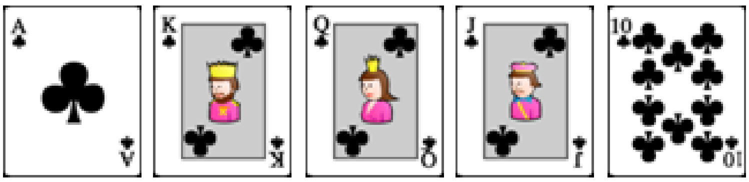


Figure 2.4: Example of a Royal Flush Hand

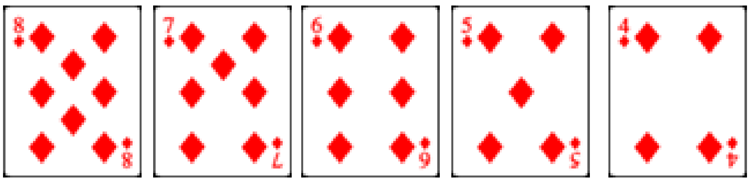


Figure 2.5: Example of a Straight Flush Hand

2.5.3 Four-of-a-Kind

Four-of-a-kind is simply a five-card hand composed of four cards of the same rank plus one unrelated card, for example Q-Q-Q-Q-2. If there are two or more hands that tie, the hand with the higher rank four-of-a-kind wins, for example four kings beat four jacks. In games with wild cards it is possible for two players to have four-of-a-kind of the same rank. In this case the one with the high card outside the four-of-a-kind wins.

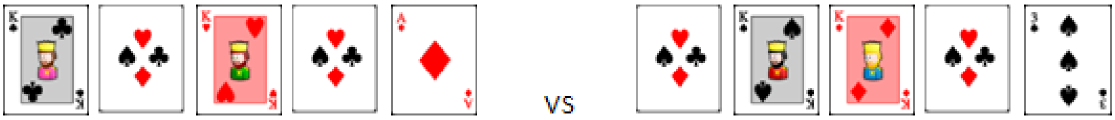


Figure 2.6: Both hands hold four kings but the left one has a higher kicker card thus it wins

2.5.4 Full House

A full house hand is composed of a three-of-a-kind and a pair, such as K-K-K-5-5. Ties are broken first by the three-of-a-kind and only then, if necessary, by the pair. For example K-K-K-2-2 beats Q-Q-Q-A-A which in turn beat Q-Q-Q-J-J.

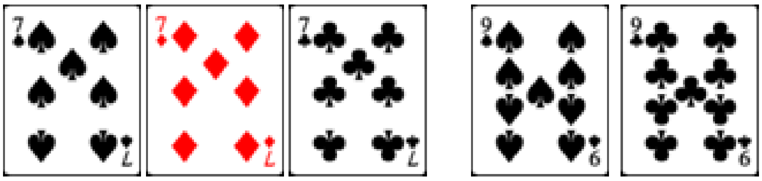


Figure 2.7: Example of a gull house hand also called "seven full of nines"

2.5.5 Flush

A flush is a hand of any five cards of the same suit, such as J-8-5-3-2, all of spades. The cards must not be in sequence otherwise it becomes a royal or straight flush. If there is a tie between flushes, the hand is decided by comparing the highest card in each flush.

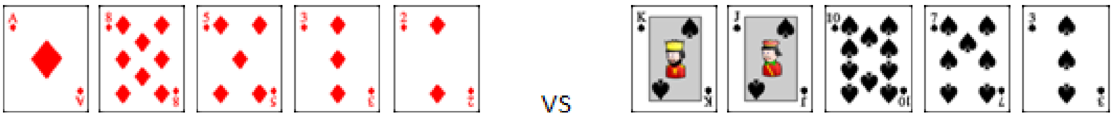


Figure 2.8: Both hands have flushes but the left one has a higher card in its flush thus it wins

2.5.6 Straight

A straight is a hand of five cards in sequence for example 8-7-6-5-4. The ace can either be high or low, like 5-4-3-2-A. However a straight cannot 'wrap around', such as 3-2-A-K-Q. If a tie between straights happens, the highest card in the sequence determines the winning hand. For example A-K-Q-J-10 beats K-Q-J-10-9. If two straights have the same value, meaning the same rank but with different suits, they split the pot.

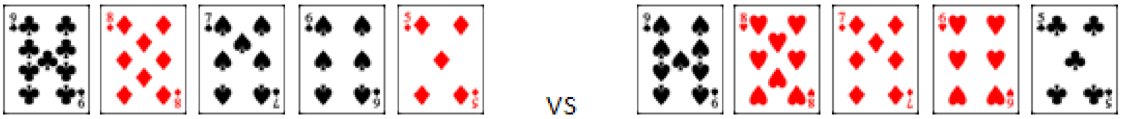


Figure 2.9: A tie between straights

2.5.7 Three-of-a-Kind

This hand is composed of three cards of the same rank together with two unrelated cards. Like before, the highest three-of-a-kind wins. If both are of the same rank then the winning hand is decide by comparing the highest card of each hand.

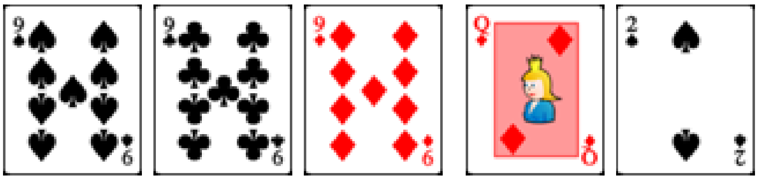


Figure 2.10: Example of a triple, also known as set

2.5.8 Two Pair

This hand features two distinct pairs of cards and a 5th unrelated card. The highest pair wins in case of a tie. If both hands have the same high pair then the second highest pair wins. If both hands have the same pairs, the remaining highest card determines the winner.

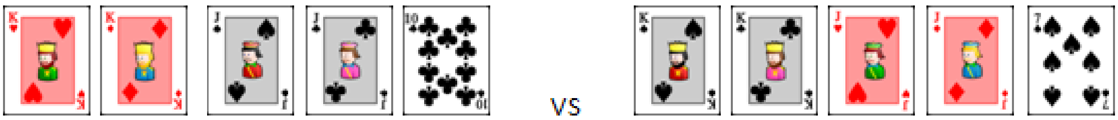


Figure 2.11: Both hands have two pairs of kings and jacks so the one with the higher kicker win

2.5.9 Pair

A pair refers to two cards of the same rank plus three unrelated cards of different ranks. The hand with the highest pair wins, for example a pair of kings beats a pair of queens. If two players have pairs of the same rank, the highest of the other three cards is compared in order to determine the winner.

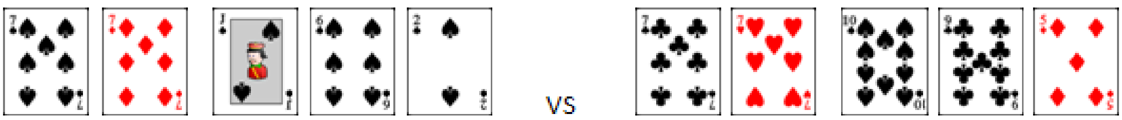


Figure 2.12: Both hands hold the same pair but the left hand has a higher kicker

2.5.10 High Card

This is basically any hand which doesn't qualify as any one of the above hands. If no hand has a pair or better then the hand with the highest card wins. If multiple people tie for the highest card, then the second highest is compared, then the third highest etc.

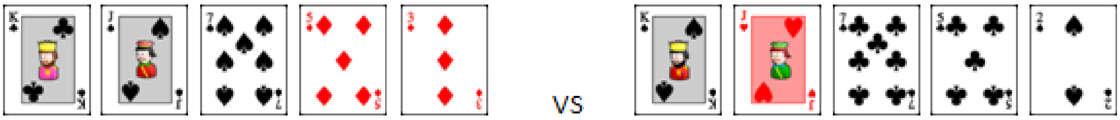


Figure 2.13: The hand on the left wins with the fifth highest card

2.5.11 Low Hands

There are some poker variations, like Razz or Lowball, where the best low hand composed of five cards in sequence determines the winning hand. In most of those games the hand 5-4-3-2-A, or ace-to-five low, is the highest ranked hand.

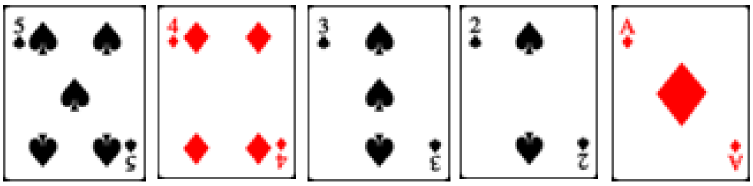


Figure 2.14: Example of the highest ranked hand in lowball. This hand is known as ‘wheel’

2.6 Betting

Betting is key to Poker since it allows to minimize losses when holding a poor hand but also allows to maximize wins with good hands. Betting is typically done in clockwise order and when it’s a player’s turn to bet the available choices are:

- Check: This option is only permitted if no player as already bet in the current round. A check is a bet of zero. By checking, the player retains the right to call or raise any bet made subsequently by another player.

- Call: To call a bet is to wager enough to match what has been bet into the pot since the last time the player bet.

- Bet: A bet is a wager of a certain amount of chips or money. The amount of a bet may be limited by the rules of the game.

- Raise: The raise is a particular kind of bet. To perform a raise the player first bets enough to match what has been bet since the last time it was his time to bet and then increases the amount wagered by betting a new amount. This new amount can be limited or not depending on the type of game being played. Although this seems a two-step operation, it is not.

- Fold: If a player decides not to check or call or raise, a player can fold. This drops out the player's hand, relinquishing any possibility of winning the pot.

Players should speak out loud their intentions when it's their turn to play and avoid going back and forth with the chips/money since it is considered rude and most of the times simply not allowed.

Betting continues until everyone calls or folds after a raise or initial bet. At the end of the round, the highest hand still in the game wins the pot.

While there are different rules for each specific variation of Poker out there, Poker really is this simple to start playing with. Yet within its simplicity lays a textured game structure that's fascinating, sometimes enjoyable, and a real challenge to master.

2.7 Flavors of Poker

Poker is a generic name for hundreds of games but they all fall within a few interrelated types. Some of those variation as well as basic rules of play are presented in this section.

2.7.1 5-Card Draw

This type of Poker rose from relative obscurity during the American Civil War to become the most popular game for almost a century.

Like in most games an *ante* must be paid, with the amount varying by game, just to get dealt cards. After the ante each player is dealt five cards face down. Starting with the player to the dealer's left each player can check, bet or raise. Once the first round of betting is complete each active player has the option of discarding from one up to five cards, if the rules of the game don't restrict it, and receive replacements from the dealer. After the *draw*, there is a final round of betting, usually starting with the player who opened the pot. In the showdown the best high hand wins. Note that in many of these games a joker is used sometimes as a wild card. [23, pp. 40].

2.7.2 Straight Draw

Straight Draw Poker is in all similar to 5-card draw but there are no wild cards. There can still be restrictions to how many cards a player is allowed to draw each time.

2.7.3 7-Card Stud

Shortly before WWII this type of Poker became the most popular variant and maintained its position for about 40 years, mostly with the help of the new and thriving Las Vegas casino industry after the state of Nevada legalized casino gambling.

The game starts with the dealer giving two cards face down to each player and one card face up to each player. The player with the highest card showing opens the first betting round. Following this betting round another card is dealt face-up to each player, followed by a betting round, followed by a third card face-up, followed by a betting round, followed by a fourth card face-up, followed by a betting round, followed by the last card dealt face-down, concluded by the final betting round.

The player that opens each betting round is the player that has the best hand showing out of the cards face-up. In the end, each player takes five cards out of seven that make up the best hand, with the best high hand winning. [84, pp. 273-274] [23, pp. 22-29]

2.7.4 Razz

Razz is played like 7-card stud. The twist is that in Razz, the lowest hand wins. Each player is dealt two hole cards, meaning they're dealt faced down, and one card faced up. The dealer then gives each active player three more cards facing up, and then a final card facing down. Each player ends up with seven cards, four face up and three face down. At the showdown the player holding the best low hand using only five of his seven cards wins the pot. In this game aces are always low and flushes and straights have no effect on the value of a hand. The best possible hand is A-2-3-4-5. [85, pp. 101-105] [23, pp. 29]

2.7.5 Lowball Draw

In Lowball the lowest hand at the table wins the pot. In standard Lowball the best low hand is A-2-3-4-5 followed by A-2-3-4-6 and so on. Although in standard Lowball straights and flushes are ignored, some tables count straights and flushes as high, and therefore, contribute to a bad Lowball hand.

The dealer starts by giving each player five cards face down. There is a round of betting, starting with the player to the dealer's left. After the initial betting round, players may draw up to five cards. Following the draw there is a final round of betting. Usually the rules of play require a 7 low or better to bet in order to win any money to put into the pot after the draw. The lowest ranking hand in the showdown wins the pot. Frequently the joker is used as a wild card. [84, pp. 275] [42, pp. 165-171]

2.7.6 High-Low Split

This name covers several popular forms of Poker. Essentially in High/Low games the pot is split between the best high hand and the low hand at showdown. This is a feature that can be added to just about any Stud poker games, so the game can be 5-card draw, 5-card stud or 7-card stud, in addition to other game's rules. Sometimes, however, the rules may require that players declare whether they are going for the high, for the low, or for both. Like in Lowball

Draw, in 5-card High/Low split games the best low hand is always A-2-3-4-5. Omaha/8, a variation of the High/Low split, requires a player to have 8 low or better to qualify for low. If no one has an 8 low or better, the best high hand wins the whole pot. [84, pp. 276]

Despite so many variations there have been three types of Poker that have, in turn, dominated the modern poker scene: 5-Card Draw, 7-Card Stud and nowadays **Texas Hold'em**.

2.8 Texas Hold'em Gameplay Mechanics

"Texas Hold'em is the Cadillac of Poker games". Although in his book [17, pp. 337] Doyle Brunson was referring specifically to No Limit, Texas Hold'em is generally considered to be the most strategically complex poker variant. This particular variant of Poker, a community card game, rose to prominence in the 1970's when it was featured as the title game in the World Series of Poker and is today, indisputably, the most popular poker game in the world.

Texas Hold'em is a game that uses the player's position at the table to strategically improve its game. This is not achieved by having players physically changing seats every round, which is not allowed, but by the use of buttons. At any given time there is a dealer's button attributed to a player. This marks the top position at the current table and also determines who the small blind and who the big blind are. These are the next two players sitting to the dealer's left. Figure 2.15 depicts the position of the blinds in regard to the dealer's button.

Figure 2.15: Positions at the table

A game of Texas Hold'em begins with what is called as the pre-flop. This is a stage where antes are paid and each active player is dealt two hole cards, facing down. After the cards have been dealt, the first round of betting takes place starting with two players being forced to bet in order to build the pot for contest. The player to the dealer's left will be paying the small blind and the next player will be paying the big blind, which is usually two times the small blind. The value of these blinds is usually increased in a define amount of time, like 1 hour in live poker games for example. The blinds and dealer positions will rotate around the table, in clockwise direction, once the winner for the pot is determined creating a new dealer, a new small blinder and a new big blinder. After all players received two hole cards the initial betting round begins, in turn, starting with the player at the left of the big blinder. At his turn each player can decide to stay in the hand by checking, calling, betting and raising or abandoning the hand by folding.

If the game continues, the dealer discards the top card facing down so it remains unseen to everyone at the table. Three community cards, collectively called the flop, are dealt face up on the table and the second round of betting occurs. From here on the betting rounds will begin with the small blinder or the immediately active player next to him. After all bets are made, the top card is once again discarded and a new face up card is dealt to the table. This fourth community card is known as the turn. At this stage another round of betting ensues just like in the flop. Finally, the dealer discards one more card from the top and deals the last fifth card, named the river, placing it face up at the center of the table and the final betting round is initiated. When this last betting round is over, the showdown starts. Figure 2.16 shows an example of the table when the showdown is reached.

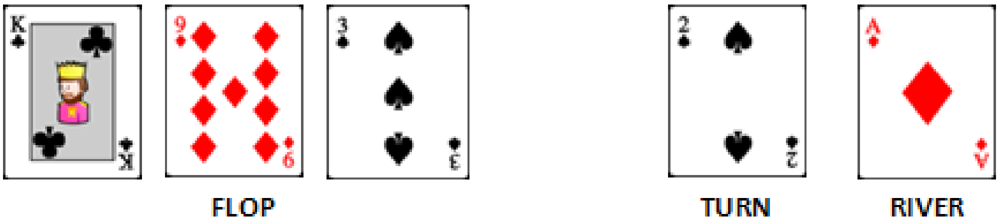


Figure 2.16: Example of the development of table cards

The showdown is a stage where two or more players have ended the betting round of the River and are actively contesting the pot. The rules for this particular stage may require for all hands to be displayed for every player to see regardless of the winning hand being determined, or may look for play history to determine which hand should be shown first. If everyone checked, or is in all-in, on the final betting round, the player who acted first is the first to show the hand. If there is wagering on the final betting round, the last player to take aggressive action by a bet or a raise is the first to show the hand. In this last case, the remaining players may chose not to show their hand but doing so relinquishes their right to contest the pot. In either case, if a tie occurs, the pot is split.

The next figure, figure 2.17, depicts an example of a showdown from [19, pp. 411] between 5 players.

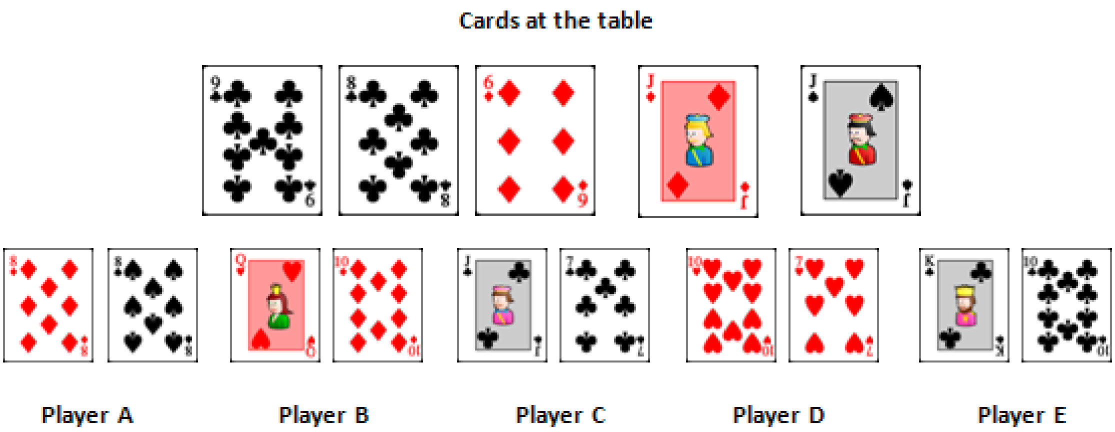
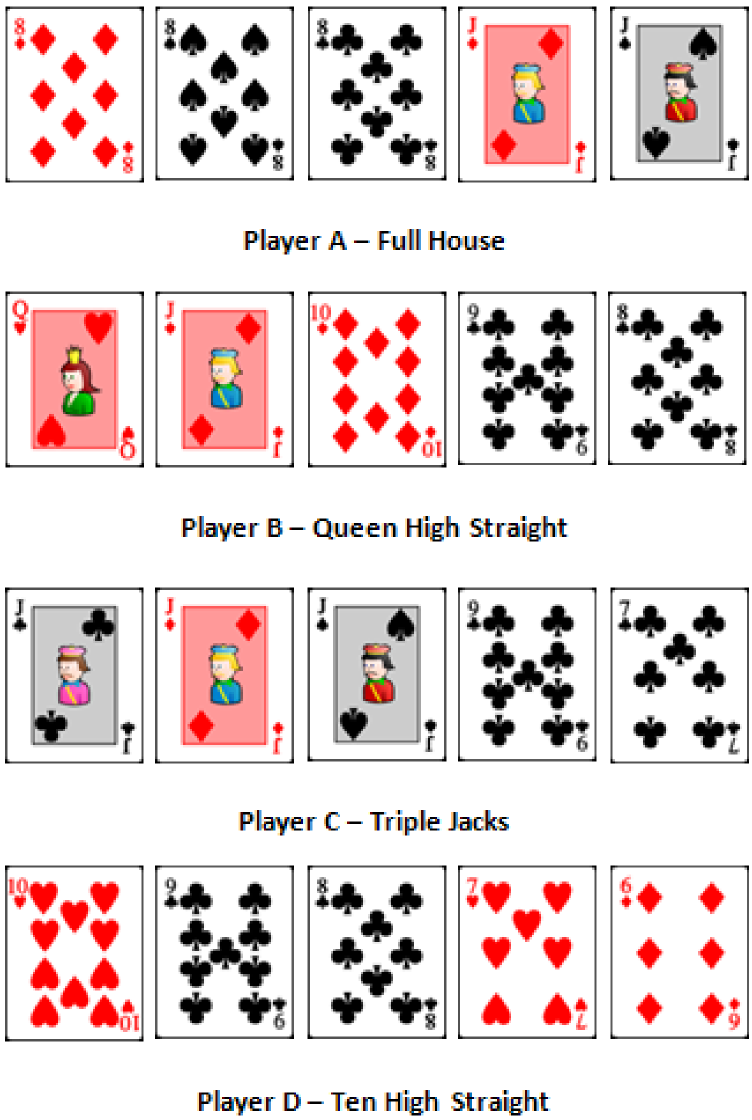
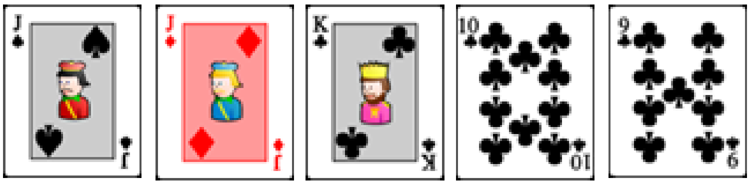


Figure 2.17: Sample showdown

Each player always plays the best five cards out of the seven available cards. The best hand for each player, in this particular scenario, is shown in the next figure.





Player E – Pair of Jacks

Figure 2.18: Final hands at showdown

As it can be seen from figure 2.18 Player A has the highest ranking hand therefore winning the pot. The other players hands ranked in the following manner: Player B had the 2nd best hand, Player D had the 3rd best hand, Player C had the 4th and Player E had the worst hand in comparison to the others.

As a final note on the game mechanics, at anytime there is only one player remaining, from all others having folded, that player is declared winner and is awarded the pot without any player having to reveal his cards.

Also, when only two players remain, the game becomes known as Heads Up. In this situation the player who possesses the dealer's button is also the small blind and the opponent is the big blind. This means that the first player to take action in the pre-flop is the dealer. After that stage that player becomes the last to take action for the rest of the game. The reason for this is to balance the position advantage gained from being at the dealer's position.

2.9 Limit Rules

Limit Texas Hold'em uses a structured betting system where the amount of each bet is strictly controlled in each betting round. There are two denominations of bets, called a small bet and a big bet. On the first two betting rounds of a Limit Texas Hold'em game, all bets and raises are the same value as the small bet, while in the other two betting rounds they are always the value of the big bet. For example if the small bet was 10€ and the big blind was 20€, a player could make a bet of 10€ and then get called by another player or get his bet raised for 10€ making a total wager of 20€. The game would progress without bets until the River card, where the players would now be able to place bets or raises of 20€ instead of 10€. There is, usually, a maximum of three raises allowed per betting round.

2.10 No-Limit Rules

A no-limit betting structure in Texas Hold'em gives it a distinct character from the Limit Poker previously mentioned, requiring a separate set of rules in many situations. No-limit means that the amount of a wager is limited only by the table stakes rule, so any part or all of a player's chips may be wagered. The minimum bet size is the amount of the minimum bring-in, unless the player is going all-in. The minimum bring-in is usually the size of the big blind. This minimum bet remains the same amount on all betting rounds. Because the amount of a wager has such a wide range, it can happen that a player places a bet of a value that other players can't call. Any of the remaining players that wish to call the bet must forcibly go into all-in. In this particular scenario, each all-in player will only contest the pot up to their wager value and not the entire pot. For example, in a game where the pot has 500€, only two players are actively contesting it. Player A has 300€ and player B has 1000€. Player B decides to bet 500€ forcing player A to be all-in, if he chooses to call the bet. The value of the pot being contested by the player A will only be of 1100€, 500€ from the initial pot plus 300€ from his bet plus only 300€ of Player B's bet. The remaining 200€ will be given back to player B if he loses the pot. Player B will be contesting the pot for the full 1300€ wagered.

Also, in no-limit games, the number of raises in any betting round is, usually, unlimited.

2.11 Pot-Limit Rules

Pot-limit games are very similar to no-limit games, following the same rules with the exception of the bet size. In pot-limit games, a bet is not allowed to exceed the pot size. The maximum amount a player can raise is the amount in the pot after the call is made. Therefore, if a pot is 100€ and someone makes a 50€ bet, the next player can call 50€ and raise the pot 200€, for a total wager of €250. Like in limit games, there can be restriction to how many raises are allowed in each betting round.

Contrary to no-limit Texas Hold'em, where the player is the one responsible for determining the pot size and not the dealer, in pot-limit Texas Hold'em the dealer is responsible for determining the pot size and enforcing the pot-size cap on wagers without waiting to be asked to do so.

2.12 Conclusions

Although it is not possible to accurately pin-point the origin of poker, its birthplace can be easily be attributed to the 19th century area surrounding the Mississippi river in the United States of America. More importantly, in over two centuries the game suffered several changes which allow it to evolve into a game of skill rather than just luck.

Along the way many different variations of poker appeared and rode the popularity wave. Despite being different and obeying different sets of rules altogether, most of them remained simple and enjoyable and that is the main reason with this particular game has resisted time and remained popular. Texas Hold'em, in particular, is the example of this. Within the game's simplicity lies a complex textured game structure that is fascinating, sometimes enjoyable, and a real challenge to master.

Chapter 3
State of the Art

"It would appear that we have reached the limits of what it is possible to achieve with computer technology, although one should be careful with such statements, as they tend to sound pretty silly in 5 years."

-- John von Neumann

This chapter will describe current tools, trends, techniques and approaches into building artificial intelligent programs capable of playing poker at a world class level.

3.1 Building a World Class Poker Player

There are mainly three different types of approaches for building artificial poker players: Heuristic-based, Simulation-based and Game Theoretic-based. What these approaches do is to typically provide a specification, a strategy or a policy, on how a player should react in each situation that can arise.

Currently the man-machine poker championship has scored both a win and a loss for computers. In its latest edition, computer program Polaris was able to beat professional poker players in a limit heads-up Texas Hold'em game, tie-in up the score and tipping the momentum scale in favor of computer programs.

3.1.1 Rule Based

A natural first intuition regarding the development of a computer program to play poker is to define a set of conditional if-then-else rules, specifying what action should be taken for each of a large set of situations that can arise. This approach would be similar to how human experts would describe how they play, on a case-by-case basis. For this reason this is also known as a heuristic-based approach.

Although intuitively reasonable, this approach has been proven to be extremely limited [1, pp. 179-181] largely because it remains overly simplistic in theory, since the abstraction of trillions of possible situations onto a much smaller number of general circumstances is unable to capture the subtlety and nuances of a strategically complex game like poker.

3.1.2 Formula Based

Another kind of heuristic-based approach is the formula-based approach. This one typically resorts to an arbitrarily complex formula or procedure to use as criterion for distinguishing cases. For example, it can have a weighted enumeration of sub-cases used to determine a hand’s value and consequent probabilistic action based on predetermined thresholds for that value. This can effectively multiply the number of distinct situations able to be identified, creating a more flexible generalization than what is achieved with those of rule-based [89].

Despite being more flexible, formula-based approaches still rely on the same principle used by rule-based counterpart. As a result, albeit to a lesser extent, they still run into many of the same liabilities of rule-based systems. These are also typically complicated to create and extremely cumbersome to maintain as more situations are abstracted onto the system.

3.1.3 Simulation Based

Simulation based approaches consist in studying the repetition of many instances in order to obtain a statistical average. In particular the Monte Carlo simulation has proven to be a powerful technique, relying on repeated computation and random or pseudo-random numbers, for modeling phenomena with significant uncertainty in inputs like the calculation of risk in business. Of course, with purely random sampling, it can take a long time for the simulation to converge on accurate estimates. In order to accelerate the process a certain degree of biased sampling is usually introduced, like selective sampling which focus on samples that provide the most information gain [6, pp. 31-34] [66, pp. 241-275].

Unfortunately, in practice, this approach can be highly volatile and result in extremely unbalanced play since the quality of the simulations depends, on the quality of the simulated play. This makes it vulnerable to too much biased values for certain situations, leading to inaccurate plays [13, pp. 12-13]. Also, in poker, future actions must be determined not depending on explicit knowledge of opponent's cards in each simulation since observation of perfect information instances cannot, in general, produce accurate results for an imperfect information situation [61, pp. 500-507] [1, pp. 185].

3.1.4 Nash Equilibrium

Game Theory is a branch of mathematics and economics that is devoted to the analysis of games. This is especially relevant since many real-world decision problems can be modeled as games. Nash Equilibrium is a concept developed from Game Theory.

A Nash Equilibrium is a strategy, for each player, with the property that no single player can do better by changing to a different strategy, meaning that no player has an incentive to deviate from the purposed strategy because the alternatives could possibly lead to a worse result. [2, pp. 12-13] This implicitly assumes the opposition of perfect players, players that will always do the best possible move, which in real poker, is definitely not the case since players are highly fallible. Nevertheless, a Nash equilibrium strategy represents a great achievement, especially in two-player zero-sum games, like in heads-up poker. If both players are playing an equilibrium strategy, the expected score for both players will be zero. If only one player is playing the equilibrium strategy he can expect to do no worse than to tie the game, since the opponent cannot do better by playing a strategy other than the equilibrium. A thorough description of this game theory solution can be found in [11].

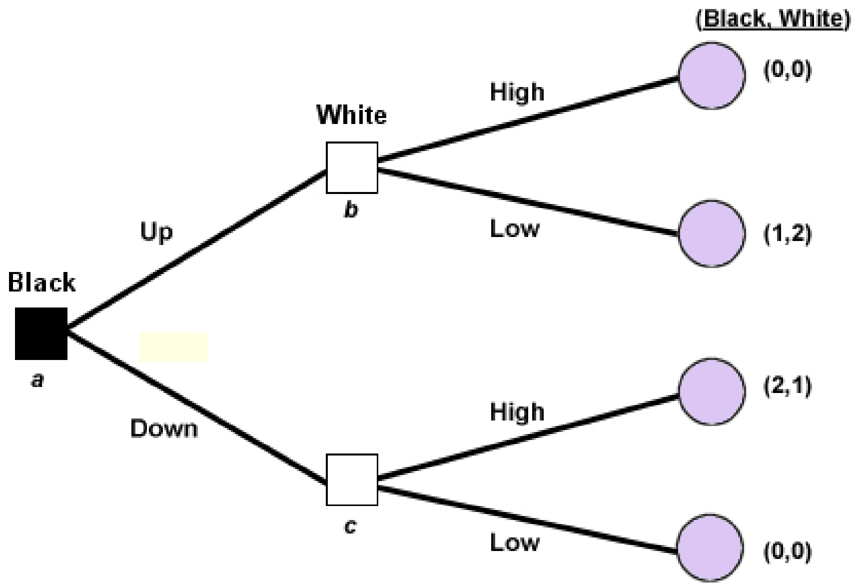


Figure 3.1: Example of a perfect equilibrium subgame. White plays low if Black plays up, and plays high if Black plays down. Each choice is based on its utility value.

The Nash equilibrium problem for two-player zero-sum sequential games can be modeled using linear programming. Unfortunately, creating a perfect Nash equilibrium strategy for a complex game like Texas Hold'em is extremely difficult and computationally unfeasible at the current time. Instead what is currently used is an approximation to the Nash equilibrium strategy, called ε-Nash Equilibrium, resulting in a suboptimal strategy that proposes a best response instead of the perfect response. ε is the value of the best response to the determined suboptimal strategy and is a measure of how far from the actual equilibrium the strategy is. If an opponent, either human or machine, makes mistakes then the equilibrium strategy can win over time.

The success of this approach can be measured by the number of artificial poker players created with this strategy as a core. CPRG has developed a wide range of these players namely the PsOpti series and the Smallbot series. Researchers from Carnegie Mellon University also created a series of artificial players based on ε-Nash equilibrium strategy [41] [43]. Notably, even non-institutions have created these players as is the case with Bluffbot, a creation of Teppo Salonen [49].

Despite the success [9] of this approach, there are implicit problems with it [1, pp. 108-109] [52]. Adding to the already mentioned accuracy tradeoff and the liability against non-optimal players, a Nash equilibrium strategy is a fixed strategy, meaning that despite being more or less time consuming, once a flaw in the strategy is discovered it can be repeatedly exploited. Also, most of these computer poker programs are oblivious to an opponent's strategy and can easily become prey to probing for weaknesses without fear of being punished for using a highly predictable style.

In short, artificial poker players based on ε-Nash equilibrium strategy play close to the optimal strategy, making it near-unbeatable by any other strategy in that abstraction. This is particularly useful since it allows defense against optimal/near-optimal opponent strategies and/or safely learning of an opponent tendencies for several hands before attempting to exploit them. Also there are situations where obtaining a quick best response can compensate for the expected cost of computing the perfect response.

3.1.5 Best Response

The best response approach is based on the paradigm that for every strategy, there is a best possible response. Calculating the best response to the opponent's strategy is very computationally expensive, so an approximation to the best response strategy is usually the solution. This approximation is called abstract game best response. The way this approach works is by choosing the action with the highest value of utility, at every information set, from the probability of the current strategy reaching every terminal node from every possible game state from that point on and the utility value associated with each terminal node. A formal description of this algorithm is presented in [2, chapter 4].

In practical terms what this means is that this approach when facing ε-Nash Equilibrium strategies, for example, is able to determine the value of ε and therefore capable of determining the lower bound on the exploitability of that particular strategy. This is important because poker is a game where exploitation of opponent's weaknesses is crucial. Poker is not about not losing against an opponent but rather making sure one wins the most against an opponent [26, pp. 11-12].

Although promising, the abstract game best response approach has requirements that limit its use in poker games like Texas Hold'em. First, the algorithm requires knowledge of how the strategy acts at every information set; so unless the opponents play in a predictable way or chooses to provide details regarding their own strategy it is difficult to calculate an abstract game best response to an opponent's strategy. Second, the abstract game best response has to be calculated in the same abstraction as the original strategy.

In short, the best response strategy computes the maximizing counter-strategy to a given static strategy. A match between a program and its abstract game best response allows the latter to determine by how much the program can be beaten. Although being a useful tool for evaluating other strategies, by itself its usefulness is limited against arbitrary opponents due to its requirements.

3.1.6 Adaptive Programs

As it should be coming clear by now, a stationary poker playing strategy is easily vulnerable to be exploited. To be able to play poker at a world class level, a player must be able to assess the game's conditions and adjust to any special circumstances. This ability is essential to mastering the game of poker, whether the player is a human or a program.

The adaptive modeling system has two properties at its core: accuracy is limited by the correctness of the relative weights assigned to each future action of the opponent; and by the equity estimations for each showdown outcome [1, pp. 196]. What this translates to is that for every combination of future chance outcomes, the net loss or gain of every future betting sequence is considered, with each being weighted by its relative likelihood. The decisions are made by using an adaptive imperfect information game-tree search, specifically the Miximax and Miximix algorithms [6, pp. 65-68]. The adaptive approach mimics the type of calculations done by human poker players, albeit at a much higher level of precision and accuracy.

In determining the correct mathematical play to make, adaptive programs can be seen as simply computing a best response but on current opponent's beliefs, which are subject to change over time. In principle this is correct but these advanced systems also refrain from continuously using the best response available, deviating from a simple best response for added benefits. The reasons for this are several: avoid predictability, since pure best response without modifications represents highly predictable reactions to certain situations; avoid exploitation, since against a player who is constantly changing styles, the over-reactive best response opponent may swing like a pendulum between belief states; increase unpredictability, since patterns are harder to be identified by opponents; and the pursuit of exploitation, because it is more profitable to exploit an error at a slower sustainable rate, so that the known weakness persist indefinitely, than it is to punish an opponent's error too severely and lead them to change their behavior. [1, pp. 194-199] [26, pp. 27-28]

Although this approach has numerous advantages over the previous approaches, it also has some practical limitations. First, it is mandatory to have good default data since the system allows for essentially any belief to be held, regardless of how irrational it might be. Second, the construction of knowledge is done by observation (of opponents and hands) and this technique requires a considerable amount of data to be effective. Third, data sparsity can also be a problem in some designs since the structure of the imperfect information game tree provides a natural separation of contexts which must be combined according to greatest similarity.

In short, the adaptive programs approach provides a very interesting set of properties that fulfill the requirements in order to build a world class poker player as defined by [5, pp. 2] and have already proven to be successful [1, pp. 166-168] [13, pp. 66-73] [10, pp. 11-13].

3.1.7 Restricted Nash Response

The Restricted Nash Response [2] is a strategy that combines the Nash Equilibrium and the Best Response approaches fairy well. It was designed to solve the tendency to lose to arbitrary opponents found in the best response strategies.

The Restricted Nash Response essentially creates a strategy that is designed to exploit opponents but will do so in a robust way, which is a way to exploit a particular opponent or class of opponents, while minimizing its own exploitability. In practical terms this means that if the strategy being played was trained against the opponent, victory is assured; if the strategy being played was not trained against the opponent then in case of defeat, it will not be by much.

This approach initially uses the Frequentist Best Response algorithm to create a model of the opponent's play and then resorts to the Counterfactual Regret Minimization algorithm to find an ε-Nash Equilibrium [2, pp. 67]. This will generate counter-strategies that provide different tradeoffs between exploitation and exploitability. The generated counter-strategies are in the set of ε-safe best responses for the counter-strategy's value of ε, making them the best possible counter-strategies, assuming the model is correct. ε-safe refers to a strategy that cannot be exploited more than ε. Crucial to this approach is also the strategy's parameter $p \in [0, 1]$ that represents the degree of confidence or belief in the accuracy of the model. The higher the p, the more it moves away from the Nash equilibrium.

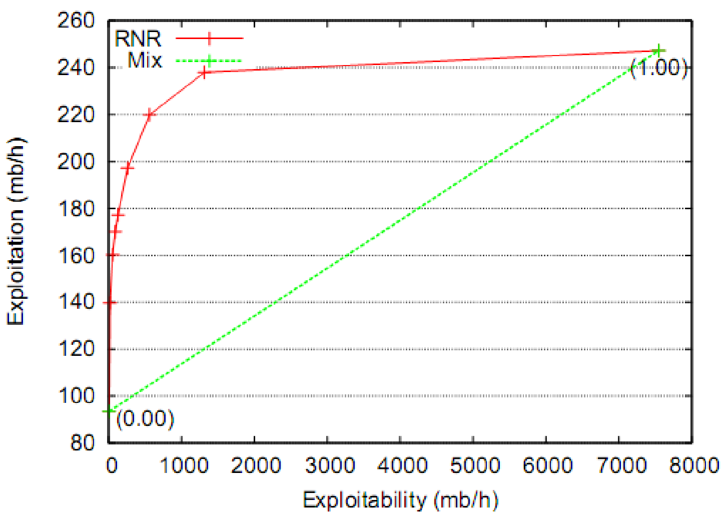


Figure 3.2: A graph from [2, pp. 71] showing the tradeoff between exploitation and exploitability for RNR agents. For any level of exploitability, the RNR counter-strategies exploit the opponent by a larger amount than a mixture with the same exploitability

There are several advantages with this approach. The Restricted Nash Response counter-strategies are robust responses to opponents, unlike traditional best responses which tend to lose against arbitrary opponents. This robustness makes the Restricted Nash Response a good candidate for use against an opponent who is suspect of exhibiting an exploitable trait. This strategy is also more efficient, capable of computing the same information as the Best Response but in a smaller amount of time; and more effective, being capable of achieving nearly the same exploitative power as best responses, but with only a fraction of the exploitability. [2, pp. 66-72]

There are also a couple of limitations with this approach. If the generated model is inaccurate or incomplete due to a limited number of observations of the opponent's actions, the restricted Nash response strategy will perform poorly. This approach is not resilient to the case of no observations being made since it is unlikely that the default policy would accurately reflect the opponent strategy. More importantly, this approach is highly sensitive to the choice of training opponent, requiring a very particular set of observations, including full information observations, in order to perform well. [95, pp. 4-5]

In short, the Restricted Nash Response is an approach for generating a range of strategies that provide good tradeoffs between exploitation and exploitability. The success of this approach has already been proven [2, pp. 70-72] despite its limitations.

3.1.8 Data Biased Response

This recent approach was designed to partly solve the limitations found in the restricted Nash response approach. [95]

The premise is that the selection of only one parameter, p, is not enough to accurately represent the problem since the accuracy of the opponent model is not uniform across all of the reachable information sets, like in the cases of limited or no observations whatsoever.

Instead of choosing only one parameter to reflect the accuracy of the entire opponent model, this approach assigns one probability to each information set I and call this mapping P_{conf}. Whenever the restricted player reaches I, the player will be forced to play according to the model with probability $P_{\text{conf}}(I)$ and choose actions freely with probability $(1 - P_{\text{conf}}(I))$. $P_{\text{conf}}(I)$ is set as a function of the number of observations of the opponent acting in information set I. As the number of observations of the opponent acting in I increase, more confidence is given to the model's accuracy. Noteworthy is the fact that if $P_{\text{conf}}(I)$ is set to 0 for some information sets, meaning that no observations were made, then the opponent model is not used at all and the player is free to use any strategy. Also noteworthy is the inclusion of Bayesian decision functions. [95, pp. 5-7]

In practical terms, comparing to the restricted Nash response, there are several improvements. First, data biased response doesn't require a default strategy. Like it was mentioned if no observation were made, any other strategy can be played although ideally it should revert to a self-play ε-Nash equilibrium strategy. Second, it embodies 'quality-assurance' since it sets a minimum number of observations in order to express any confidence in the model's accuracy while implementing linear and curve confidence functions for a trustworthy assessment of the model's accuracy.

In short Data Biased Response is, currently, the ideal approach for generating counter-strategies, especially since it avoids the outlined shortcomings of the restrictive Nash response approach while providing better performance in the most favorable conditions for the existing approaches [95, pp. 5].

3.1.9 Teams of Computer Programs

This is also a valid approach and one that intuitively sounds promising. When facing an unknown opponent, there is no information regarding which strategy should be used. For instance, an ε -Nash equilibrium strategy is unlikely to lose but it will not win by a large margin. If, for example, a certain program plays better against arbitrary strategies and plays worst against static strategies, and there's another program that plays in the opposite way then it is very likely that using both might improve their overall performance.

This approach was already tested in the 2006 AAAI Computer Poker Competition [29] with an entry by CPRG called Hyperborean06 [2, pp. 35-36]. Essentially this was two different programs that were playing under a shared coach agent. The coach agent would evaluate the game state and decide which program should be used to play each hand. Post-tournament analysis showed that Hyperborean06 performed better than either strategy on its own would have [2, pp. 35-36].

3.2 Relevant Techniques

Throughout this chapter several algorithms and techniques have been mentioned. These are useful, if not critical, in order to build a strong artificial poker player. This section identifies and describes those and other relevant techniques currently being used to build effective artificial poker players.

3.2.1 Neural Networks

An artificial neural network, or neural network, is a popular machine learning data structure based on observation, loosely inspired by biological neural networks. These are often used to model complex relationships between inputs and outputs or to find patterns in data, which in poker context means, trying to predict the opponent's next action in any given scenario.

Consisting on interconnected processing units, called neurons or nodes, these send signals to one another and change their structure depending on the sum of their incoming signals during the learning phase.

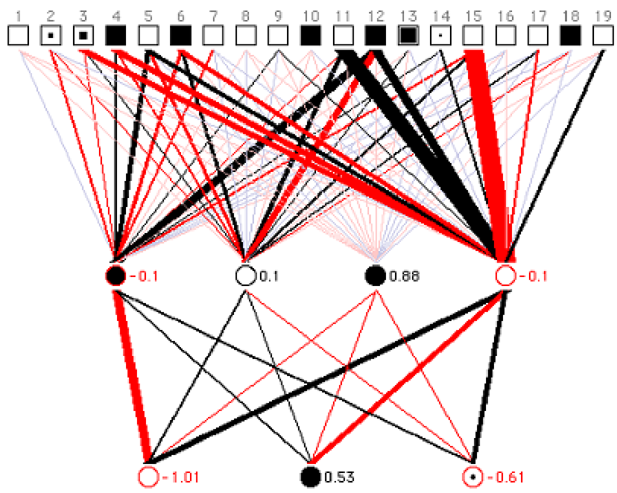


Figure 3.3: A neural network from [96, pp. 22] accurately predicting an opponent's future action, in this case, a call

Neural networks can be built and trained without any domain-specific knowledge. They also provide reasonable accuracy and can be extremely robust. However, they rarely produce better results than a formal system built with specific domain knowledge. Furthermore it can be difficult to extract the knowledge a neural net has learned from a training set [64, pp. 3] [6, pp. 38-39].

The usefulness of poker neural nets [6, pp. 38-43] was demonstrated successfully in the implementation and development of computer poker players [69].

3.2.2 Expectimax

Expectimax was one of the first techniques developed to deal with heuristic search into action-selection for poker. It became the counterpart of the minimax search, a perfect information search technique, used for domains with a stochastic element [1, pp. 110].

The premise was to give up having to determine the best actions for both the player and the opponent within the search and instead consider the game from one player's perspective only. This different problem formulation resulted in a calculation that returns the best way of playing against a specific opponent and not a hypothetical worst-case opponent. This allows parts of the imperfect information game tree to be merged together into the expectimax tree. More specifically, expectimax combined the minimization and maximization nodes of minimax search with something called chance nodes, which is a representation of a stochastic event. The value of a chance node would be determined by the sum of the values of each of the children of that node, weighted by the probability of that event taking place.

Using expectimax search and opponent modeling for decision-making in poker was first explored in Aaron Davidson’s work [6] and latter extended in Schauenberg master’s thesis [13].

3.2.3 Miximix and Miximax

Miximix and Miximax are two variants of the Expectimax search on poker game. These algorithms compute the expected value at each decision nodes of an imperfect information game tree by modeling them as chance nodes with probabilities based on the information known or estimated about the domain and the specific opponent [1, pp. 111].

The expected value calculated at each node of the game tree is based on the predictions of the opponent’s action probabilities at each of his choice nodes and the utility for reaching each terminal node. Once the expected values for each possible action at each information set are determined, one could simply select the option with the maximum value. That is exactly the premise of the Miximax, where the tree contains mixed nodes for the opponent's decisions and max nodes for the strategy’s own decisions. But as it was correctly pointed out [1, pp. 111] this could lead to predictable play. This is precisely the point in Miximix, where the goal is to obtain a high utility value but also to use a mixed strategy to avoid being predictable to the opponent.

Both of these techniques have been widely used in the creation of computer poker players and are the basis for many of the CPRG development designs. Thorough description of both techniques can be found in [1, pp. 110-113] [10].

3.2.4 Frequentist Best Response

This recent technique, introduced and described in [2], is an enhanced version of the best response technique previously described.

The Frequentist best response [2, pp. 56-65] is an offline approach for avoiding some of the drawbacks present in best response. Essentially, for a given opponent strategy this technique observes as many full information examples of that strategy playing, in a non-abstract way, as possible. By full information it is meant that it needs to see all private and public cards of every game, even in cases where one player folds. With this information, it uses frequency counts of its actions to form an opponent model in a specific abstract game.

After enough observations are made, a best response approach is used to calculate a good response counter-strategy to the opponent model. Frequentist best response can calculate a counter-strategy in any abstraction. If a larger (or better) abstraction is used, the performance of the technique's counter-strategies is improved since the larger abstraction gives a more precise representation of the hand strength of the counter-strategy's cards. However, more training data is also expected since there are more information sets that require observation.

The result is a set of counterstrategies that are opponent dependent, which means they are ideal against the opponent intended to beat since it was the one on which the training data was created, and therefore are not good strategies to use in general [2, pp. 64].

3.2.5 Counterfactual Regret Minimization

This technique, fully described in [2], is an enhancement to the ε-Nash equilibrium strategies. As previously described, these approximate to an unbeatable strategy, with the intent that they will do no worse than tie against any opponent.

Regret can be described as similar to the economics concept of an opportunity cost — if you take some action a and receive utility u(a) when you could have taken a utility maximizing action a^* to receive utility u(a^*), then your regret is the difference u(a^*) − u(a). [2, pp. 39]

Based on past results showing that the quality of strategies directly improves as the size of the abstraction used increases, the counterfactual regret minimization [2, pp. 38-53] was developed to use the largest abstraction that fits within the (reasonable) limits of computer memory and time. This technique constructs two strategies that play repeated games of poker against each other. Both start the match with an arbitrary set of action probabilities from all information sets with equal probability and, after each hand, both strategies will adjust their play to minimize their regret. As the number of training games played by the pair of strategies increases, their regret minimizing behavior will cause them to approach Nash equilibrium in the abstract game, going towards unbeatable, making the utility of the best response to a certain strategy decrease.

Contrary to previous ε-Nash equilibrium based techniques, this one requires memory linear in the number of information sets, not game states. This allows solving much larger abstractions than were possible with the previous methods. The result is an ε-Nash equilibrium strategy that is closer to the real game's Nash equilibrium since it uses abstractions two orders of magnitude larger than had been achieved by previous methods [65].

3.2.6 Gameshrink

This is a powerful automated abstraction technique developed in order to tackle the difficult problem of equilibrium computation. Gameshrink is an algorithm for automatically abstracting games in such a way that any equilibrium in the smaller (abstracted) game corresponds directly to an equilibrium in the original game, with the added benefit of being computed drastically faster than the original game.

In regards to Texas Hold'em, this technique is the core for Carnegie Mellon University computer poker player GS1 [31] and the basic principle for the GS2 [32] and GS3 [38]. Specifically, the technique involves a combination of pre-computed optimal strategies, resorting to suit isomorphism, and solving real-time linear programming problems in such a way that the resulting size of the state-space abstraction is manageable by an equilibrium approximation algorithm [54]. Suit isomorphism is an observation technique that has been extensively used in pre-computations, enabling the reduction of the number of distinct hands in a poker game. For example, A♠ A♣ is equivalent to A♦ A♥. This particular technique also applies to many community card histories since 2♠ 3♠ 4♠ 5♠ is equivalent to 2♣ 3♣ 4♣ 5♣, for example.

The result, once mapped back into the original game, is a strategy capable of being competitive against various opponents [31, pp. 5-6]. Unfortunately it also possesses some drawbacks [32, pp. 3] which were the main reason for improving on this technique in subsequent GS artificial poker players.

3.3 Computer Poker Basics

The fundamentals of poker strategy are determined by the probabilistic nature of the game and the psychological understanding of the opponents. For artificial poker players, this is no different.

Artificial poker players, also known as bots, are expected to easily compute "poker's math" accurately. But poker is a mathematical complex game. Specifically, limit Texas Hold'em has a search space size of $O\,(10^{18})$ [1, pp. 107] and the no-limit version is even more complex. In order to accurately determine the correct mathematical play in a reasonable timeframe, it becomes mandatory to reduce the complexity of the game. To that end, researchers have come up with different ways of creating abstractions that make the game simpler to analyze.

This section describes the basic mathematical foundations of poker and the basic abstraction techniques utilized in more advance approaches.

3.3.1 Betting Strategy

Betting strategy in Texas Hold'em poker is usually separated into pre-flop and post-flop. The reason for this is that strategy in the pre-flop is significantly different from the post-flop stages; because no board cards have been dealt (flop) strategy at pre-flop is simpler than later in the game.

For instance, there are $C(52, 2) = 1326$ unique possible hands prior to the flop. Since there are no cards on the board, the suit of the card is irrelevant and many of these hands become equivalent. Using this knowledge there are only 169 distinct hand types in pre-flop Hold'em, which is far less than the possible 1.070.190 distinct combinations of two cards in post-flop stages (1081 possible opponent hands in the flop multiplied by the 990 possible turn and river cards) and this number increases exponentially the more opponents are playing. An exhaustive analysis of all match-ups of a player against nine opponents, in a Texas Hold 'em game, requires evaluating each possible board for each distinct starting hand against each possible combination of hands held by nine opponents, which is more than 21 octillion (approximately $2{,}117 \times 10^{28}$).

For this reason, a popular approach to the pre-flop strategy is to apply an expert system (check Poker Agents section of this work) be it based on empirical knowledge [86, pp. 14] [35, pp. 31, 72, 115] [39, pp. 38-40] or in simulations [6, pp. 20] [1, pp. 41] [26, pp. 261-269].

After the flop, complex algorithms and abstractions are utilized to simplify and determine a hand's value at each stage. Hands are not only ranked against other possible hands the opponents may have but also a hand potential is estimated, aiming to determine the chance that a hand has to improve or deteriorate in the next stages of the game.

3.3.2 Bucketing

Bucketing is a common and successful technique for drastically simplify games [1, pp. 89]. The idea is to partition the possible cards held by a player and on the board into buckets (sometimes called groups or bins) with the intent of separating hands that share similar strategic properties into the same bucket. This is very close to how a human would reason about a given hand. For instance, it doesn't really matter if the hold cards are K♣ 2♦ or K♠ 3♥ since both hands would probably be perceived as (be in the bucket) "a king and a low card". Similarly, one possible approach for bucketing is to divide hands into buckets based on their strength such that weak hands are grouped into low numbered buckets and strong hands are grouped into high numbered buckets.

Assuming an even distribution of hands into buckets, if more buckets are used then each bucket will contain fewer hands. Since all hands in a bucket are played according to the same action probabilities, this leads to fewer cases where a strategy is forced to consider suboptimal action probabilities for a particular hand. [2, pp. 23]

In the bucket abstraction, a strategy is defined over bucket sequences and not over cards. The bucket sequence is the sequence of buckets in which the cards were placed into on each round. For example, following the proposed approach, if a player had a weak hand on the pre-flop but the flop cards made it a strong hand then the hand may have been in bucket 1 on the pre-flop and now be in bucket 5 at the flop stage. To find the probability for every transition either sampling (fast and inaccurate) or enumeration (slow and accurate) techniques can be used. With the probability to move from each bucket at a betting round to another bucket on the next betting round, the model is built and the pseudo-optimal strategy created.

However, this is a one dimensional solution to a multidimensional problem. A hand cannot be categorized completely by only one parameter, so there may be subtle differences between hands in the same bucket that would require different action probabilities. Like all other abstraction techniques this is a compromise solution between the abstracted and the real game.

Nevertheless it is a powerful method of abstraction and advanced bucketing techniques have been developed (for examples see [2, pp. 24-27] [38, pp. 5]) and used successfully in several different artificial poker players.

3.3.3 Opponent Modeling

As stated in section 3.1.6 of this work, '*to be able to play poker at a world class level, a player must be able to assess the game's conditions and adjust*'. Opponent modeling is frequently considered to be a cornerstone of any competitive poker agent, be it human or machine, and rightfully so as it has been corroborated by the success of artificial poker agents (seen section 3.4) as well as demonstrated with dedicated research [24, pp. 66-70] [64] [100, pp. 44-54].

Limited observations, stochastic observations, imperfect information, and dynamic behavior are the main challenges presented to opponent modeling but although these challenges are not yet completely overcome, there are techniques to cope with them. Currently two popular approaches (machine learning models) are used to build an opponent's model in poker: decision-trees and artificial neural networks. Both are based on statistic observation and both require a substantial amount of good data in order to be accurate. These are by no means the only possible successful approaches to agent modeling as related research sugests [98, pp. 36-58] [99, pp. 7-8] [96, pp. 43-50].

3.4 Poker Agents

The number of autonomous artificial poker players, or poker agents for shorter, has been increasing in recent years. Many have been created during, and as a part of, academic research but as this research achieves more and becomes widespread, so does the number of individuals tackling and researching on this subject. This section provides a description and resources for the most popular poker agents at this time.

3.4.1 Loki

Loki was the first poker agent implementation made the University of Alberta CPRG. Loki [12] [68, pp. 2-5] used a probabilistic formula-based approach, incorporating the GNU poker library [76] high-speed hand comparators as a core function, and expert systems designed by the author, to play (differently) each stage of the game. The play is controlled by two components: a hand evaluator and a betting strategy [48, pp. 3-6]. Loki was also developed using realistic games against human opposition. For this purpose the program participated in an on-line poker game [48, pp. 7], running on the Internet Relay Chat (IRC).

Although somewhat successful [68, pp. 5-7] [48, pp. 6-7] Loki had limitations like requiring extensive knowledge, having complicated multi-component systems, being difficult to build on [1, pp. 194-195 and 201] and having low accuracy for opponent modeling. It also wasn't capable of adaptive play, producing a single playing style.

3.4.2 Poki

Poki [7] [6, pp. 17-34] is a complete object-oriented rewrite, also by the CPRG, of the previously mentioned Loki program. It is described as a simulation based system, which consists of playing out many likely scenarios, keeping track of the expected value of each betting action. Like Loki, it used expert system to guide the betting strategy but had neural networks introduced to improve its opponent modeling abilities [1, pp. 60-61] and featured the miximax and miximix techniques to achieve more robust searches on imperfect game-trees.

Poki was also developed / tested resorting to IRC poker play, performing extremely well by consistently winning against human competition [6, pp. 56-58] of intermediate level playing strength. It performed so well that it became licensed for two widely popular commercial products, the video-game Stacked and software trainer Poker Academy.

Poki was designed to play full-ring (up to 10 players) limit Texas Hold'em but despite its success in ring games, in games with few opponents Poki's playing strength decreased significantly, becoming very weak in heads up games. This happened because the program couldn't adapt its strategy fast enough to exploit its opponents or prevent its own exploitation thus becoming too predictable and trusting of its opponents' actions.

And so, it became clear early on that Poki's approach would be inadequate to achieve the goal of surpassing all human players. [1, pp. 182]

3.4.3 PsOpti Family

This is the name of a series of artificial players designed to play heads-up limit Texas Hold'em. Created by the CPRG this series is known for its use of a game-theoretic approach (Nash equilibrium).

The PsOpti family of strategies is created by converting the abstract extensive game into a sequence form. The sequence form can then be transformed as a series of constraints in a linear program [21, pp. 750–759] and be solved to find the approximation to the Nash equilibrium. However, the linear programming required to solve the entire abstract game was considerable and additional simplifications like abstractions and/or separating the game into two phases were used. The techniques used to build this type of agents are described in detail in [9].

This series is currently up to its seventh version (PsOpti0 - PsOpti7). They're differences range from minor improvements over previous versions to play strategically different styles. For example PsOpti4 is less exploitable than PsOpti6 and PsOpti7, but PsOpti6 and PsOpti7 play a strategically different style that is useful against some opponents [2, pp. 16].

Noteworthy is the fact that PsOpti4 and PsOpti6 were combined to form Hyperborean06, the winner of the 2006 AAAI Computer Poker Competition [29]. Also noteworthy is the fact that PsOpti4 was licensed to commercial product Poker Academy under the name of SparBot.

3.4.4 Bluffbot

Bluffbot is a SparBot clone [1, pp. 191] created by an individual developer. According to its author, Bluffbot is a combination of an expert system and a game-theoretic pseudo-optimal strategy [49].

At its core is a hand made approximation of game theoretic equilibrium strategy based on the domain expertise of its creator. Its system includes various plays from professional poker players combined with pseudo-optimal bluffing, bluff catching and value betting strategies, which mean weighted decisions based on expected values. Human testing was also done to optimize the strategies even further [50].

The first version of Bluffbot competed in the first ever AAAI Computer Poker Competition [29] achieving the second place. The latest version, BluffBot 2.0 won the first place in the no-limit Texas Hold'em series of the second AAAI Computer Poker Competition in 2007 [30].

3.4.5 GS Family

The GS series are a Carnegie Mellon University creation. The initial GS1 poker agent [31] made use of a powerful abstraction technique to create an approximation to the GameShrink algorithm and build a competitive player at the time [31, pp. 5-6]. Unfortunately the GameShrink approximation technique was discovered to have major drawbacks [32, pp. 3-4] and there was no statistically significant evidence to demonstrate that GS1 was better or worse than the competition [G3, pp. 2].

GS2 [32] saw the introduction of an improved abstraction algorithm and a method for computing leaf payoffs of truncated games [G2, pp. 3-6]. This allowed their player to achieve the third place in the series competition at the 2006 AAAI Computer Poker Competition [29]. However, several disadvantages were identified [2, pp. 29-31]. First, solving the linear programming in real-time is not a task that can, currently, be computed quickly as intended. Second, because of design decisions, GS2 separated the game into two phases, early and late, which prevented it from having an accurate estimate of the utility of certain lines of play.

Their latest autonomous poker agent, the GS3 [38], introduced a new abstraction algorithm for sequential imperfect information games [34] as well as abstract and game-theoretically analyze all four betting rounds in one run, rather than splitting the game into phases [38, pp. 3-6]. This new approach led them to a second place in no-limit and the third place in the limit Texas Hold'em at the 2007 AAAI Computer Poker Competition [30].

3.4.6 Smallbot Family

Smallbot 1239, 1399 and 2298 are ε-Nash equilibria strategies produced by the University of Alberta CPRG. Their names come from their generation number and number of iterations of the applied algorithm. For instance, Smallbot 2298 is named for being a "second generation bot after 298 iterations of the range of skill algorithm" [94, pp. 791].

Smallbot family bots are based on a published technique [94] called Range of Skill. This method does not directly involve solving a linear program. The main idea is to create a sequence of agents, where each agent can defeat the previous agents in the sequence by at least ε. For any game and value for ε, eventually, the sequence approaches within ε of a Nash equilibrium, and no further agents are able to defeat it by more than ε [2, pp. 31]. Also, this technique allowed for a consistent, whole-game strategy to be created instead of overlapping strategies than split the game into different phases.

Smallbot 1239 and 1399 are considered weaker than PsOpti4 poker agent [2, pp. 16] but Smallbot 2298 performed quite well when tested, even beating all competitors present in 2006 AAAI Computer Poker Competition [94, pp. 791-792].

3.4.7 Vexbot

Vexbot [10] was the first successful adaptive player to be created by the University of Alberta CPRG. This artificial player's strategy core was to build an opponent model online (while playing) and then, using the miximix search technique, calculate strategies and exploit the opponent model effectively.

To be effective, two types of information were provided by the model: opponent action probabilities at their choice nodes, and the utility for reaching each terminal node. The latter was determined by measuring the frequency of the opponent's actions for each betting history. The former is the product of the size of the pot and the probability of winning. [2, pp. 33]

The ability to adapt their play by learning from the opponent was a great advantage at the time, achieving much success both against human and computer players [10, pp. 12], and an important breakthrough in the construction of a world class poker player. Unfortunately it featured several disadvantages [1, pp. 201]. One of these is was at the start of a match, at a stage where the program didn't know the rules of poker and could develop impossible beliefs about the game [2, pp. 34]. Therefore the requirement for solid effective default data was crucial. Another one was the substantial amount of hands required to be played before an effective opponent model could be generated [1, pp. 154-155].

Noteworthy is the fact that this program was licensed into the commercial software trainer Poker Academy.

3.4.8 BRPlayer

BRPlayer is the successor to Vexbot. It was named so since it plays a best-response strategy. Like its predecessor, BRPlayer doesn't employ a static strategy. It records observations about its opponents' actions, and develops a model of their style of play. It continually refines its model during play and uses this knowledge of the opponent to try to exploit his weaknesses.

Both of these programs share the same action-selection search procedure, mininix. They differ only in the type of opponent model used. Both the BRPlayer and Vexbot use a context tree for modeling their opponent's action frequencies and they both assume chance node outcomes occur with uniform probability. The difference between the two player's models lies in how they model their opponent's showdown information. [13, pp. 73]

Although an improvement, BRPlayer suffered from most of the same disadvantages found in Vexbot [2, pp. 34]. Experiments to test the BRPlayer's performance against human competition were never setup but BRPlayer was a major component in the University of Alberta poker program, Poki-X, in 2005 World Poker Robot Championship [22] match against poker professional player, Phil Laak.

3.4.9 Polaris

Polaris is a set of different poker agents that featured in the Man vs. Machine contest promoted by the University of Alberta. This particular contest of limit heads-up Texas Hold'em opposes several human players against the computer poker player developed by the CPRG.

The first version of Polaris was comprised of agents that used ε-Nash equilibrium, restricted Nash response and an experimental aggressive ε-Nash equilibrium strategy [2, pp. 80-84]. This version featured in 2007's first edition of the Man vs. Machine contest against two professional poker players. Although not victorious in the contest [45] [2, pp. 79-84], post game analysis and player comments about Polaris led the research team conclude that poker agents were quickly approaching world champion levels of play in this game [2, pp. 84-86].

The second version of Polaris, named Polaris II, featured in the second edition of the Man vs. Machine contest, in 2008. This time six professional poker players were invited to outmatch the computer player in a six match game. Polaris performance was the best ever and at the end of the competition [46] [47] Polaris became known as the first artificial poker player to beat a team of professional human players. This version of Polaris was subject to much improvement from the previous version and featured a recently published [81] new technique, based on importance sampling, which greatly improved the accuracy of its estimators [81, pp. 8].

3.5 Poker Tools

Currently there are many tools capable of assisting poker players and researchers. These are available as commercial products or open-source projects, with natures ranging from statistical to empirical to behavioral. In this section, the most prominent examples of these tools are described.

3.5.1 IRC Database

This is a resource tool covering several types of poker, for example 7-Stud, No-Limit Hold'em, Pot-Limit Omaha and so on. From 1995 to 2001, 10 million complete hands of poker were collected using an observing program called Observer that sat in on IRC poker channels and logged the details of every game. This hand database has been made publicly available to further the development of poker artificial intelligence research [58].

This database was crucial for the development of two of the most popular artificial poker agents, Loki and Poki.

3.5.2 DIVAT

This is a post-game analysis tool developed by University of Alberta CPRG to be used in limit heads-up Texas Hold'em poker games. The (Darse) Ignorant Value Assessment Tool (DIVAT) is a system designed to reduce the variance due to stochastic outcomes, which in layman terms means it attempts to remove the luck elements as much as possible, and analyze the quality of decisions in the game of poker.

The DIVAT tool consists of two main components: a baseline strategy [37, pp. 20] and an expected value calculation [37, pp. 30]. The first is used to compare to the player's actual decisions. This baseline acts as a reasonable strategy to expect both players to play. The second is used to estimate the difference between the player's actual decisions and the actions recommended by the baseline strategy.

Although this is a tool of statistical nature, it is statistically unbiased as proved by [62]. This means that the long-term expected value from the DIVAT assessment is guaranteed to match the long-term expected value of money earnings [1, pp. 18]. Even when the sample size is small, the lower variance factor assures that it remains useful in determining which of two players is better.

It should be noted that this tool requires that hand histories being analyzed be full-information. Full-information means that all public and private cards must be known, even if one player folds. It is also important to note that although DIVAT reduces variance, it is not free from it.

Although variants of this tool have been developed [2, pp. 74] [81, pp. 3], a beta version of the original tool is publicly available at [28].

3.5.3 Pokersource Poker-Eval Library

The Pokersource [76] poker-eval library is, probably, the most widely-used poker hand evaluator**.** It's very fast, highly optimized, thoroughly examined for over more than ten years of use and includes support for multiple poker variants like Texas Hold'em, Omaha, 7-Card Stud, Low and High-Low games, etc.

Figure 3.4: Pokersource card mask format

The poker-eval library is implemented in a highly optimized, heavily macro'd C programming language but also provide language mappings for popular high-level languages like .NET, Java and Python. Everything is expressed either as a sequence of bits on which various operations are performed, or as a lookup table from which pre-computed values are stored. Poker hands are represented as a sequence of 52 bits, one for each card in the deck. This abstraction is called a card mask or hand mask and it can be used to store N number of cards, where N is any number between 0 and 52.

Figure 3.4 maps each of the possible 52 cards represented by a single bit in a 64-bit integer. Note that marked by X are 12 bits that are unused.

3.5.4 Poker Stove

A player's equity in a pot is his expected share of the pot, expressed either as a percentage (probability of winning) or an expected value (amount of pot * probability of winning). Poker stove [77] is a free popular poker odds calculator that excels at determining equity in Texas Hold'em.

Poker stove is a calculator which facilitates equity calculation using ranges of hands or hand distributions. What this means is that rather than just being able to calculate how a hand matches up against specific hands, it makes it possible to evaluate our equity against a range of hands. For example, figure 3.5 shows the equity percentage of a hand holding Q♦Q♣ against a range of hands holding any pocket pair of sevens or higher, any A-J suited or any A-K suited, and all A-K off-suited. Both hands are also against a hand holding any possible two card combination.

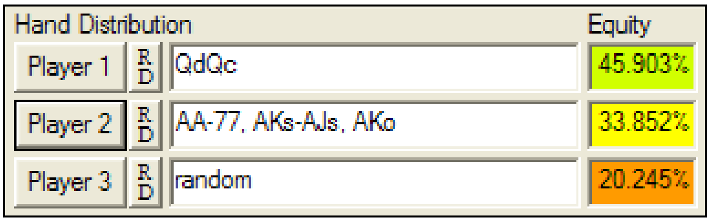


Figure 3.5: Poker stove equity results

Because even professional poker players are only able to seldom know the exact hand that an opponent has, a tool that allows inputting a range of hands instead of a single possible hand, is a very useful program both for training and for post-game analysis.

3.5.5 XPokerEval Library

The purpose of the XPokerEval [93] library is to gather and easily make available, all of the most relevant publically available open-source poker hand evaluators projects. This library groups all these projects into an easy-to-build Visual Studio solution for Windows platforms but also maintains available the original source files in C, C++, C#, Java, and other programming languages, with brief descriptions, sample usage and complete source code.

This library contains 13 of the most popular open source projects regarding poker hand evaluation, including the previously mentioned Pokersource poker-eval and a very promising new evaluator known as Two plus Two (or RayW Hand) evaluator.

This is undoubtedly a very valuable set of tools to any poker researcher or enthusiast with computer programming knowledge.

3.5.6 Poker Academy

Poker Academy is, probably, the world's most popular commercial software trainer for Poker.

Based on the research by the University of Alberta CPRG it is a Texas Hold'em program that simulates the experience of playing in a real online poker room. It simulates both limit Hold'em and no-limit Hold'em, ring and tournament play. Its added value comes from the world renown AI used for its bots specifically Sparbot, Vexbot (for heads-up limit games) and Poki (for ring limit games).

As previously mentioned in section 3.4.7, Vexbot doesn't have a great grasp of basic strategy but it is extremely adaptive to a player's play. Sparbot is the opposite, it doesn't try to adapt to a play style but rather play an almost theoretically perfect game of poker. While it does not play a truly theoretically correct game, it will certainly be more correct than any human can manage. Poki is used to demonstrate computer programs effectiveness in non heads-up games of Texas Hold'em.

For limit play other bots are available, for example Jagbot, a simple basic-strategy player with a complete inability to read opponents, or Simbot, which simulates the outcome of different possibilities in each stage in order to decide a move. For no-limit play, there is also Jambot, based on David Sklansky's "System" from [87, pp. 122] and Oddbot which deliberately makes random plays from time to time.

This software offers many other features but one is noteworthy, the Meerkat API [72]. Meerkat API is a Java API for developing bots in Poker Academy. This means that it is possible to create a custom computer opponent, plug it into Poker Academy and see how it does against both human and other AIs. For this reason, Poker Academy has already been used as a testbed for research on poker AI [67] [16]. The Meerkat API available for download [73] contains the meerkat-api.jar file to compile, API documentation, instructions and a sample agent to start with. Unfortunately, knowledge of programming languages is extremely advisable (if not outright required) to fully use and understand this feature's capabilities.

3.5.7 OpenHoldem

OpenHoldem is an open source screen scraping framework and a programmable logic engine for the online Texas Hold'em poker game.

This project sprung from a functional clone of the commercial software WinHoldem [91]. Made available under the GPL v3 license [33] OpenHoldem project continued the foundation of programmable artificial poker players established by WinHoldEm and soon started to extend and enhance the platform beyond what is provided by its commercial counterpart.

This project features several interesting and valuable tools [71] like a screen scraping and interpreting game states engine which is used to create profiles, also known as table maps, of the client's display; and an environment where various poker situations can be tested out without needing to connect to a live play or real money poker table. The project also features a logic engine for making poker decisions based on game states and table, and a simplistic scripting language (using the Spirit parser library) for describing how poker decisions should be made.

For researchers, there's also the added value of dealing with an open source project and, therefore, having access to source code and a reasonable-sized community.

3.6 Conclusions

This is currently a very exciting and promising time for poker research. Major developments have occurred in the last decade regarding algorithms, techniques and approaches in order to create a world class artificial poker player. Some recent breakthroughs, like solving the game's complexity up to $O\ (10^{12})$, only attest to the success of current lines of research.

The research on the stochastic nature of poker games has revealed the effect of variance in general. Tools and methods to reduce variance, like DIVAT or duplicate poker, have been developed and implemented successful which hold added value for domains outside of science. Whenever someone wants to learn “the truth” in a world beset with randomness, reducing the variance is a critical part of solving the problem.

Many tools have become publicly available and some of those in the form of open source projects with a considerable community supporting them, which can only be seen as the reflection on the interest and success of recent research in this domain. However, even in commercial products, most of the tools available to the general public either require detailed knowledge of computer programming or are simply not extensive enough to provide the creation of a complete strategy to the game. Therefore there is still room for improvement in order to broadening the target audience of these tools.

Meanwhile, several successful artificial poker players have been created. Recently Polaris II, an artificial player created by the University of Alberta CPRG, was able to beat a team of professional human poker players. However, one must not overreact to a computer beating a team of professional poker players in light of current research. Almost all research, with the notable exception of Poki, has been focused on the two-player variant (heads-up) of Texas Hold’em. This is understanding since heads-up (limit) Hold’em is the simplest version of the game and is exactly the kind of game where a computer is expected to excel. No-limit and specially ring play remain very difficult and complex challenges to beat.

Chapter 4
Practical Work

"You insist that there is something that a machine can't do. If you will tell me precisely what it is that a machine cannot do, then I can always make a machine which will do just that"

-- John Von Neumann

As part of this thesis contribution, improvements to the LIACC's Texas Hold'em Simulator were expected. This chapter will describe the work done and the improvements made.

4.1 LIACC's Texas Hold'em Simulator

The Artificial Intelligence and Computer Science Laboratory (LIACC) is a research center at the University of Porto. This center has developed a poker simulator system called LIACC's Texas Hold'em Simulator which was previously used as a testbed for research in poker domain [24] [55].

The system was developed with C/C++ programming languages to implement a standard server-client architecture communicating through TCP/IP and a specific protocol for game related messages. Two versions of the client software were developed, one to allow human players to interact with the system and another for autonomous artificial players running AI poker playing algorithms. The server defines the usual characteristics of a Texas Hold'em game (initial stack of each player, value of the blinds, dealer's button, etc.) and is able to support two up to ten different clients simultaneously. Hand logging and game history are also supported.

The game protocol used is based and made compatible to the one developed for the annual AAAI Computer Poker Competition (AAAICPC) hosted by the University of Alberta Computer Poker Research Group. The server is responsible for maintaining the state of the match. At each change of state, all players receive a match state. They can respond by sending a response message, which echoes the state and the action they wish to take. The game state (described in detail later) contains the information about the visible cards and the betting sequence. The player to act, the state of the pots and the bankrolls are determined from this information.

4.2 Communication Protocol

For compatibility with the AAAICPC protocol (detailed description of this protocol can be found together with AAAICPC server and client example code in [79]) all messages of this protocol are required to be followed by a carriage return and a line feed (ASCII 13 and ASCII 10 symbols).

When a client connects to the server, the first message sent, also known as handshake, indicates the protocol version being used:

`VERSION:2.0.1` (1)

When enough clients are connected, the server can start the game by sending a message to every client with the game characteristics. The message is composed of eight fields, separated by the ':' symbol and described as follows:

`START:Type:MaxPlayer:APlayers:Stack:SBlind:BBlind:Stage` (2)

1. START: This field indicates the type of message.
2. Type: Identifies the type of game being played. 0 is for limit games and 1 is for no-limit games.
3. MaxPlayers: Defines the maximum number of players allowed to be simultaneously connected to the server. Valid values range from 2 to 10.
4. APlayers: Indicates the number of active players (in binary).
5. Stack: This field sets the initial bankroll each player receives in the start of the game.
6. SBlind: Determines the value of the small blind during the game.
7. BBlind: Determines the value of the big blind during the game.
8. Stage: This field sets the limit stage of the simulation. Valid values range from 0 to 3. 1 limits the game to the Flop, 2 limits the game to the Turn, 3 limits the game to the River and 0 doesn't limit the game to any stage.

At each change of state in the game a match state is sent to each player. A match state consists of a hand number, a seat and a game state. All match state messages are sent before any player's action and are sent to all players. An example of such a message is:

`MATCHSTATE:0:26:cc/:|JdTs/2d2c6h` (3)

The example used follows a simple ordered structure of five fields separated by the ':' symbol. The information held in each position of (3) is as described

1. MATCHSTATE: This field indicates the type of message. Used in such way so that future protocols may hold other information.

2. 0: This field indicates the number of the player receiving the message.
3. 26: This field indicates the number of the game being played
4. cc/: This field serves as a mini history of a particular round, separating the actions taken in each betting stage with a '/' symbol. In the example, "cc/" means that 2 calls have been made at the first round of betting. Valid actions are represented by letters. *c* means call or check, *r* means bet or raise and *f* means fold.

`|JdTs/2d2c6h`: This field holds the hand history of the current round. Dealt cards are separated by stages using the '/' symbol. The '|' symbol is used to identify the player who is in the dealer's position. If the '|' appears in the beginning of the field then the player is the dealer in that round. Conversely if '|' doesn't appear then the player is not in the dealer's position. For the given example, player 0 is in the dealer's position, his hole cards are J♦ T♠ and the cards in the flop are 2♦ 2♣ 6♥.

When the players issue a response to the match state, the message format is the same as previously described but with an extra field to indicate the desired action from the player. For example (4), indicates that player 0 is performing a bet or a raise.

```
MATCHSTATE:0:26:cc/:|JdTs/2d2c6h:r    (4)
```

4.3 Conceptual Language

Building computer agents is something that typically requires considerable knowledge and practice with computer programming languages. It is seldom that researchers facilitate the creation of these agents in a particular domain. Although understandable, it proves to be a hamper to technology dissemination and the LIACC's Texas Hold'em Simulator is no exception. A way to avoid this limitation is by creating a high-level language of concepts specific to the domain research which, in this case, is one of poker's variants and then support it with user-friendly interfaces.

Previous related work developed such a language calling it PokerLANG [55, pp. 24-37]. PokerLANG is a language based on the grammar of Coach UniLANG [88, pp. 183-192] and CLANG [15]. These two languages were created for the RoboCup competition in order to allow coaches to change the behavior of simulated soccer players during their games in the Simulated League, one of the available competitions in the RoboCup.

The work presented here is heavily based on the PokerLANG, expanding the language in order to support the creation of broader complex strategies for the autonomous artificial players based on the LIACC's Texas Hold'em Simulator.

4.3.1 Basic Concepts

These are the building blocks, or basic concept definitions, of the proposed language:

```
<HAND>::= <CARD> <CARD>
<CARD>::= <CARD_VALUE> <SUIT>
<CARD_VALUE>::= ace|king|queen|jack|10|9|8|7|6|5|4|3|2
```

```
<SUIT>::= diamonds | hearts | spades | clubs
<SET_OF_HANDS>::= <HAND> | {<HAND>}
<NUM_PLAYERS>::= 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10
<BET_VALUE>::= <INTEGER>
<RAISE>::= <BET_VALUE>
<STACK_VALUE>::= <INTEGER>
<COMP>::= < | > | <= | >= | ==
```

4.3.2 Predictors

Predictors are concepts that extrapolate on hidden information and hold estimations on some specific information. Since it's based on uncertainty, the accuracy of each concept depends on its ability to convert uncertainty into a probability factor. The predictors defined in this language are:

```
<PREDICTOR>::= <IMPLIED_POT_ODDS> | <OPPONENT_HAND> |
               <TYPE_OPPONENT> | <IMAGE_AT_TABLE> | <STEAL_BET>
```

4.3.2.1 IMPLIED_POT_ODDS

The concept of implied pot odds account for the payoff from winning future bets as well as the current pot. In other words, if one has a strong hand that is likely to win, one needs to consider not only the money currently in the pot but also all other bets that opponents must call in future rounds.

```
<IMPLIED_POT_ODDS>::= <REAL>
```

4.3.2.2 OPPONENT_HAND

This is probably the most desirable predictor in poker, one that would allow a player to accurately estimate an opponent's hand. Unfortunately accuracy for this is daunting complex task. An opponent's hand can be influenced by many factors like what type of hands does the opponent usually play, position at the table, size of the stack, etc.

```
<OPPONENT_HAND>::= <HAND> | <SET_OF_HANDS>
```

In order to extrapolate such a prediction, other predictors are also required to play a part in the formula, for example the TYPE_OF_OPPONENT predictor.

4.3.2.3 TYPE_OF_OPPONENT

Like the previous concept, the type of opponent is also used to estimate probable plays from an opponent. Fortunately, this predictor is less complex to be able to estimate with reasonable accuracy. Currently, four categories are used to identify what type of play the opponent is.

```
<TYPE_OF_OPPONENT>::= tight_passive | tight_aggressive |
                      loose_passive | loose_aggressive
```

The chart in figure 4.1 further clarifies how the classifications are made.

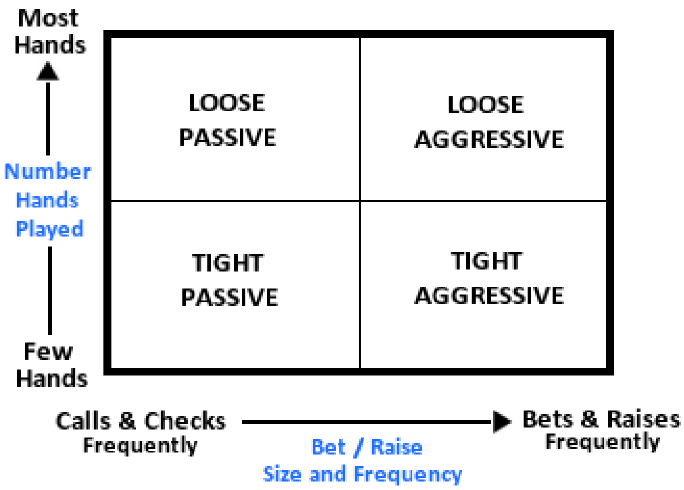


Figure 4.1: Type of opponent categorization

4.3.2.4 IMAGE_AT_TABLE

A player's image at the table is usually a decisive factor in real poker games. As mentioned earlier, deception and exploitation are valid sources of advantage that should/must be utilized. Just as TYPE_OF_OPPONENT allows a player to categorize an opponent, so it can be used to categorize a player himself and identify what opponent might perceive of the player. For instance, if a player is seen as a tight player, bluffs will likely have a higher probability of success. On the opposite, if a player is loose playing, attempts to trap opponents will be successful most of the times. This concept is defined in the same way that the TYPE_OF_OPPONENT predictor but implemented using the information that is made available to other players.

```
<IMAGE_AT_TABLE>::= <TYPE_OF_OPPONENT>
```

4.3.2.5 STEAL_BET

This is based on what is called as stealing in poker, which essentially allows a player to win the pot regardless of his hand strength. This predictor attempts to estimates the amount of chips a given player is required to bet to achieve exactly that. This play, obviously, depends highly on the type of opponents and stack sizes. Its accuracy is also related to the OPPONENT_HAND predictor.

```
<STEAL_BET>::= <BET_VALUE>
```

4.3.3 Evaluators

Evaluators basically hold the information that a player can gather just from observing the game. Contrary to predictors these concepts don't deal with uncertainty and are, therefore easier to define and less complex to implement. Nevertheless, evaluators are crucial to poker strategy and are taken into account on almost every possible play. The evaluators defined in this language are:

```
<EVALUATOR>::=  <NUMBER_OF_PLAYERS>  |  <STACK>  |  <HAND_STRENGTH>  |
                <HAND_REGION_STRENGTH>  |  <POT_ODDS>  |
                <POSITION_AT_TABLE> | <SITUATION_AT_TABLE>
```

4.3.3.1 NUMBER_OF_PLAYERS

This evaluator simply holds the number of players currently playing a hand. This evaluator affects the effectiveness of every move, since most plays take into account the number of opponents a player is currently facing. For example the probability of a given hand remaining strong up to showdown stage, or forcing everyone to fold when playing a given hand, is typically higher if there are fewer opponents than if there are many opponents seeking to win the pot.

```
<NUMBER_OF_PLAYERS>::=<NUM_PLAYERS>
```

4.3.3.2 STACK

This evaluator keeps an updated count of the stack of a given player. Although in poker a stack is usually the amount of chips that a player owns, instead of simply storing the exact number of chips owned, this definition also uses the values of blinds and antes to determine the number of playing units available. Playing units are simply the remaining number of times a player can play a hand given the current game state. The resulting values are in integers, rounded upwards.

```
<STACK>::=<INTEGER>
```

4.3.3.3 HAND_STRENGHT

The HAND_STRENGTH evaluator holds the percentage of a given hand winning against a specific hand. In case of matching against a set of hands, an average percentage value is given regarding the hand strength against all other hands.

```
<HAND_STRENGTH>::= 0 | ... | 100
```

4.3.3.4 HAND_REGION_STRENGHT

Some of the greatest moves in poker are made without true value in the hole cards but more often than not the game dramatically depends on the value of the owned cards. What this evaluator does is to categorize the received hand according to pre-defined expert table or a custom table in order to evaluate their strength. Such table is divided in different regions where each region represents a set of hands, grouped by value.

```
<HAND_REGION_STRENGTH>::= <PREDEFINED_HAND_REGION> |
                          <HAND_REGION_DEFINITION>
<HAND_REGION_DEFINITION>::= <HAND_REGION_NAME> {<HAND>}
<HAND_REGION_NAME>::=[string]
<PREDEFINED_HAND_REGION>::= a | b | c | d | e
```

4.3.3.5 POT_ODDS

This evaluator holds the “return of investment” one has from calling a given bet. This definition expects odds to return a percentage.

```
<POT_ODDS>::= 0 | ... | 100
```

4.3.3.6 POSITION_AT_TABLE

Position is one of the most advantageous parameters to exploit in poker. By position what is usually meant is the place where a given player stands at the table in relation to the small and big blinds. As stated in the section 2.8 of this work, the order of bets is determined by the blinds which mean that if a player is last to act, then he gains advantage since he will bet after all other players have made their moves. This evaluator determines the quality of a player’s position. This definition allows the evaluator to work based on an expert pre-defined region group or any other custom region groups.

```
<POSITION_AT_TABLE>::= <INTEGER>
```

4.3.3.7 SITUATION_AT_TABLE

This is another evaluator which will hold a quality value. In this case, the evaluator will determine the quality of the current financial situation of a given player in regards to the table’s current taxes. Based on a player stack while taking both blinds and antes in consideration, this evaluator will categorize a player’s situation into a pre-defined expert table or into a custom table.

```
<STACK>::= <PREDEFINED_STACK_REGION> | <STACK_REGION_DEFINITION>
<PREDEFINED_STACK_REGION>::= green_zone | yellow_zone | orange_zone |
                             red_zone | dead_zone
   <STACK_REGION_DEFINITION>::= <STACK_REGION_NAME> <STACK_INTERVAL>
<STACK_INTERVAL>::= <MIN_STACK><COMP> <STACK_VALUE> <COMP><MAX_STACK>
<STACK_REGION_NAME>::= [string]
<MIN_STACK>::= <STACK_VALUE>
<MAX_STACK>::= <STACK_VALUE>
```

4.3.4 Actions

There are several playing moves that are available to be used in a poker game. These poker moves are specific ways of handling a hand to achieve a goal. With this definition of action, a player can choose well known pre-defined poker plays or he can create his own.

```
<ACTION>::=  {<PREDEFINED_ACTION><VAR> | <DEFINED_ACTION><VAR>}
<PREDEFINED_ACTION>::=  <STEAL_THE_POT> | <SEMI_BLUFF> |
                        <CHECK_RAISE_BLUFF> | <SQUEEZE_PLAY> |
                        <CHECK_CALL_TRAP> | <CHECK_RAISE_TRAP>
 <DEFINED_ACTION>::= <ACTION_NAME>{<PRE_FLOP_ACTION> | <FLOP_ACTION> |
                                  <TURN_ACTION> | <RIVER_ACTION>}
<PRE_FLOP_ACTION>::= {<BET_VALUE><VAR>}
<FLOP_ACTION>::= {<BET_VALUE><VAR>}
<TURN_ACTION>::={<BET_VALUE><VAR>}
<RIVER_ACTION>::={<BET_VALUE><VAR>}
<VAR>::= 0 | ... | 100
```

All possible betting actions should be accompanied with a value (in percentage) to be used as variance in <VAR> value. The reason for this is to provide less predictability when calling or placing bets since the betting value will seldom be repeated for the same play.

4.3.4.1 STEAL_THE_POT

Stealing the pot is a poker play regularly used by all players. It basically consists in making a large enough bet to force all other players fold, relinquishing possible improvements to their hands in future cards. It's a move that usually requires a good position at the table or the majority of opponents to be tight players. In this definition, this action depends directly from the previously defined predictor STEAL_BET.

```
<STEAL_THE_POT>::= <STEAL_BET>
```

4.3.4.2 SEMI_BLUFF

A semi-bluff is similar to the STEAL_THE_POT play. The main difference is that stealing the pot is a play used when a player doesn't have a likely chance to win in a showdown. With the semi-bluff, a player might not have a good enough hand but the chances of improving and getting the best hand are big. Therefore this action should take in to account the HAND_STRENGTH evaluator and the probability of getting the necessary card(s) to improve the hand.

```
<SEMI_BLUFF>::= <STEAL_BET>
```

4.3.4.3 CHECK_RAISE_BLUFF

The check-raise bluff is a common deceptive play which gives a glimpse of the strategic complexity of a poker game. This is a play usually done when the first player believes that an opponent has an inferior hand and will not call a direct bet, but that he may attempt to bluff, allowing the first player to win more money than he would just by betting straightforwardly. The definition for this action allows a player to specify a list of size bets to use when dependent on the value of the bet to call.

```
<CHECK_RAISE_BLUFF>::= {<BET_TO_CALL><BET_VALUE>}
<BET_TO_CALL>::= <INTEGER>
```

4.3.4.4 SQUEEZE_PLAY

The squeeze play is an advanced delicate play, usually made at pre-flop, where a player late to play performs a strong enough raise to make his opponents fold in light of the potential premium hand he projects to have. A typical situation for a squeeze play will occur when a loose-aggressive opponent opens for a raise pre-flop. Another opponent will then call this raise and the action comes to the player. At this point the player makes a large over-bet or goes all-in, causing both the initial raiser and caller to fold, winning the pot.

```
<SQUEEZE_PLAY>::= <BET_VALUE>
```

This is a situational play which means its success will depends more on the situation at the table instead of card strength. To do a move like this the player must hold a middle to late position and should have a very tight image at the table. Also, the effectiveness of this play stems from the first opponent's loose-aggressive image.

4.3.4.5 CHECK_CALL_TRAP

Trapping is one of the key moves in poker. There are different forms of trapping but all of them have one objective: take the largest possible amount of money from the opponents. The check-call trap is one example of trapping. If a player has the nuts or a very good hand and there's the possibility of opponents having a decent or good hands, the player may give all the action to the adversaries, indicating that he'd missed the flop, and just call all the way for the win.

```
<CHECK_CALL_TRAP>::= check | call
```

4.3.4.6 CHECK_RAISE_TRAP

The check-raise trap is another form of trapping, usually employed after the flop, or in some cases, after the turn. If a player makes a big poker hand, he may choose to deceptively hint he may have missed the flop. If an opponent bets, there's a big chance the player can win the pot with a strong raise revealing the strength of his poker hand. If the opponent throws his hand away, the player just won one more bet than presumably would have had if he'd bet out straightaway. If the opponent calls, there's significantly more money in the pot to be won. Like all trapping moves, this one is also a situational play.

```
<CHECK_RAISE_TRAP>::= check | <RAISE>
```

4.3.5 Strategy and Tactic

Strategies and tactics are the top two definitions from which every other definition springs. A strategy is composed of several tactics which, in turn, is a consequence of the composition of one or more of the previously defined concepts.

```
<STRATEGY>::= {<ACTIVATION_CONDITION> <TACTIC>}
<ACTIVATION_CONDITION>::= {<EVALUATOR>}
<TACTIC>::= <PREDEFINED_TACTIC> | <TACTIC_NAME><TACTIC_DEFINITION>
<PREDEFINED_TACTIC>::= loose_agressive | loose_passive |
                       tight_aggressive | tight_passive
<TACTIC_NAME>::= [string]
<TACTIC_DEFINITION>::={<BEHAVIOUR> <VALUE>}
<BEHAVIOUR>::= {<RULE>}
<RULE>::= {<EVALUATOR> | <PREDICTOR>} <ACTION>
```

As can be seen these two make the *framework* which will hold the remaining concepts, or in other words, the structure of the system.

4.4 Implementation

Up until now, the AI algorithms in the autonomous artificial players supported by the LIACC Texas Hold'em Simulator were hardcoded. The conceptual language previously defined in this chapter paved the way for an easy to use, rule and formula-based system capable of creating artificial poker strategies for the LIACC Texas Hold'em players. These strategies are stored in files instead of hardcoded into players and can be created without computer programming knowledge.

This section describes a possible implementation of the conceptual language PokerLANG. This is by no means the only possible way to achieve this and, if nothing else, the choices presented here might help other researchers make their own decisions in how to and how not to approach a given concept.

4.4.1 Predictors

4.4.1.1 IMPLIED_POT_ODDS

Implied pot odds are not a precise calculation contrary to pot odds. A possible formula to determine this value is depicted in (1). The expected values are essentially *guesses* based on opponent's betting history.

$$\text{implied pot odds} = \frac{money\_in\_pot + expected\_bets\_from\_opponents}{amount\_to\_call + expected\_costs\_with\_future\_betting} \tag{1}$$

4.4.1.2 OPPONENT_HAND

Without an advanced opponent modeling technique, this predictor will be solely based on statistical and history knowledge of opponent's plays. For each hand played until showdown, an entry is made saving the "type" of opponent identified, the opponent's hole cards and betting history, as well as community cards. These entries can then be used in a case-based approach to determine patterns of play and predict the opponent's hand.

4.4.1.3 TYPE_OF_OPPONENT

By monitoring how regularly an opponent plays a hand and how regularly he bets, it is possible to determine what type of player he is. The number of hands played by an opponent identifies if he is a loose player or a tight player; loose means he plays a lot of hands and tight means the opposite.

$$\text{play\_style} = \frac{number\_of\_hands\_played}{total\_number\_of\_hands} \tag{2}$$

The nature of a player's bets (or in this case he's play_style) identifies if he is a passive or an aggressive player; aggressive means that a player raises or bets often while passive means that a player is more prone to just call bets.

$$\text{aggression\_factor} = \frac{bets + raises}{calls} \tag{3}$$

Like the play_style, the aggression_factor is a ratio and not a percentage. A value of 1.0 here implies that the player makes bets about as often as he calls them. Therefore an aggression_factor above 1.0 identifies the player as aggressive. Conversely, an aggression_factor below or equal to 1.0 identifies the player as passive.

4.4.1.4 IMAGE_AT_TABLE

The most straightforward implementation of this is to follow the suggestion regarding the implementation of the TYPE_OF_OPPONENT predictor. The main difference is that instead of monitoring the opponent's statistics and plays, it's the player's that are monitored and used in calculations.

4.4.1.5 STEAL_BET

In implementing this predictor, a limit to the value of the bet should be placed. By using a decimal value to represent fractions of a player stack, one can set the maximum value of this type of play to be half or 1/3 of the stack size or any other value. Determining the amount to bet could be done by first attributing multiplying factors to the type of opponent, for example tight_loose = 3 and tight_aggressive = 5,5, and then by simply calculating:

$$\text{steal\_bet} = amount\_to\_call \times type\_of\_opponent \tag{4}$$

If the steal bet needed at the table is less than the limit, then the predictor is satisfied.

4.4.2 Evaluators

4.4.2.1 NUMBER_OF_PLAYERS

This one should simply monitor the number of active players in any given stage of the game.

4.4.2.2 STACK

Playing units are an accurate measure of the remaining number of times a player can play a hand given the current game state:

$$\text{stack} = \frac{player\_stack\_value}{big\ blind + antes} \tag{5}$$

4.4.2.3 HAND_STRENGHT

A common implementation route for this usually resorts to pre-calculated table lookups. Current state of the art, as mentioned in the Poker Tools section of the previous chapter, points to the use of XPokerEval library resources. The RayW Hand evaluator is one of its resources.

It is based on the abstraction of approximately 2.6 million unique five-card poker hands into just 7462 distinct poker hand values [97]. Resorting to prime numbers for encoding each of the thirteen card ranks, cards are attributed a unique product value, and by utilizing a particular bit scheme, cards can be represented effectively and benefit from bit-wise operations efficiency.

Algorithm to obtain hand value

Require: load of the Lookup Table to memory

```
function getHandValue (set of myCards) return equivalence value of the 7-card hand
    value : int
    currentCard : int
    for j ← 1 to 7 do
    currentCard ← myCards[ j ]
    value ← lookup_at_table (value + currentCard)
    return value
```

Figure 4.2: Algorithm to obtain the equivalence value of a given hand

This particular hand evaluator is able to evaluate an average of one-hundred and forty thousand hands per second (the calculated experimental result was of 142,779,680 hands per second) which places it as arguably the fastest public hand evaluator available at this time. It requires the generation of a particular table which, after generated, will hold over thirty-two million entries (32,487,834 to be precise) and occupy approximately 126 MB of space. The lookup algorithm itself is extremely simple. In order to lookup a given 7-card poker hand, it traces a path through the table, performing one lookup per card. When the last card is reached, the value obtained is the official equivalence value of the hand.

According to the definition for this evaluator, the resulting equivalence value must be converted to a (win) probability percentage.

4.4.2.4 HAND_REGION_STRENGHT

Given the fact that extensive research on poker hand strength has been performed ever since the game started to make an impact on American society, initially by expert poker players and more recently by researchers, poker hands can be accurately divided into groups of similar strength value and thus facilitate the recognition / recollection of strong and weak hands. Popular examples of this are from professional poker player David Sklansky [87] or from Darse Billings's research [1, pp. 43]. Figure 4.3 represent a possible five pre-defined region chart dividing hands according to their strength.

AA, KK, AKs	A
QQ, AK, JJ, TT	B
AQs, 99, AQ, 88, AJs	C
77, KQs, 66, ATs, 55, AJ	D
KQ, 44, KJs, 33, 22, AT, QJs	E

Figure 4.3: Dan Harrington's groups

4.4.2.5 POT_ODDS

There are several different ways to estimate pot odds. For instance, in order to determine the current odds for calling a bet is formulated as:

$$\text{immediate\_pot\_odds} = \frac{amount\_to\_call}{pot\_size + amount\_to\_call} \qquad (6)$$

4.4.2.6 POSITION_AT_TABLE

One possible way to try to measure the quality of a player's position can be given by:

$$\text{position\_quality} = position - (number\_of\_aggressive\_players + number\_of\_tight\_players) \qquad (7)$$

The *position* value can be determined by monitoring how many players will play or have played until it's a player's turn to bet. This implies that later play will receive higher position value. The higher the position_quality value the better.

4.4.2.7 SITUATION_AT_TABLE

In this case, the idea is to determine the quality of the player's current situation at the table. This allows a player to assess if he's operating in a good shape or not. The formula used to determine this could be:

$$\text{situation\_at\_table} = \frac{player\_stack\_value}{small\_blind\_value + big\_blind\_value + antes} \qquad (8)$$

Since this formula can return a large range of values a corresponding qualitative table can be used to better identify the situation the player is currently in. For example:

Table 2: Example of a quality table based on SaT formula

Name	Situation_at_table (SaT) value
Green Zone	SaT >= 40
Yellow Zone	40 > SaT >= 20
Orange Zone	20 > SaT >= 10
Red Zone	10 > SaT >= 5
Dead Zone	SaT < 5

4.4.3 Actions

4.4.3.1 STEAL_THE_POT

This is an action based on the definition of the STEAL_BET predictor. Regardless of what hand a player holds, if a large enough bet is made, probably everyone will fold. Every time this situation happens, that play is stored. After a significant number of hands, it is possible to calculate an average bet that causes this situation with success.

$$\text{steal\_the\_pot} = \frac{\sum bets\_made}{number\_of\_times\_every\_player\_folds\_after\_bet} \tag{9}$$

4.4.3.2 SEMI_BLUFF

The semi-bluff is somewhat similar to the STEAL_THE_POT play. The main difference is that this action should also take in to account the HAND_STRENGTH evaluator and the probability of getting the necessary card(s) to improve the hand.

$$\text{semi\_bluff} = \frac{\sum bets\_made}{number\_of\_times\_every\_player\_folds\_after\_bet \times (100 - HAND\_STRENGTH)} \tag{10}$$

4.4.3.3 REMAINING PRE-DEFINED ACTIONS

The remaining actions have one characteristic in common, which is the fact that they're composed by more than one action themselves. Implementing the remaining actions requires the monitoring of not only the hands history but also combining them with most of the evaluators and predictors described here. A first natural intuition is to simply develop a case-based rule system where each hand being played is monitored and the evaluators / predictors related to the play's definition are determined. The decision could then be made by checking each possible case. For example for the CHECK_RAISE_TRAP:

$if\ HAND\_STRENGHT \geq 90$ (11)

```
then check
```

$if\ previous\_play\ == check\ AND\ opponent\_play\ == bet$

```
then <RAISE>
```

4.4.3.4 REGULAR ACTIONS

The expression 'regular actions' refers to the group of basic actions that is available to every player, namely call, bet, check, raise or fold. The implementation of these requires a value to be passed (in every case except for folding) representing the amount to be wagered. In this implementation the user determines a value to be multiplied by the bet to be matched (in case of pre-flop) or by the value of the pot (in case of post-flop).

4.5 Agent's Architecture

The selected architecture is loosely based on the architecture presented here [96, pp. 27]. The information flows are created from the interactions between the different components in order to commit to the best possible actions in a given situation. In figure 4.4 one is able to see an overview of the architecture.

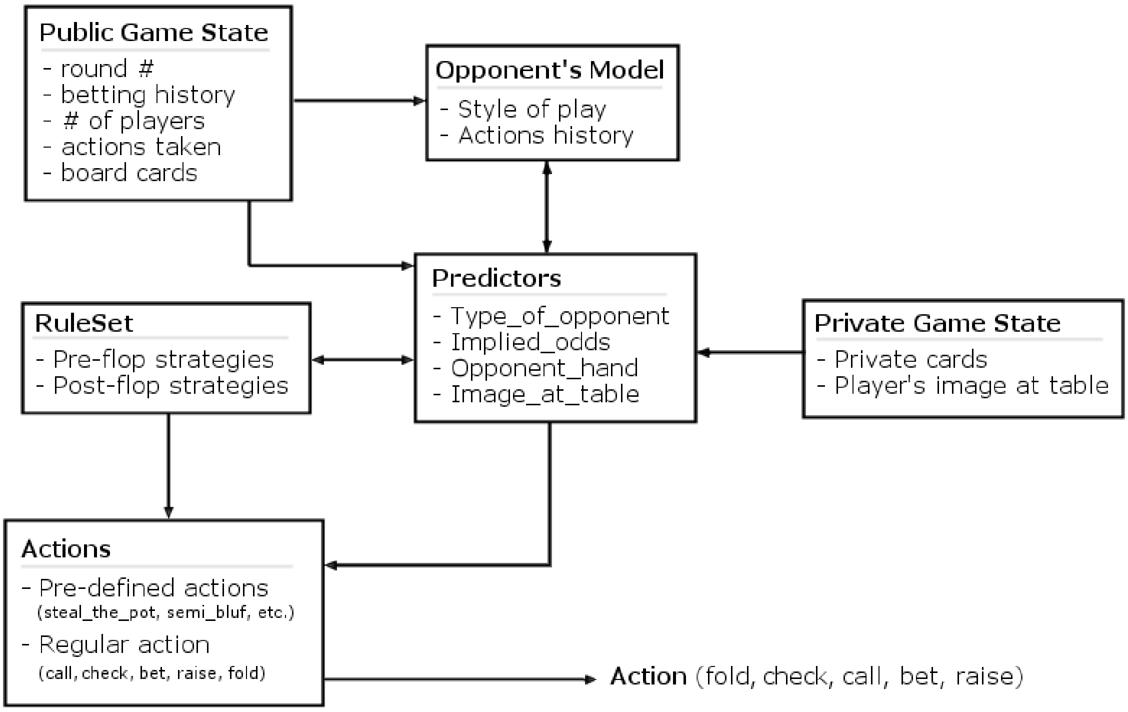


Figure 4.4: High-level agent architecture

As expected, the predictors take a central role in the architecture of the proposed agent. Although not crucial to be able to play, they do have an important role in competitive play. Part of the competitiveness of the agent is also based on the opponent's model being built. The opponent's model despite not being truly adaptive is expected to be able to reflect the current play style of the opponent and for that it requires not only the information from predictors but also from the public game state.

An example of the information flow could be something like:

1. Information is gathered from both the public game state (i.e. number of players) and the private game state (i.e. the agent's private cards) and used to update the value of the predictors.

2. The opponent's model is updated to reflect the opponent's current style (if so predicted).

3. The rule set is checked in order to determine if a particular strategy was planned for the current game state, for example one of the 6 pre-defined actions. If so, then the sequence of planned action(s) is taken into account. If not, then regular actions are performed based on the evaluators.

4.6 Interface

As previously mentioned in this chapter, this implementation is target for the LIACC's Texas Hold'em simulator client. A new button, labeled 'Builder' was added to the client's toolbar and, in doing so, integrated the strategy builder application in the LIACC's poker client for easier packaging, as shown in figure 4.5.

Figure 4.5: LIACC's texas hold'em simulator client main screen

The main interface for the builder application can be seen in the next figure. Figure 4.6 shows the interface designed to create and manage rules / rule sets. The menu is located in the toolbar enabling users to manage their rule sets conveniently. Rule sets should be distinguished between themselves by their name, therefore the <Name> parameter should be changed to a unique value if more than a rule set is desired.

Figure 4.6: Builder main screen

The rules are managed within the Rules section box. Each rule is separated horizontally for easier viewing. Here users can add, change or remove components to a rule. Components are added through the 'add' button. When pressed, a new interface will appear allowing users to select which particular component they want to use, as shown in figure 4.7.

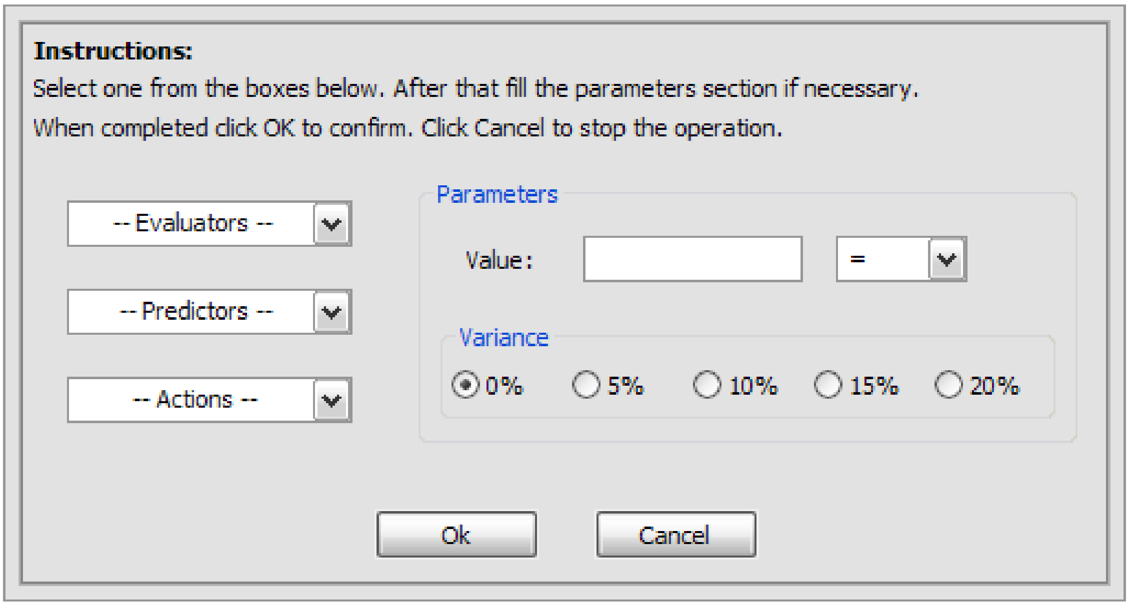


Figure 4.7: Components main screen

Only one of the combo boxes should be used at a time. When pressed, a component's respective list of possible selections is displayed. For example figure 4.8 shows the list of available evaluators that can be chosen.

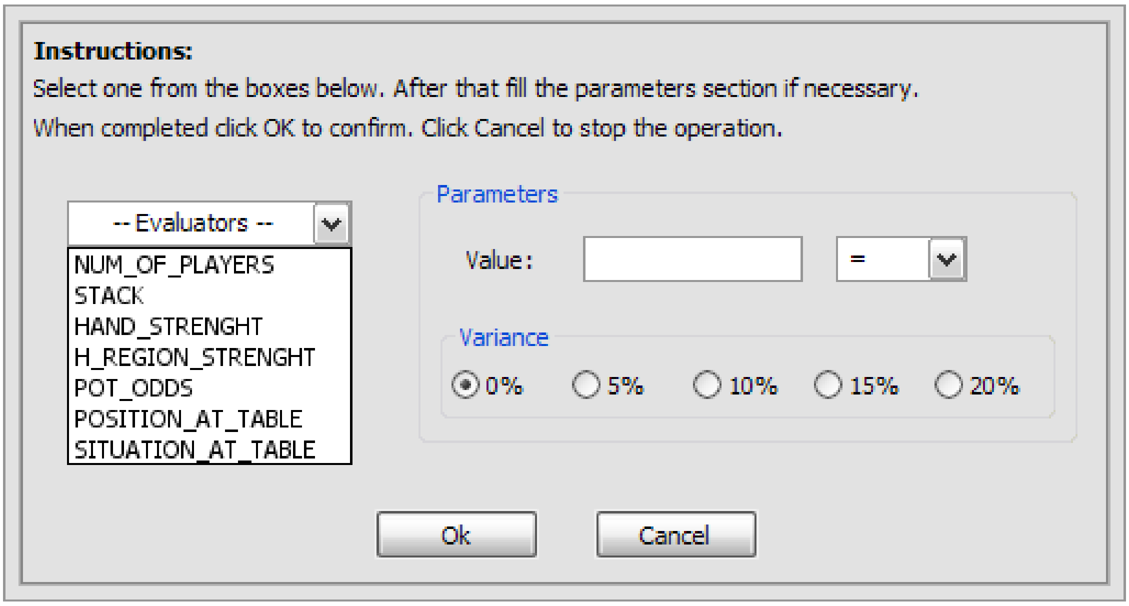


Figure 4.8: Evaluators list

Notice that some of the components require the Parameters box to be completed, namely the value field. For example, if the selected evaluator is HAND_STRENGHT then, like previously defined in section 4.3.3.3, this evaluator requires a percentage win value to be compared to. Following that example, figure 4.9 depicts a possible scenario.

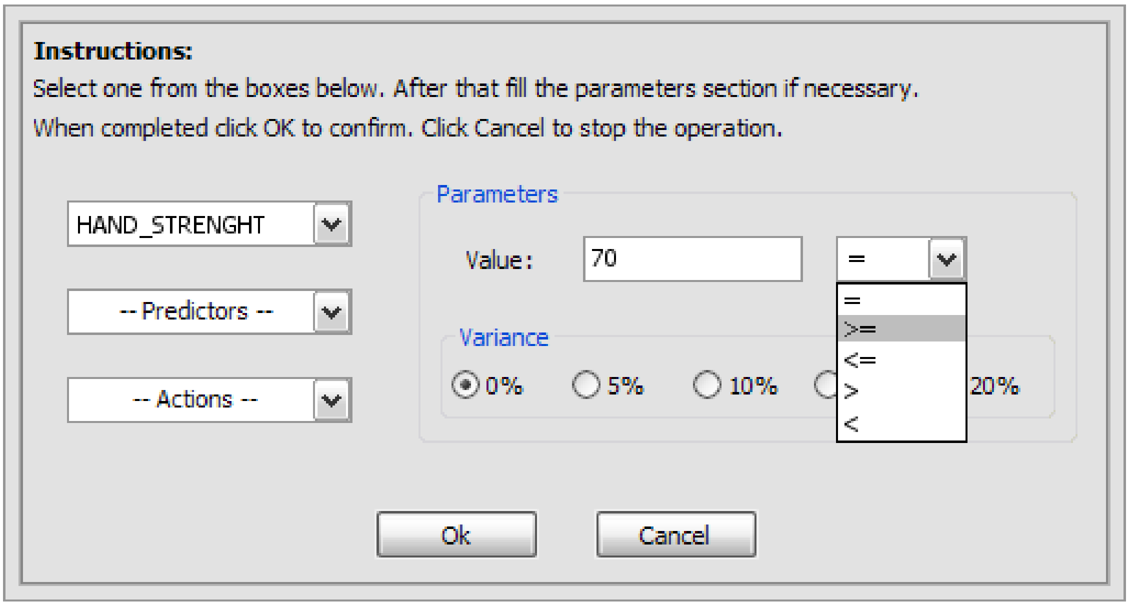


Figure 4.9: HAND_STRENGHT evaluator requires a win probability greater or equal to 70%

Once the component is completely configured, pressing the Ok button will bring the user back to the builder main screen with the appropriate changes. If the Cancel button is pressed, the user is also brought back to the main screen but no changes are made. Figure 4.10 shows the main screen updated according to the examples shown.

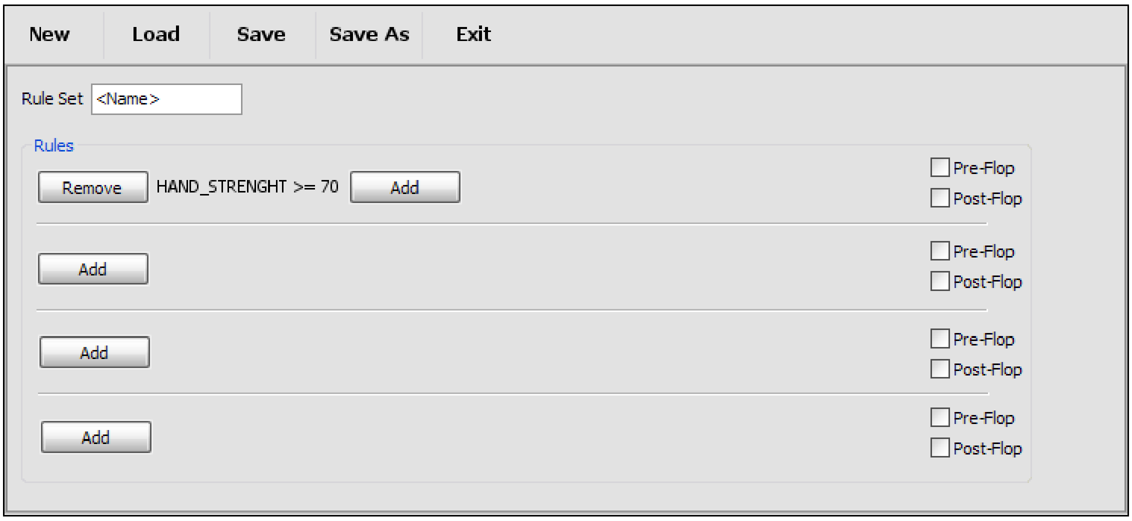


Figure 4.10: HAND_STRENGHT evaluator successfully added to a rule

A rule can be composed of evaluators, predictors and actions, therefore more of these components can be added as shown in figure 4.11.

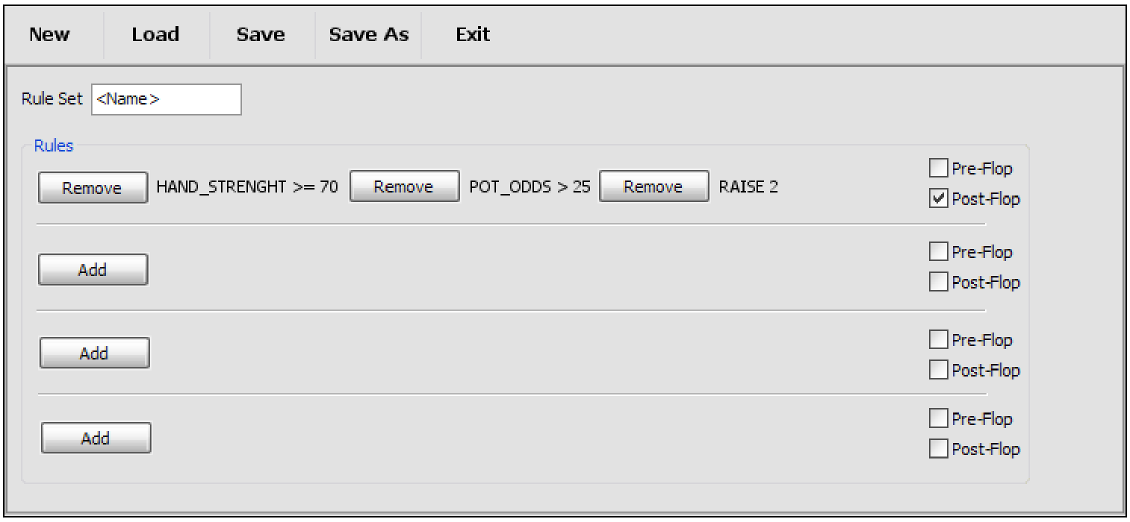


Figure 4.11: Rule completed

Notice that at the end of each rule there are two check-boxes. These are intended to indicate at which stage of the game the rule should be used. Ideally at least one of these check-boxes should be selected otherwise the rule will not be used.

4.7 Conclusions

An immediate benefit of adopting the proposed language of concepts is in improving communication. Every concept is accurately described and despite possible differences in implementation, when discussed, their meaning is less susceptible to misunderstandings.

The language covers all of the basic poker concepts as well as more complex strategy concepts. This provides an excellent support tool when trying to create an application like the strategy builder for LIACC's Texas Hold'em Simulator client.

Unfortunately the current version of the application has several limitations. First, it doesn't possess any kind of mechanism to validate if a rule is or is not well constructed. If a rule is created with only evaluators or predictors, the builder will not signal an error. This means that users are expected to understand the domain in which this application resides. Second, the application doesn't offer any kind of qualitative feedback on the rules themselves. For instance, at pre-flop if the user creates a rule to go all-in every time his hand strength evaluator assesses a very low probability of success (i.e. holding a hand of 2♦7♣) the builder will not inform the user of that highly probable mistake. Third, the application doesn't support the creation of custom actions or custom evaluators although their definition is present in PokerLANG. Fourth, the application only supports three components per rule and a total of four rules per strategy. This implies that truly competitive strategies are nearly impossible in this version of the application.

Nevertheless, the most important aspect to be held is the fact that this application can potential have users creating their strategies like in figure 4.12 instead of having them deal with what is shown at figure 4.13.

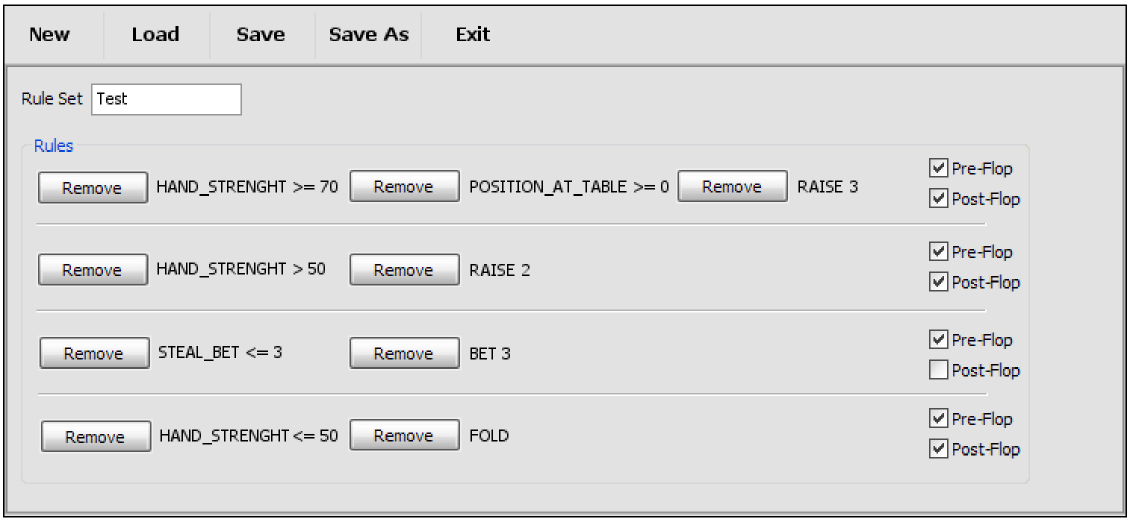


Figure 4.12: Strategy example

The strategy presented in figure 4.12 tries to model a tight-aggressive agent. Most of the time this agent will try to play premium hands only and betting very strong when doing so. It will also try to occasionally steal the blinds and bets at pre-flop if the predicted wager is less than 3 times the amount to call. Every other hand will simply be folded.

Truth be told, this is a static, very straightforward example of a strategy and is expected to be very prone to exploitability. But besides the personal desire, no requirement was made regarding the competitiveness aspect of the agents created by the use of this tool.

```
PokerAgent::PokerAgent(MyThread *trd)
{
    athread=trd;
    srand(time(0));
    //this will hold the 7-card hand to evaluate
    cards=(card *) malloc (7*sizeof(card));
}

PokerAgent::~PokerAgent()
{ free(cards); }

//Method to return which move will be executed
//Return value > 0 --> raise/bet (indicating the amount)
//Return value = 0 --> check
//Return value = -1 --> fold
int PokerAgent::decision(card *pHand,card *board,int round,int n,
                         int money,int min_bet_table,int bet,
                         int playerBet,int position)
{
        int bet_to_call = bet - playerBet;
        int check=0;
        int min=min_bet_table;

        for(int i=0; i<7; i++)
        {
            if(i<2)
                cards[i]=pHand[i]; //private hand
            else
                cards[i]=board[i-2]; //board cards
        }
        if(round==0) //At pre-flop
        {
            float ev = evaluators::hand_Strengh(player);
            int p = evaluators::position_at_table(position);
            if(ev>=70 && p >= 0) //first rule
            {
                if(bet_to_call == 0)
                    return (min * 3);// RAISE 3
                else if(money< bet_to_call)
                    return money;
                else
                    return (bet_to_call * 3);// RAISE 3
            }
            if(ev>50) //second rule
            {
```

Figure 4.13: Snippet of code partially equivalent to the strategy presented in figure 4.12

Chapter 5
Results & Conclusions

"I have not failed. I've just found 10.000 ways that won't work"

--Thomas Edison

5.1 Simulation Scenarios

Two different scenarios were used for experiment and testing. In the first scenario, some of the predictors were evaluated individually. To perform this experiment, two different agents were used: one of a very simple and static nature that would raise in every single hand regardless of what his hold cards were; and another which featured a more dynamic behavior including bluffing and trapping. The former was created so it was purposely predictable and allow for a very straightforward assessment of the predictor performance. The latter would be posing a more challenging scenario and be used to give a more reliable assessment of the predictor. The agent of dynamic nature was previously created in [100] and featured in the current version of the LIACC's Texas Hold'em Simulator. Fifty hands were observed for the first agent and one hundred hands were observed for the second agent.

In the second scenario, the strategy presented in figure 4.12 was used to create an agent (from here on known as "Tighty") to face three other agents in a heads-up format. This scenario would allow the agent to be evaluated as a whole.

5.2 Experimental Results

5.2.1 First Scenario Experiment

As can be seen from figure 5.1, the TYPE_OF_OPPONENT predictor identified the play style of the static agent, as would be expected from such a predictable agent.

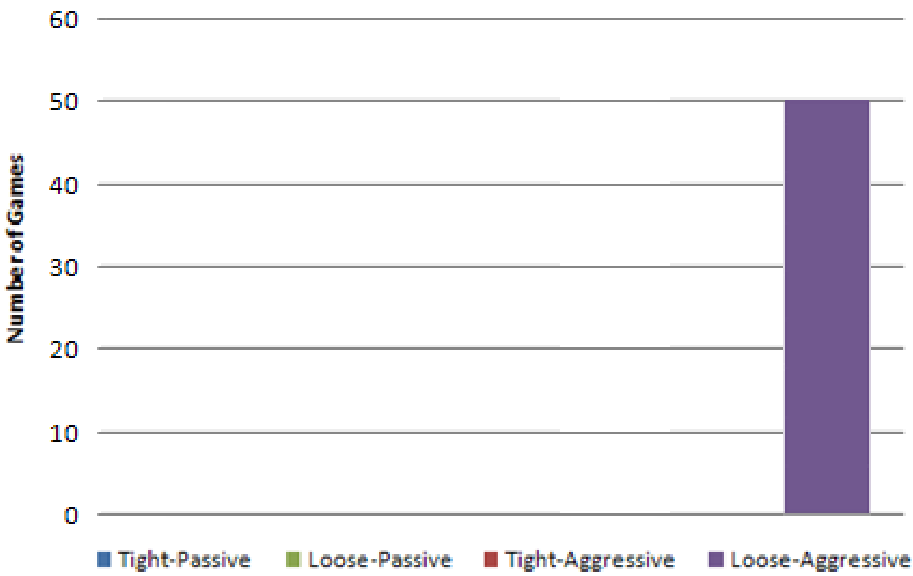


Figure 5.1: TYPE_OF_OPPONENT results facing a static agent

Things seem to hold up even when the opponent presents a more dynamic behavior. According to [100, pp. 33-34] the “Fish” agent is a loose-passive player, which means it will play almost every hand and will be raising with very good hands (like top pairs down to pair of tens and A-K, A-Q and A-J). As it can be seen from figure 5.2 the predictor had the situation well covered.

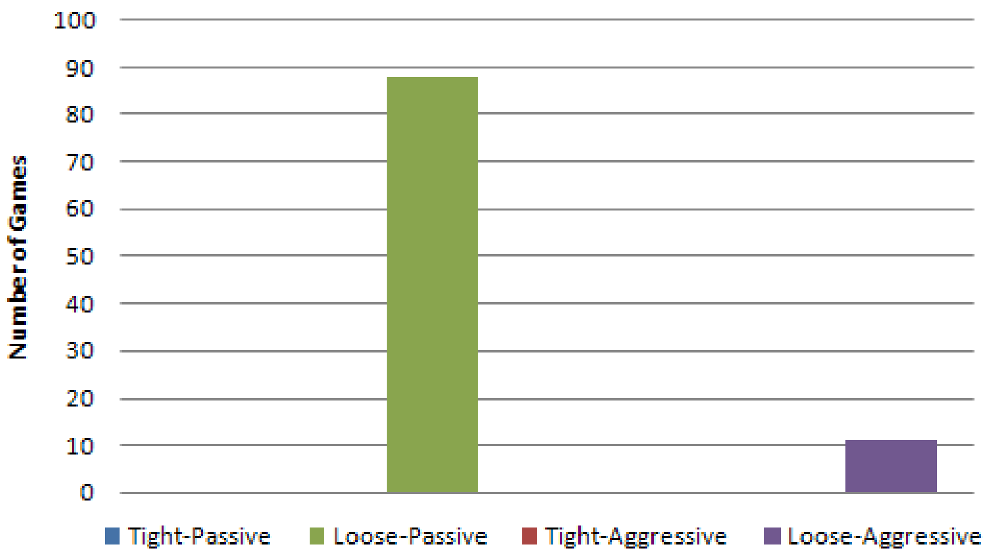


Figure 5.2: TYPE_OF_OPPONENT results facing a dynamic agent

Through analysis of the game's history, it is shown that there was a series of games where the Fish agent raised many hands, which partially explain the results.

The OPPONENT_HAND predictor was also subject to test during the Fish agent observations. Table 3 reveals that the implementation of this predictor, based on history cases only, is unable to accurately predict an opponent's hand given the simulation scenario. One can argue that not enough observations were carried out but this implementation should be reviewed anyway.

Table 3: OPPONENT_HAND prediction results

# of games played	# of hands (hole cards) predicted	# of equivalent hands predicted	# of region hands predicted
100	2	6	10

The implementation of this predictor is solely based on recording the actions made by an opponent and then compare those against future actions by the same opponent to determine patterns. Out of 100 hands, the predictor only got a total of 18 predictions correct. Analysis of the results shows that early to mid games the results were disastrous simply because there was no data to compare to. Only towards the end did the predictor manage to accurately predict the opponent's hand and even so mostly by grouping similar sequence of actions and forming a group of hands to compare to.

5.2.2 Second Scenario Experiment

As mentioned previously in this chapter, three different agents were used in this scenario. The first two agents were the same as the ones used in the first scenario experiment. Figure 5.3 depicts the result of the Tighty against the 'always-raise' agent.

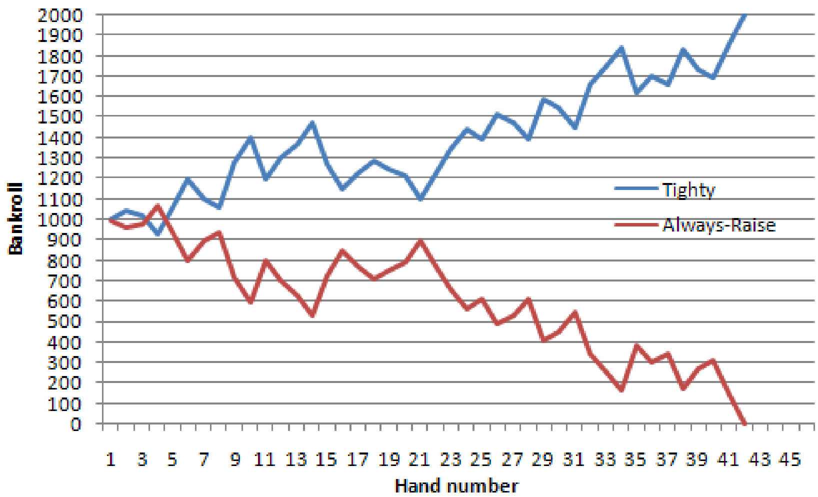


Figure 5.3: Tighty vs 'always-raise' agent

This was a slightly expected result since intuitively one would think that having a complex, although static, strategy would assure a victory over a very basic static strategy. Although this might be true in some cases, in this case, the reason for Tighty's victory is most likely due to its selective play above everything else.

Against the Fish agent, Tighty's selective playing was enough to compete with its opponent. Slowly but steadily, Tighty increased his bankroll and eventually won the game, as shown in figure 5.4.

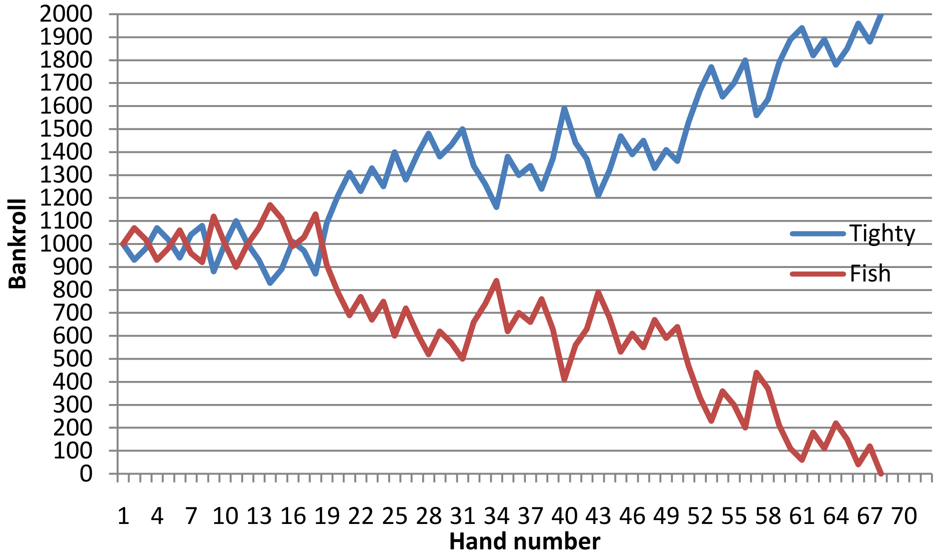


Figure 5.4: Tighty vs fish

The third agent, and last, used in this experiment was appropriately named Random. Random followed no strategy whatsoever. The decision to bet, raise, call, check or fold was based on a random generator. The only limitation was that a maximum bet was enforced to prevent the agent from going all-in in the first hand. This experiment was done to assess the impact of *luck* in this game. Would a completely random strategy provide better results than a selective, although not extremely competitive, strategy?

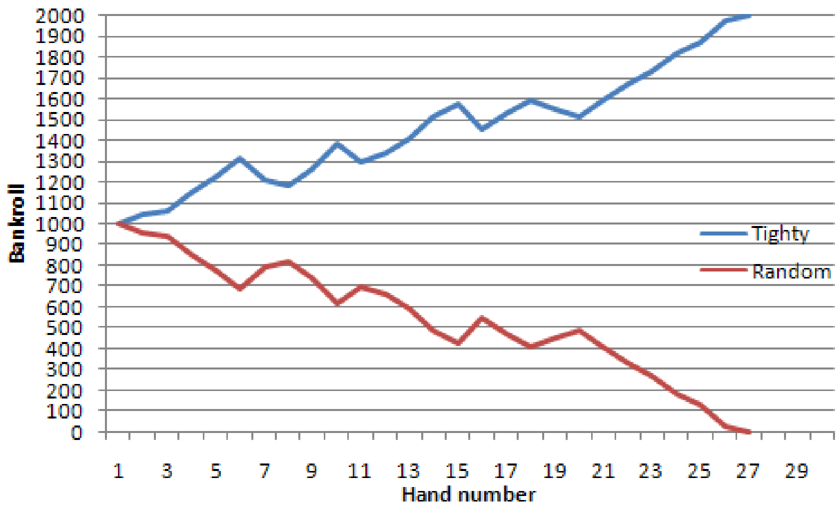


Figure 5.5: Tighty vs random

The impact of variance, commonly known as luck, in this game is undeniable but as figure 5.5 shows, luck will only get you so far.

5.3 Thesis Contributions

The main contribution of this thesis is the definition of a conceptual language in the domain of poker, particularly in the Texas Hold'em variant, capable of describing and supporting a large number of the game's concepts.

The second contribution of this thesis is the creation of a user-friendly application that, with further development, will allow non-IT poker players to create competitive strategies in the LIACC's Texas Hold'em Simulator.

5.4 Conclusion & Future Work

Regarding the strategy builder application, the room for improvement is tremendous. Starting with the limitations indicated in section 4.7, one should tackle those in order to allow more complex strategies to be deployed. But to be truly competitive the agent's modeling techniques should evolve beyond the simple statistic, case-based observations it currently implements. This would allow for the creation of adaptive agents which are far less exploitable than the current static agents being built. Also, the poker language of concepts presented here could also be reviewed and further expanded to cover even more concepts.

It is unquestionable that poker and research on poker are 'on fire', with its popularity fanned by a combination of television, technology and, for some, the allure of big money. Although there's still a long way to go before proclaiming the creation of a world class artificial poker player, current lines of research have been proved successful. There are still entirely new paths and new architectures to be explored which only increases the likelihood that poker will continue to provide a vibrant and demanding domain for researchers.

Bibliography

[1] D. Billings:. *"Algorithms and Assessment in Computer Poker",* University of Alberta, 2006. Available online at http://poker.cs.ualberta.ca/papers/billings.phd.html

[2] M. Johanson:. *"Robust Strategies and Counter-Strategies: Building a Champion Level Computer Poker Player"*, University of Alberta, 2007. Available online at http://poker.cs.ualberta.ca/papers/Papers/johanson.msc.pdf

[3] S. Russell, P. Norvig:. *"Artificial Intelligence, A Modern Approach"*, Pearson Education, 2nd Edition, 2003

[4] K. Korb, A. Nicholson, N. Jitnah:. "*Bayesian poker"*, in Uncertainty in Artificial Intelligence UAI'99, 1999. Available online at https://eprints.kfupm.edu.sa/28067/1/28067.pdf

[5] D. Billings, D. Papp, J. Schaeffer, D. Szafron:. *"Poker as a Testbed for Machine Intelligence Research"*, University of Alberta, 1998

[6] A. Davidson:. *"Opponent Modeling in Poker: Learning and Acting in a Hostile and Uncertain Environment"*, University of Alberta, 2002

[7] D. Billings, A. Davidson, J. Schaeffer, D. Szafron:. *"The Challenge of Poker"*, Artificial Intelligence Journal, 2002. Available online at http://poker.cs.ualberta.ca/papers/AIJ02.html

[8] Andy's Playing Cards:. "*Âs Nas"*, access in June 2009. Available online at http://playingcards.freewebpages.org/cards79.htm

[9] D. Billings, N. Burch, A. Davidson, R. Holte, J. Schaeffer, T. Schauenberg, D. Szafron:. *"Approximating Game-Theoretic Optimal Strategies for Full-Scale Poker",* in Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI-03), 2003, pp. 4-8. Available online at www.cs.ualberta.ca/~darse/Papers/IJCAI03.pdf

[10] D. Billings, A. Davidson, T. Schauenberg, N. Burch, M. Bowling, R. Holte, J. Schaeffer, D. Szafron:. *" Game tree search with adaptation in stochastic imperfect information games"*, University of Alberta, 2004. Available online at http://www.cs.ualberta.ca/~bowling/papers/04cg.ps.gz

[11] F. Oliehoek:. *"Game theory and AI: a unified approach to poker games",* University of Amesterdam, 2005, pp. 18-24. Available online at http://staff.science.uva.nl/~faolieho/docs/ Oliehoek05MScThesis.pdf

[12] D. Papp:. *"Dealing with imperfect information in poker",* University of Alberta, 1998. Available online at http://www.cs.ualberta.ca/~jonathan/Grad/papp/thesis.html

[13] T. Schauenberg:. *"Opponent modeling and search in poker"*, University of Alberta, 2006. Available online at http://poker.cs.ualberta.ca/papers/Papers/schauenberg.msc.pdf

[14] J. Schaeffer, N. Burch, Y. Björnsson, A. Kishimoto, M. Müller, R. Lake, P. Lu, S. Sutphen:. *"Checkers is Solved"*, Science Vol. 317. no. 5844, 2007

[15] M. Chen, K. Dorer, E. Foroughi, F. Heintz, Z. Huang, S. Kapetanakis, K. Kostiadis, J. Kummeneje, J. Murray, I. Noda, O. Obst, P. Riley, T. Steffens, Y. Wang, X. Yin:. *"RoboCup Soccer Server Users Manual for Soccer Server Version 7.07 and later"*, 2002. Available online at http://sserver.sourceforge.net/docs/manual.pdf

[16] R. Andersson:. *"Pseudo-Optimal Strategies in No-Limit Poker",* Umea University, 2006, pp. 29. Available online at http://www.cs.umu.se/~c01ran/poker/report.pdf

[17] D. Brunson:. "*Doyle Brunson's Super System 2 - A Course in Power Poker*", Cardoza Publishing, 2005

[18] J. Cowell:. "*Thirty Years Passed Among the Players in England and America"*, New York: Harper & brothers, 1844

[19] D. Brunson:. "*Super System - A Course in Power Poker*", Cardoza Publishing, 3rd Edition, 2002

[20] J. Schaeffer, R. Lake, P. Lu, M. Bryant:. *"CHINOOK: The world man-machine checkers champion"*, AI Magazine 17, 1996

[21] D. Koller, N. Megiddo, B. Stengel:. *"Fast algorithms for finding randomized strategies in game trees"*, in Annual ACM Symposium on Theory of Computing STOC'94, 1994

[22] Poker Academy:. *"World Poker Robot Championship"* Poker-Academy.com access in June 2009. Available online at http://www.poker-academy.com/wprc

[23] R. Cooke, J. Bond:. *"Cooke's Rules of Real Poker"*, ConJelCo, 2005

[24] J. Ferreira:. *"Opponent Modelling in Texas Hold'em: Learning Pre-flop Strategies in Multiplayer Tables"*, Faculty of Engineering of the University of Porto, 2008

[25] R. Harroch, L. Krieger:. "*Poker for Dummies*", Wiley Publishing Inc., 2004

[26] P. Gordon:. *"Phil Gordon's Little Green Book"*, Simon Spotlight Entertainment, 2005

[27] Wikipedia, the free encyclopedia:. *"Human-Computer Chess Matches"*, Wikipedia.org access in June 2009. Available online at http://en.wikipedia.org/wiki/Human-computer_chess_matches

[28] DIVAT:. *"DIVAT Analysis Service"*, University of Alberta access in June 2009. Available online at http://www.cs.ualberta.ca/~mkan/divat/

[29] M. Littman, M. Zinkevich:. *"The 2006 AAAI Computer Poker Competition"*, ICGA Journal 29, 2006, pp. 166

[30] AAAI Computer Poker Competition:. *“2007 AAAI Computer Poker Competition”* University of Alberta access in June 2009. Available online at http://www.cs.ualberta.ca/~pokert/2007/index.php

[31] A. Gilpin, T. Sandholm:. *“A competitive Texas Hold’em poker player via automated abstraction and real-time equilibrium computation”*, in Association for the Advancement of Artificial Intelligence AAAI’06, 2006. Available online at http://www.cs.cmu.edu/~sandholm/texas.aaai06.pdf

[32] A. Gilpin, T. Sandholm:. *“Better Automated Abstraction Techniques for Imperfect Information Games, with Application to Texas Hold’em”*, in AAMAS’07, 2007. Available online at http://www.cs.cmu.edu/~sandholm/gs2.aamas07.pdf

[33] B. Smith:. *“A Quick Guide to GPLv3”*, Free Software Foundation Inc., 2007. Available online at http://www.gnu.org/licenses/quick-guide-gplv3.pdf

[34] S. Hoda, A. Gilpin, J. Peña:. *“A gradient-based approach for computing Nash equilibria of large sequential games”*, Carnegie Mellon University, 2007. Available online at http://andrew.cmu.edu/user/shoda/1719.pdf

[35] P. Hellmuth:. *“Play Poker Like the Pros”*, Collins Living, 2003

[36] R. F. Foster:. *“Foster’s Complete Hoyle: An Encyclopedia of Games”*, Frederick A. Stokes Company, 1937

[37] M. Kan:. *“Postgame Analysis of Poker Decisions”*, University of Alberta, 2007. Available online at http://poker.cs.ualberta.ca/papers/Papers/kan.msc.pdf

[38] A. Gilpin, T. Sandholm, T. Sørensen:. *“Potential-aware automated abstraction of sequential games, and holistic equilibrium analysis of Texas Hold’em poker”*, in Association for the Advancement of Artificial Intelligence AAAI’07, 2007. Available online at http://www.cs.cmu.edu/~sandholm/gs3.aaai07.pdf

[39] D. Harrington, B. Robertie:. *“Harrington on Hold'em Expert Strategy for No Limit Tournaments, Vol. 1: Strategic Play”*, Two Plus Two Publishing, 2004

[40] M. Malmuth, L. Loomis:. “*The Fundamentals of Poker”*, Two Plus Two Pub, 2000, pp. 3

[41] A. Gilpin, T. Sandholm:. “*A competitive texas Hold’em poker player via automated abstraction and real-time equilibrium computation”*, in Proceedings of the 21st Conference on Artificial Intelligence, 2006. Available online at http://www.cs.cmu.edu/~sandholm/texas.aaai06.pdf

[42] K. Mix:. “*The Game Day Poker Almanac Official Rules of Poker*”, Flying Pen Press LLC, 2007

[43] A. Gilpin, T. Sandholm:. *“Better automated abstraction techniques for imperfect information games, with application to texas Hold’em poker”,* in Autonomous Agents and Multi-Agent Systems, 2007, article no. 192

[44] J. Hildreth:. “*Dragoon Campaigns to the Rocky Mountains: A History Of The Enlistment, Organization And First Campaigns Of The Regiment Of U. S. Dragoons 1836”*, Kessinger Publishing LLC, 2005

[45] Poker Academy:. *"Man vs Machine – Results"*, Poker-Academy.com access in June 2009. Available online at http://www.poker-academy.com/man-machine/results.php

[46] Poker Academy:. *"Man vs Machine 2008 – Results"*, Poker-Academy.com access in June 2009. Available online at http://www.poker-academy.com/man-machine/2008/results.php

[47] University of Alberta:. *"The Second Man-Machine Poker Competition"*, University of Alberta CPRG access in June 2009. Available online at http://www.cs.ualberta.ca/~games/poker/man-machine/Results/

[48] D. Billings, D. Papp, J. Schaeffer, D. Szafron:. *"Opponent Modeling in Poker"*, in Proceedings for the Association for the Advancement of Artificial Intelligence AAAI'98, 1998, pp. 7. Available online at http://poker.cs.ualberta.ca/papers/Papers/AAAI98.pdf

[49] T. Salonen:. *"The official website for bluffbot"*, BluffBot.com, access in June 2009. Available online at http://www.bluffbot.com

[50] PokerAI:. *"Bluffbot – PokerAI"*, PokerAI.org access in June 2009. Available online at http://pokerbots.org/wiki/index.php/Bluffbot

[51] J. Shi, M. Littman:. *"Abstraction methods for game theoretic poker",* CG '00 revised papers from the Second International Conference on Computers and Games, Springer-Verlag, 2002

[52] Y. Shoham, R. Powers, T. Grenager:. *"Multi-agent reinforcement learning: a critical survey"*, Stanford University, Technical Report, 2003

[53] ESPN:. *"World Series of Poker History"*, ESPN.go.com access in June 2009. Available online at http://espn.go.com/eoe/wsop/history.html

[54] A. Gilpin, T. Sandholm:. *"Finding Equilibria in Large Sequential Games of Imperfect Information"*, in ACM Conference on Electronic Commerce EC'06, 2006. Available online at http://www.cs.cmu.edu/~gilpin/papers/extensive.ec06.pdf

[55] P. Mendes:. *"High-Level Language to build Poker Agents"*, Faculty of Engineering of the University of Porto, 2008

[56] J. Nash:. *"Non-Cooperative Games"*, Annals of Mathematics Vol. 54, No. 2, 1951, pp. 286-295. Available online at http://www.scribd.com/doc/7215014/NonCooperative-Games-John-Nash

[57] J. Green:. "*An Exposure of the Arts and Miseries of Gambling"*, G. B. Zieber, 1843

[58] M. Maurer:. *"IRC Poker Database"* University of Alberta CPRG access in June 2009. Available online at http://games.cs.ualberta.ca/poker/IRC/

[59] J. von Neumann, O. Morgenstern:. *"The Theory of Games and Economic Behavior",* Princeton University Press, 2nd Edition, 1947

[60] H. Bohn:. "*Bohn's New Handbook of Games"*, Henry George Bohn, 1850, pp. 384

[61] I. Frank, D. Basin, H. Matsubara:. *"Finding optimal strategies for imperfect information games"*, in American Association of Artificial Intelligence National Conference, AAAI'98, 1998

[62] M. Zinkevich, M. Bowling, N. Bard, M. Kan, D. Billings:. *"Optimal unbiased estimators for evaluating agent performance"*, in American Association of Artificial Intelligence National Conference AAAI'06, 2006. Available online at http://www.cs.ualberta.ca/~bowling/papers/06aaai-divat.pdf

[63] M. Buro:. *"The othello match of the year: Takeshi Murakami vs. Logistello*", International Computer Chess Association Journal 20, 1997

[64] P. McNally, Z. Rafi:. *"Opponent Modeling in Poker Using Machine Learning Techniques"*, Northwestern University, 2008. Available online at http://www.cs.northwestern.edu/~ddowney/courses/349/projects/poker/Poker.pdf

[65] M. Zinkevich, M. Johanson, M. Bowling, C. Piccione:. *"Regret minimization in games with incomplete information"*, in Neural Information Processing Systems Conference NIPS07, 2007

[66] B. Sheppard:. "*World-championship-caliber Scrabble*", Artificial Intelligence 134, 2002

[67] J. Simm:. *"AI System for Online Poker"*, Tallinn University of Technology, 2007, pp. 33. Available online at http://www.lambda.ee/images/f/fb/Poker_net_ai_system.pdf

[68] D. Billings, L. Peña, J. Schaeffer, D. Szafron:. *"Using probabilistic knowledge and simulation to play poker"*, in American Association of Artificial Intelligence National Conference AAAI'99, 1999. Available online at http://www.cs.ualberta.ca/~darse/Papers/AAAI99.html

[69] A. Davidson:. *"Using Artificial Neural Networks to Model Opponents in Texas Hold'em"*, University of Alberta, 1999. Available online at http://spaz.ca/aaron/poker/nnpoker.pdf

[70] BankrollBoost:. "*2008 WSOP Main Event Payouts"*, BankrollBost.com, access in June 2009. Available online at http://www.bankrollboost.com/poker_blog/2008-wsop-main-event-payouts/

[71] OpenHoldem:. *"Open Holdem Bot "*, from Google Code access in June 2009. Available online at http://code.google.com/p/openholdembot/

[72] Poker-Academy Forums:. *"Meerkat API and AI Discussion"* from Poker-Academy.com access in June 2009. Available online at http://forums.poker-academy.com/viewforum.php?f=3

[73] Poker-Academy:. *"Poker Academy – Our Poker Community"* from Poker-Academy.com access in June 2009. Available online at http://www.poker-academy.com/community.php

[74] J. McLeod:. *"Poch/Le Poque"*, Pagat.com access in June 2009. Available online at http://www.pagat.com/stops/poch.html

[75] Académie Universelle des Jeux:. *"Le Jeu Du Poque",* Académie Universelle des Jeux, access in June 2009. Available online at http://coronelle.ovh.org/Accessoires/Les plateaux de jeux/le_jeu_de_la_poque_ou_poch.htm

[76] PokerSource:. *"PokerSource: poker hand evaluator and more"*, PokerSource.sourceforge.net access in June 2009. Available online at http://pokersource.sourceforge.net

[77] PokerStove:. *"Poker Stove: Poker Software and Analysis"*, from PokerStove.com access in June 2009. Available online at http://www.pokerstove.com

[78] D. Parlett:. *"A History of Poker*", Pagat.com access in June 2009. Available online at http://www.pagat.com/vying/pokerhistory.html

[79] AAAI Computer Poker Competition:. *"2009 Computer Poker Competition"*, from University of Alberta CPRG access in June 2009. Available online at http://www.cs.ualberta.ca/~pokert/2009/code.php

[80] R. Ciaffone:. *"Robert's Rules of Poker"*, published online under the Creative Commons License, 2005, pp. 4-8. Available online at http://www.lasvegasvegas.com/pdf/RobertsRules.pdf

[81] M. Bowling, M. Johanson, N, Burch, D. Szafron:. *"Strategy Evaluation in Extensive Games with Importance Sampling"*, in Proceedings of the 25th Annual International Conference on Machine Learning, 2008. Available online at http://www.cs.ualberta.ca/~games/poker/papers/Papers/ICML08.pdf

[82] S. Smith:. "*Theatrical Management in the West and South for Thirty Years"*, Harper & Brothers, 1868

[83] A. Awwad:. "*The History of 7 Card Stud Poker*", EzineArticles.com access in June 2009. Available online at http://ezinearticles.com/?The-History-of-7-Card-Stud-Poker&id=111152

[84] D. Sklansky:. "*The Theory of Poker*", Two Plus Two Publishing, 2005

[85] D. Sklansky:. "*Sklansky on Poker*", Two Plus Two Publishing, 1994

[86] D. Sklansky, M. Malmuth:. *"Hold'em Poker for Advanced Players"*, Two Plus Two Publishing, 1994

[87] D. Sklansky:. *"Tournament Poker for Advanced Players"*, Two Plus Two Publishing, 2002

[88] L. Reis, N. Lau:. *"COACH UNILANG – A Standard Language for Coaching a (Robo) Soccer Team"*, RoboCup-2001: Robot Soccer World Cup V, Springer LNAI Vol. 2377, 2002

[89] S. Schumacher:. "*Probabilistic Versus Deterministic Data Matching: Making an Accurate Decision"*, Information-management.com access in June 2009. Available online at http://www.information-management.com/specialreports/20070118/1071712-1.html

[90] F. R. Wallace:. "*Poker: A Guaranteed Income for Life"*, Crown Pub, 1977, appendix A

[91] R. Bornert:. *"WinHoldEm Home"*, from Winholdem.net access in June 2009. Available online at http://www.winholdem.net

[92] WSOP:. "*World Series of Poker*", WorldSeriesOfPoker.com access in June 2009. Available online at http:///www.worldseriesofpoker.com

[93] J. Devlin:. *"The XPokerEval Library"*, from CodingTheWheel.com access in June 2009. Available online at http://www.codingthewheel.com/archives/poker-hand-evaluator-roundup#xpokereval

[94] M. Zinkevich, M. Bowling, N. Burch:. *"A new algorithm for generating strong strategies in massive zero-sum games"*, in Proceedings of the 22nd Conference on Artificial Intelligence (AAAI-07), 2007. Available online at http://pokerai.org/wiki/papers/AAAI07-125.pdf

[95] M. Johanson, M. Bowling:. *"Data Biased Robust Counter Strategies"*, Proceedings of the 12th International Conference on Artificial Intelligence and Statistics (AISTATS), 2009. Available online at http://www.cs.ualberta.ca/~johanson/papers/dbr_presentation.pdf

[96] R. Sêca:. *"An Intelligent Poker-Agent for Texas Hold'em"*, Faculty of Engineering of the University of Porto, 2008

[97] Cactus Kev:. *"Enumerating Five-Card Poker Hands"*, from SuffeCool.net access in June 2009. Available online at http://www.suffecool.net/poker/7462.html

[98] N. Bard:. *"Using State Estimation for Dynamic Agent Modelling"*, University of Alberta, 2008. Available online at http://poker.cs.ualberta.ca/papers/Papers/bard.msc.pdf

[99] F. Southey, M. Bowling, B. Larson, C. Piccione, N. Burch, D. Billings, C. Rayner:. *"Bayes' bluff: Opponent Modelling in Poker"*, in Proceedings 21st Conference on Uncertainty in Artificial Intelligence, 2008. Available online at http://www.cs.ualberta.ca/~bowling /papers/05uai.pdf

[100] D. Félix:. *"Artificial Intelligence Techniques in Games with Incomplete Information"*, Faculty of Engineering of the University of Porto, 2008

Appendix A: Glossary of Poker Terms

This appendix, adapted from [1], contains informal definitions of common poker terms used in this work and in poker games. More extensive and precise poker glossaries are available on the Internet, such as http://www.seriouspoker.com/dictionary.html, at the online Wikipedia http://en.wikipedia.org/wiki/Poker_jargon, and at http://conjelco.com/pokglossary.html.

- Act. To make a play (bet, call, raise, or fold) at the required time (compare in turn).
- Action Card. In Texas hold 'em or other community card games, a card appearing on the board that causes significant betting action because it helps two or more players. For example, an ace on the flop when two players each hold an ace.
- All-in. To have one's entire stake committed to the current pot. Action continues toward a side pot, with the all-in player being eligible to win only the main pot.
- All-in Equity. The expected value income of a hand assuming the game will proceed to the showdown with no further betting (i.e., a fraction of the current pot, based on all possible future outcomes).
- Ante. A forced bet in which all players put an equal amount of money or chips into the pot before the deal begins.
- Bad Beat. An unlucky loss. In particular, losing a game where the opponent probably should have folded, but instead got extremely lucky to win.
- Bet. To make the first wager of a betting round (compare raise).
- Bet for Value. To bet with the expectation of winning if called (compare bluff).
- Big Bet. The largest bet size in Limit poker (e.g., 20€ in 10€-20€ Hold'em).

- Big Blind (sometimes called the Large Blind). A forced bet made before the deal of the cards (e.g., 10€ in a 10€-20€ Texas Hold'em game, posted by the second player to the left of the button).

- Blind. A forced bet made before the deal of the cards (see small blind and big blind).

- Bluff . To play a weak hand as though it were strong, with the expectation of losing if called (see also semi-bluff and pure bluff, compare bet for value).

- Board (or Board Cards). The community cards shared by all players.

- Board Texture. Classification of the type of board, such as having lots of high cards, or not having many draws (see dry).

- Bottom Pair. A pair (or set) made by matching the lowest-ranking board card with one (or two) of one's private hand (compare top pair).

- Button. The last player to act in each betting round in Texas Hold'em. Also called the dealer button, representing the person who would be the dealer in a home game.

- Call. To match the current level of betting. If the current level of betting is zero, the term check is preferred.

- Cap. (a) The maximum number of raises permitted in any single round of betting (typically four in Limit Hold'em, but occasionally unlimited). (b) To make the last permitted raise in the current betting round (e.g., after a bet, raise, and re-raise, a player caps the betting).

- Check. To decline to make the first wager of a betting round (compare call).

- Check-Raise. To check on the first action, with intention of raising in the same betting round after an opponent bets.

- Community Cards. The public cards shared by all players.

- Connectors. Two cards differing by one in rank, such as 7-6. More likely to make a straight than other combinations.

- Dominated. A Texas Hold'em hand that has a greatly reduced chance of winning against another because one or both cards cannot make a useful pair (e.g., K-Q is dominated by A-K, A-Q, A-A, K-K, and Q-Q, but not by A-J or J-J).

- Draw. A holding with high potential to make a strong hand, such as a straight draw or a flush draw (compare made hand).

- Draw Potential. The relative likelihood of a hand improving to be the best if it is currently behind.

- Drawing Dead. Playing a draw to a hand that will only lose, such as drawing to a flush when the opponent already holds a full house.

- Drawing Hand. A hand that has a good draw (compare made hand).

- Dry. Lacking possible draws or betting action, as in a dry board or a dry game.

- Equity (or Pot Equity). An estimate of the expected value income from a hand that accounts for future chance outcomes, and may or may not account for the effects of future betting (e.g., all-in equity).

- Expected Value (EV) (also called mathematical expectation). The average amount one expects to win in a given game situation, based on the payoffs for each possible random outcome.

- Flop. The first three community cards dealt in Hold'em, followed by the second betting round (compare board).

- Fold. To discard a hand instead of matching the outstanding bet, thereby losing any chance of winning the pot.

- Fold Equity. The equity gained by a player when an opponent folds. In particular, the positive equity gained despite the fact that the opponent's fold was entirely correct.

- Forward Blinds. The logical extension of blinds for heads-up (two-player) games, where the first player posts the small blind and the second player (button) posts the big blind (compare reverse blinds). (Both rules are seen in practice, with various casinos and online card rooms having different policies for multi-player games that have only two active players).

- Free-Card Danger. The risk associated with allowing an opponent to improve and win the pot without having to call a bet (in particular, when they would have folded).

- Free-Card Raise. To raise on the flop intending to check on the turn.

- Game. (a) A competitive activity in which players contend with each other according to a set of rules (in poker, a contest with two or more players). (b) A single instance of such an activity (in poker, from the initial dealing of the cards to the showdown, or until one player wins uncontested).

- Game Theory. Among serious poker players, game theory normally pertains to the optimal calling frequency (in response to a possible bluff), or the optimal bluffing frequency. Both depend only on the size of the bet in relation to the size of the pot.

- Hand. (a) A player's private cards (e.g., two hole cards in Hold'em). (b) One complete game of poker.

- Heads-up. A two-player (head-to-head) poker game.
- Hole Card. A private card in poker (Texas Hold'em, Omaha, 7-Stud, etc.).
- Implied Odds. (a) The pot odds based on the probable future size of the pot instead of the current size of the pot (positive or negative adjustments). (b) The extra money a strong hand stands to win in future betting rounds (compare reverse implied odds).
- In Turn. A player, or an action, is said to be in turn if that player is expected to act next under the rules.
- Joker. A 53rd card used mostly in draw poker games. The joker may usually be used as an Ace, or a card to complete a straight or flush, in high games, and as the lowest card not already present in a hand at low.
- Kicker. A side card, often deciding the winner when two hands are otherwise tied (e.g., a player holding Q-J when the board is Q-7-4 has top pair with a Jack kicker).
- Large Blind (usually called the Big Blind). A forced bet made before the deal of the cards (e.g., 10€ in 10€-20€ Hold'em, posted by the second player to the left of the button).
- Live Game. A game with a lot of action, usually including many unskilled players, especially maniacs.
- Loose Game. A game having several loose players.
- Loose Player. A player who does not fold often (e.g., one who plays most hands at least to the flop in Hold'em).
- Made Hand. A hand with a good chance of currently being the best, such as top pair on the flop in Hold'em (compare draw).
- Maniac. A player who is very aggressive. This type of player plays a lot of hand, raises frequently and often bluffs (see loose player).
- Mixed Strategy. Handling a particular type of situation in more than one way, such as to sometimes call, and sometimes raise.
- Offsuit. Two cards of different suits (also called unsuited, compare suited).
- Open-Ended Draw. A draw to a straight with eight cards to make the straight, such as 6-5 with a board of Q-7-4 in Hold'em.
- Outs. Cards that will improve a hand to a probable winner (compare draw).
- Pocket Cards. See hole cards.

- Pocket Pair. Two cards of the same rank, such as 6-6. More likely to make three of a kind than other combinations (see set).

- Post-flop. The actions after the flop in Texas Hold'em, including the turn and river cards interleaved with the three betting rounds, and ending with the showdown.

- Pot. The common pool of all collected wagers during a game.

- Pot Equity (or simply Equity). An estimate of the expected value income from a hand that accounts for future chance outcomes, and may or may not account for the effects of future betting (e.g., all-in equity).

- Pot Odds. The ratio of the size of the pot to the size of the outstanding bet, used to determine if a draw will have a positive expected value.

- Pre-fop. The first round of betting in Texas Hold'em before the flop, beginning with the posting of the blinds and the dealing of the private hole cards.

- Pure bluff. A bluff with a hand that can only win if the opponent folds (compare semi-bluff).

- Pure Drawing Hand. A weak hand that can only win by completing a draw, or by a successful bluff.

- Raise. To increase the current level of betting. If the current level of betting is zero, the term bet is preferred.

- Raising for a Free-card. To raise on the flop intending to check on the turn.

- Rake. A portion of the pot withheld by the casino or host of a poker game, typically a percentage of the pot up to some maximum, such as 5% up to $3.

- Re-raise. To increase to the third level of betting after a bet and a raise.

- Reverse Blinds. A special rule sometimes used for heads-up (two-player) games, where the second player (button) posts the small blind and the first player posts the big blind (compare forward blinds). (Both rules are seen in practice, with various casinos and online card rooms having different policies for multi-player games that have only two active players).

- Reverse Implied Odds. The unaccounted (negative) money a mediocre hand stands to lose in future betting rounds (compare implied odds (b)).

- River. The fifth community card dealt in Hold'em, followed by the fourth (and final) betting round.

- Satellite. A tournament in which the prize is a free entrance to another (larger) tournament.

- Semi-bluff . A bluff when there are still cards to be dealt, with a hand that might be the best, or that has a reasonable chance of improving to the best if it is called (compare pure bluff).

- Second pair. Matching the second highest community card in Hold'em, such as having 7-6 with a board of Q-7-4.

- Session. A series of games, typically lasting several hours in length.

- Set. Three of a kind, formed with a pocket pair and one card of matching rank on the board. This is a very powerful and well-disguised hand (compare trips).

- Shark. A professional player.

- Short-handed Game. A game with less than the full complement of players, such as a Texas Hold'em game with five or fewer players.

- Showdown. The revealing of cards at the end of a game to determine the winner.

- Side pot. A second pot for the remaining active players after another player is all-in.

- Slow-play. To check or call a strong hand as though it were weak, with the intention of raising in a later betting round (compare smooth-call and check-raise).

- Small Bet. A small bet size in a game of poker.

- Small Blind. A forced bet made before the deal of the cards (e.g., 10€ in a 10€-20€ Texas Hold'em game, posted by the first player to the left of the button).

- Smooth-call. To only call a bet instead of raising with a strong hand, for purposes of deception (as in a slow-play).

- Suck Out. A situation when a hand heavily favored to win loses to an inferior hand after all the cards are dealt (see bad beat).

- Suited. Two cards of the same suit, such as both Hearts. More likely to make a flush than other combinations (compare offsuited or unsuited).

- Table Image. The general perception other players have of one's play.

- Table Stakes. A poker rule allowing a player who cannot match the outstanding bet to go all-in with his remaining money, and proceed to the showdown (also see side pot).

- Texture of the Board. Classification of the type of board, such as having lots of high cards, or not having many draws (see dry).

- Tight Player. A player who usually folds unless the situation is clearly profitable (e.g., one who folds most hands before the flop in Hold'em).

- Tilt. Emotional upset, mental confusion or frustration in which a player adopts a less than optimal strategy, usually resulting in poor play and poor performance.

- Time Charge. A fee charged to the players in a poker game by a casino or other host of the game, typically collected once every 30 minutes.

- Top Pair. Matching the highest community card in Hold'em, such as having Q-J with a board of Q-7-4.

- Trap. To play a strong hand as though it were weak, hoping to lure a weaker hand into betting. Usually a check-raise or a slow-play.

- Trips. Three of a kind, formed with one hole card and two cards of matching rank on the board. A strong hand, but not well-disguised (compare set).

- Turn. The fourth community card dealt in Hold'em, followed by the third betting round.

- Unsuited. Two cards of different suits (also called offsuit, compare suited).

- Value Bet. To bet with the expectation of winning if called (compare bluff).

- Wild Game. A game with lots of raising and re-raising. Also called an action game.